Internet Draft
draft-ietf-idn-dunce-00.txt
April 16, 2001
Expires in six months (October 2001)

John C Klensin
AT&T Labs

DUNCE: A proposal for a Definitely Unencumbered New Compatible [ACE] Encoding

Status of this memo

Abstract

This document describes a transformation method for representing non-ASCII characters in host name parts in a fashion that is completely compatible with the current DNS. It is a potential candidate for an ASCII-Compatible Encoding (ACE) for internationalized host names, as described in the comparison document from the IETF IDN Working Group. This method is based exclusively on long-established mechanisms for denoting the positions of characters in tables, but included variations for compressing that information, also based on long-established mechanisms.

1. Introduction

1.1 Context

There is a strong world-wide desire to use characters other than plain ASCII in host names. Host names have become the equivalent of business or product names for many services on the Internet, so there is a need to make them usable by people whose native scripts are not representable by ASCII. The requirements for

internationalizing host names are described in the IDN WG's requirements document, [IDNReq].

The IDN WG's comparison document [IDNComp] describes three potential main architectures for IDN: arch-1 (just send binary), arch-2 (send binary or ACE), and arch-3 (just send ACE). DUNCE is an ACE that can be used with protocols that match arch-2 or arch-3. It is known as "dumb ACE" because it does not attempt any particular optimization of string patterns, relying instead on either names extended to longer length using DNS extension mechanisms [EDNS] or compression if length optimization is desired (without optimization, the maximum effective length of a DUNCE-encoded name would be about 14 characters). DUNCE specifies an ACE format as specified in ace-1 in [IDNComp]. Further, it specifies an identifying mechanism for ace-2 in [IDNComp], namely ace-2.1.1 (add hopefully-unique legal tag to the beginning of the name part).

In formal terms, DUNCE describes a mechanism for specifying character positions in the ISO/IEC 10646 [ISO10646] coded character set (whose assignment of characters is synchronized with Unicode [Unicode3]) and the rules for using that scheme in the DNS. Since it is a simple method of designating those characters, it probably does not meet the definition of a "charset" as defined in [IDNReq].

The DUNCE protocol has the following features:

- There is exactly one way to convert internationalized host parts to and from DUNCE parts. Host name part uniqueness is preserved.

- Host parts that have no international characters are not changed.

- Names using DUNCE have lengths exactly proportionate to the number of characters (from IS 10646) in the names themselves plus the introducer tag. I.e., DUNCE is not dependent on the code positions in the tables, the relationships of characters in the name, or other coding factors.

- This specification utilizes the well-known Base64 encoding [MIME] or the obvious Base 32 variation [RACE] as a means of shortening the coded strings to permit longer names.

It is important to note that the following sections contain many normative statements with "MUST" and "MUST NOT". Any implementation that does not follow these statements exactly is likely to cause damage to the Internet by creating non-unique representations of host names.

1.2 Author's Disclaimer

This document was written for the convenience of the IDN WG, in case (or for the next time) someone suggests that there are no plausible mechanisms for encoding internationalized names into the DNS which

are unencumbered by any intellectual property rights claims, at least any plausible one.

The author continues to believe that no DNS-based approach is going to solve the "IDN" problem as it is perceived by users and company/enterprise domain name registrants and continues to hold the strong hypothesis that, if non-DNS solutions are needed, it is probably not desirable to further complicate the DNS and risk unknown problems and incompatibilities [DNSROLE].

1.3 Terminology

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

Hexadecimal values are shown preceded with an "0x". For example, "0xa1b5" indicates two octets, 0xa1 followed by 0xb5. Binary values are shown preceded with an "0b". For example, a nine-bit value might be shown as "0b101101111".

Examples in this document use the notation from the Unicode Standard [Unicode3] as well as the ISO 10646 names. For example, the letter "a" may be represented as either "U+0061" or "LATIN SMALL LETTER A".

DUNCE converts strings with internationalized characters into strings of US-ASCII that are acceptable as host name parts in current DNS host naming usage. The former are called "pre-converted" and the latter are called "post-converted".

