Internet Draft   Patrik Faltstrom
draft-ietf-idn-idna-03.txt   Cisco
July 20, 2001   Paul Hoffman
Expires in six months   IMC & VPNC

Internationalizing Host Names In Applications (IDNA)

Abstract

The current DNS infrastructure does not provide a way to use internationalized host names (IDN). This document describes a mechanism that requires no changes to any DNS server or resolver that will allow internationalized host names to be used by end users with changes only to applications. It allows flexibility for user input and display, and assures that host names that have non-ASCII characters are not sent to DNS servers or resolvers.

1. Introduction

In the discussion of IDN solutions, a great deal of discussion has focused on transition issues and how IDN will work in a world where not all of the components have been updated. Earlier proposed solutions require that user applications, resolvers, and DNS servers to be updated in order for a user to use an internationalized host name. Instead of this requirement for widespread updating of all components, the current proposal is that only user applications be updated; no changes are needed to the DNS protocol or any DNS servers or the resolvers on user's computers.

This document is being discussed on the ietf-idna@mail.apps.ietf.org mailing list. To subscribe, send a message to ietf-idna-request@mail.apps.ietf.org with the single word "subscribe" in the body of the message.

1.1 Design philosophy

Many proposals for IDN protocols have required that DNS servers be updated to handle internationalized host names. Because of this, the person who wanted to use an internationalized host name had to be sure that their request went to a DNS server that was updated for IDN. Further, that server could only send queries to other servers that had been updated for IDN because the queries contain new protocol elements to differentiate IDN name parts from current host parts. In addition, these proposals require that resolvers must be updated to use the new protocols, and in most cases the applications would need to be updated as well.

These proposals would require that the application protocols that use host names as protocol elements to change. This is due to the assumptions and requirements made in those protocols about the characters that have always been used for host names, and the encoding of those characters. Other proposals for IDN protocols do not require changes to DNS servers but still require changes to most application protocols to handle the new names.

Updating all (or even a significant percentage) of the existing servers in the world will be difficult, to say the least. Updating applications, application gateways, and clients to handle changes to the application protocols is also daunting. Because of this, we have designed a protocol that requires no updating of any name servers. IDNA still requires the updating of applications, but only for input and display of names, not for changes to the protocols. Once a user has updated these, she or he could immediately start using internationalized host names. The cost of implementing IDN may thus be much lower, and the speed of implementation could be much higher.

1.2 Terminology

The key words "MUST", "SHALL", "REQUIRED", "SHOULD", "RECOMMENDED", and "MAY" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Structural Overview

In IDNA, users' applications are updated to perform the processing needed to input internationalized host names from users, display internationalized host names that are returned from the DNS to users, and process the inputs and outputs from the DNS.

2.1 Interfaces between DNS components in IDNA

The interfaces in IDNA can be represented pictorially as:

```
                   +-------+
                   | User |
                   +-------+
                      ^
                      |Input and display: local interface methods
                      |(pen, keyboard, glowing phosphorus, ...)
    +-----------------|------------------------------+
    |                 v                              |
    |         +--------------------------+           |
    |         |       Application        |           |
    |         +--------------------------+           |
    |                  ^       ^                     |
    | Call to resolver:|       |Application-specific |
    |   nameprepped ACE|       |protocol:            |
    |                  v       |predefined by the    | End system
    |         +----------+     |protocol or defaults |
    |         | Resolver |     |to nameprepped ACE   |
    |         +----------+     |                     |
    |               ^          |                     |
    +---------------|----------|---------------------+
       DNS protocol:|          |
     nameprepped ACE|          |
                    v          v
         +-------------+    +---------------------+
         | DNS servers |    | Application servers |
         +-------------+    +---------------------+
```

This document uses the generic term "ACE" for an ASCII-compatible encoding. After the IDN Working Group has chosen a specific ACE, this document will be updated to refer to just that single ACE. Until that time, an implementor creating experimental software must choose an ACE to use, such as RACE or LACE or DUDE.

2.1.1 Entry and display in applications

Applications can accept host names using any character set or sets desired by the application developer, and can display host names in any charset. That is, this protocol does not affect the interface between users and applications.