The protocol actually contains three variations (three dunces ?):

DUNCE1 Direct encoding, with the result that the maximum length of names will be about 14 characters.

DUNCE2 Encoding using Base64 (or Base32, see section 3), with a longer maximum name length

DUNCE3 Compression using the <<TBD>> method, with a maximum name length that will typically be longer than DUNCE2.

DUNCE1 will, in practice, probably be usable only in conjunction with extended-length DNS labels.

1.4 IDN summary

Using the terminology in [IDNComp], DUNCE specifies an ACE format as specified in ace-1. Further, it specifies an identifying mechanism for ace-2, namely ace-2.1.1 (add hopefully-unique legal tag to the beginning of the name part).

The length characteristics of DUNCEn are discussed above. Except

where compression is used, the number of characters in a name that can be encoded in a DNS label will be invariant with the positions or scripts from which those characters are derived.

2. Host Part Transformation

According to [STD13], host parts must be case-insensitive, start and end with a letter or digit, and contain only letters, digits, and the hyphen character ("-"). This, of course, excludes any internationalized characters, as well as many other characters in the ASCII character repertoire. Further, domain name parts must be **63 octets or shorter in length.**

2.1 Name tagging

All post-converted name parts that contain internationalized characters begin with the string "bl--". (Of course, because host name parts are case-insensitive, this might also be represented as "Bl--" or "bL--" or "BL--".) The string "bl--" was chosen because it represents the first two characters of the English expletive "bleech", which is an editorial observation on the context in which this specification is being written. The string "bl--" will change to other strings with more appropriate properties in future versions of this draft.

Note that a zone administrator might still choose to use "bl--" at the beginning of a host name part even if that part does not contain internationalized characters. Zone administrators SHOULD NOT create host part names that begin with "bl--" unless those names are post-converted names. Creating host part names that begin with "bl--" but that are not post-converted names may cause two distinct problems. Some display systems, after converting the post-converted name part back to an internationalized name part, might display the name parts in a possibly-confusing fashion to users. More seriously, some resolvers, after converting the post-converted name part back to an internationalized name part, might reject the host name if it contains illegal characters.

2.2 Converting an internationalized name to an ACE name part

To convert a string of internationalized characters into an ACE name part, the following steps MUST be preformed in the exact order of the subsections given here.

If a name part consists exclusively of characters that conform to the host name requirements in [STD13], the name MUST NOT be converted to DUNCE. That is, a name part that can be represented without DUNCE MUST NOT be encoded using DUNCE. This absolute requirement prevents there from being two different encodings for a single DNS host name.

If any checking for prohibited name parts (such as ones that are prohibited characters, case-folding, or canonicalization) is to be done, it MUST be done before doing the conversion to an ACE name part.

Characters outside the first plane of characters (those with codepoints above U+FFFF) MUST be represented using surrogates, as described in the UTF-16 description in ISO 10646.

The input name string consists of characters from the ISO 10646 character set in big-endian UTF-16 encoding. This is the pre-converted string.

2.2.1 Check the input string for disallowed names

If the input string consists only of characters that conform to the host name requirements in [STD13], the conversion MUST stop with an error.

2.2.2 Represent each character by its column and row position.

2.2.2.1 For DUNCE1...

Mechanisms for describing code point positions by a printable (and ASCII) column and row position date to very early code point tables and were believed to have been used for BCD and Baudot. The earliest references readily available to the author are those for [EBCDIC] and [ASCII], but those cited are not even the original references for those coding and techniques. Note that these techniques are all in the public literature and have been widely practiced. In these notations, column and row positions are typically separated by slashes or commas, but, as long as a number system is used that permits representation in a fixed number of digits, simple catenation is well-known as well. For example, in ASCII, the coding position for the character "M" is variously represented as 4/13, 4,13 or 0413 (decimal notation) or 4/D, 4,D, or 4D (hexadecimal notation).

For DUNCE1, each code point is represented by its column and row position, each expressed as two hexidecimal digits. E.g., Latin character upper case M would become 040D.