An IDNA-aware application can accept and display internationalized host names in two formats: the internationalized character set(s) supported by the application, and in an ACE. Applications MAY allow ACE input and output, but are not encouraged to do so except as an interface for special purposes, possibly for debugging. ACE encoding is opaque and ugly, and should thus only be exposed to users who absolutely need it.

The optional use, especially during a transition period, of ACE encodings in the user interface is described in section 3. Because name parts encoded with ACE can be rendered either as the encoded ASCII characters or the proper decoded characters, the application MAY have an option for the user to select the preferred method of display; if it does, rendering the ACE SHOULD NOT be the default.

Host names are often stored and transported in many places. For example, they are part of documents such as mail messages and web pages. They are transported in the many parts of many protocols, such as both the control commands and the RFC 2822 body parts of SMTP, and the headers and the body content in HTTP.

In protocols and document formats that define how to handle specification or negotiation of charsets, IDN host name parts can be encoded in any charset allowed by the protocol or document format. If a protocol or document format only allows one charset, IDN host name parts must be given in that charset. In any place where a protocol or document format allows transmition of the characters in IDN host name parts, IDN host name parts SHOULD be transmitted using whatever character encoding and escape mechanism that the protocol or document format uses at that place.

All protocols that have host names as protocol elements already have the capacity for handling host names in the ASCII charset. Thus, IDN host name parts can be specified in those protocols in the ACE charset, which is a superset of the ASCII charset that uses the same set of octets.

2.1.2 Applications and resolvers

Applications communicate with resolver libraries through a programming interface (API). Typically, the IETF does not standardize APIs, although there are non-standard APIs specified for IPv6. This protocol does not specify a specific API, but instead specifies only the input and output formats of the host names to the resolver library.

Before converting the name parts into ACE, the application MUST prepare each name part as specified in [NAMEPREP]. The application MUST use ACE for the name parts that are sent to the resolver, and will always get name parts encoded in ACE from the resolver.

IDNA-aware applications MUST be able to work with both non-internationalized host name parts (those that conform to [STD13] and [STD3]) and internationalized host name parts. An IDNA-aware application that is resolving a non-internationalized host name part MUST NOT do any preparation or conversion to ACE on any non-internationalized name part.

2.1.3 Resolvers and DNS servers

An operating system might have a set of libraries for converting host names to nameprepped ACE. The input to such a library might be in one or more charsets that are used in applications (UTF-8 and UTF-16 are likely candidates for almost any operating system, and script-specific charsets are likely for localized operating systems). The output would be either the unchanged name part (if the input already conforms to [STD13] and [STD3]), or the nameprepped, ACE-encoded name part.

DNS servers MUST use the ACE format for internationalized host name parts.

If a signalling system which makes negotiation possible between old and new DNS clients and servers is standardized in the future, the encoding of the query in the DNS protocol itself can be changed from ACE to something else, such as UTF-8. The question whether or not this should be used is, however, a separate problem and is not discussed in this memo.

2.1.4 Avoiding exposing users to the raw ACE encoding

All applications that might show the user a host name that was received from a gethostbyaddr or other such lookup SHOULD update as soon as possible in order to prevent users from seeing the ACE. However, this is not considered a big problem because so few applications show this type of resolution to users.

If an application decodes an ACE name but cannot show all of the characters in the decoded name, such as if the name contains characters that the output system cannot display, the application SHOULD show the name in ACE format instead of displaying the name with the replacement character (U+FFFD). This is to make it easier for the user to transfer the name correctly to other programs. Programs that by default show the ACE form when they cannot show all the characters in a name part SHOULD also have a mechanism to show the name with as many characters as possible and replacement characters in the positions where characters cannot be displayed. In many cases, the application doesn't know exactly what the underlying rendering engine can or cannot display.

In addition to the condition above, if an application decodes an ACE name but finds that the decoded name was not properly prepared according to [NAMEPREP] (for example, if it has illegal characters in it), the application SHOULD show the name in ACE format and SHOULD NOT display the name in its decoded form. This is to avoid security issues described in [NAMEPREP].

2.1.5 Automatic detection of ACE

An application which receives a host name SHOULD verify whether or not the host name is in ACE. This is possible by verifying the prefix in each of the labels, and seeing whether or not the label is in ACE. This MUST be done regardless of whether or not the communication channel used (such as keyboard input, cut and paste, application protocol, application payload, and so on) is encoding with ACE.

The reason for this requirement is that many applications are not ACE-aware. Applications that are not ACE-aware will send host names in ACE but mark the charset as being US-ASCII or some other charset which has the characters that are valid in [STD13] as a subset.

2.1.6 Bidirectional text

In IDNA, text storage and display follows the rules in the Unicode standard [Unicode3.1]. In particular, all Unicode text is stored in logical order; the Unicode standard has an extensive discussion of how to deal with reorder glyphs for display when dealing with bidirectional text such as Arabic or Hebrew. See [UAX9] for more information.

3. Name Server Considerations

It is imperative that there be only one encoding for a particular host name. ACE is an encoding for host name parts that use characters outside those allowed for host names [STD13]. Thus, a primary master name server MUST NOT contain an ACE-encoded name that decodes to a host name that is allowed in [STD13] and [STD3].

Name servers MUST NOT have any records with host names that contain internationalized name parts unless those name parts have be prepared according to [NAMEPREP]. If names that are not legal in [NAMEPREP] are passed to an application, it will result in an error being passed to the application with no error being reported to the name server. Further, no application will ever ask for a name that is not legal in [NAMEPREP] because requests always go through [NAMEPREP] before getting to the DNS. Note that [NAMEPREP] describes how to handle versioning of unallocated codepoints.

The host name data in zone files (as specified by section 5 of RFC 1035) MUST be both nameprepped and ACE encoded.

4. Root Server Considerations

Because there are no changes to the DNS protocols, adopting this protocol has no effect on the DNS root servers.

5. Security Considerations

Much of the security of the Internet relies on the DNS. Thus, any change to the characteristics of the DNS can change the security of much of the Internet.

This memo describes an algorithm which encodes characters that are not valid according to STD3 and STD13 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.

When detecting an ACE-encoded host name, and decoding the ACE, care must be taken that the resulting value(s) are valid characters which can be handled by the application. This is described in more detail in section 2.1.4.

Host names are used by users to connect to Internet servers. The security of the Internet would be compromised if a user entering a single internationalized name could be connected to different servers based on different interpretations of the internationalized host name.

Because this document normatively refers to [NAMEPREP], it includes the security considerations from that document as well.

6. References

[NAMEPREP] Paul Hoffman & Marc Blanchet, "Preparation of Internationalized Host Names", draft-ietf-idn-nameprep.

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

[STD3] Bob Braden, "Requirements for Internet Hosts -- Communication Layers" (RFC 1122) and "Requirements for Internet Hosts -- Application and Support" (RFC 1123), STD 3, October 1989.

[STD13] Paul Mockapetris, "Domain names - concepts and facilities" (RFC 1034) and "Domain names - implementation and specification" (RFC 1035, STD 13, November 1987.

[UAX9] Unicode Standard Annex #9, The Bidirectional Algorithm. http://www.unicode.org/unicode/reports/tr9/

[Unicode3.1] The Unicode Standard, Version 3.1.0: The Unicode Consortium. The Unicode Standard, Version 3.0. Reading, MA, Addison-Wesley Developers Press, 2000. ISBN 0-201-61633-5, as amended by: Unicode Standard Annex #27: Unicode 3.1 <http://www.unicode.org/unicode/reports/tr27/tr27-4.html>.

B. Changes from the -02 draft

Editorial changes throughout

2.1.1: Major changes to the second paragraph. Added major text to fourth paragraph.

2.1.4: Added to the end of the second paragraph. Added the third paragraph.

2.1.6: Complete change.

6: Added [Unicode3.1] and [UAX9].

C. Authors' Addresses

Patrik Faltstrom
Cisco Systems
Arstaangsvagen 31 J
S-117 43 Stockholm Sweden
paf@cisco.com

Paul Hoffman
Internet Mail Consortium and VPN Consortium
127 Segre Place
Santa Cruz, CA 95060 USA
phoffman@imc.org