Catenate all such four-digit strings in the same order that the characters appeared in the original label.

2.2.2.2 For DUNCE2...

Code each character into the 16-bit representation of that character in IS 10646 BMP (plane 0), taking the column positions before the row ones. Catenate the strings thus formed in the same order that the characters appeared in the original label.

When the complete string is formed, convert it to Base64 (or Base32,

see section 3) encoding, as specified in [MIME].

2.2.2.3 For DUNCE3...

Code each character into a 16-bit representation and then catenate the strings, as for DUNCE2. Then compress the resulting bit string, using <<TBD>>. Some compression mechanisms produce, or can easily be altered to produce, case-insensitive ASCII encodings. The results of such compressions can be used directly. Others produce a binary result which will then need to be converted using Base64 (or Base32, see section 3).

2.2.3 Prepend "bl--" to the encoded string and finish

Prepend the characters "bl--" to the encoded string. This is the host name part that can be used in DNS resolution.

2.3 Converting a host name part to an internationalized name

The input string for conversion is a valid host name part. Note that if any checking for prohibited name parts (such as prohibited characters, case-folding, or canonicalization is to be done, it MUST be done after doing the conversion from an ACE name part.

If a decoded name part consists exclusively of characters that conform to the host name requirements in [STD13], the conversion from DUNCE MUST fail. Because a name part that can be represented without DUNCE MUST NOT be encoded using DUNCE, the decoding process MUST check for name parts that consists exclusively of characters that conform to the host name requirements in [STD13] and, if such a name part is found, MUST beconsidered an error (and possibly a security violation).

2.3.1 Strip the "bl--"

The input string MUST begin with the characters "bl--". If it does not, the conversion MUST stop with an error. Otherwise, remove the characters "bl--" from the input string. The result of this step is the stripped string.

2.3.2 Decode the stripped string

2.3.2.1 For DUNCE1...

Divide the stripped string into chunks of four hexidecimal digits each. If the string is not an exact multiple of four characters in length, or if any character is outside the range 0...9...F, report an error. Use the hex-encoded row and column positions to reconstruct the original characters, then catenate them to form the resulting string.

2.3.2.2 For DUNCE2...

Apply a Base64 decoding to reconstruct the original binary string and use that string to restore the original character codes.

2.3.2.3 For DUNCE3...

Apply a Base64 decoding if needed, uncompress the string to restore the original binary, then use that string as above.

2.3.3 Check the internationalized string for disallowed names

If the internationalized string consists only of characters that conform to the host name requirements in [STD13], the conversion MUST stop with an error.

3. Using Base64 (or Base32)

The RACE [RACE] specification and its variations use a Base32 encoding to avoid difficulties with case-insensitivity of the coded names. Since DNS implementations are required to preserve the case of names that are deposited, the author naively believes that it ought to be possible to use the more efficient Base64 encoding for DUNCE. If he is wrong, which is probable, DUNCE2 and, if needed, DUNCE3 can be easily altered to use Base32. Note that DUNCE1 and compression mechanisms that automatically produce case-insensitive ASCII encodings do not depend on the use of Base64 (or Base32) encodings.

4. 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. Thus, DUNCE makes no changes to the DNS itself.

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.

DUNCE is designed so that every internationalized host name part can be represented as one and only one DNS-compatible string. If there is any way to follow the steps in this document and get two or more different results, it is a severe and fatal error in the protocol.

5. References

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

[BASE64] N. Freed & N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC **2045. November 1996.**

[DNSROLE] J Klensin, "Role of the Domain Name System", Work in progress, draft-klensin-dns-role. (Current version is -00, November 2000.)

[EBCDIC] TBS - original S/360 _Principles of Operation_ manual.
[ENDS] Paul Vixie, "Extension Mechanisms for DNS (EDNS0)", RFC 2671. August 1999.

[IDNComp] Paul Hoffman, "Comparison of Internationalized Domain Name Proposals", draft-ietf-idn-compare.

[IDNReq] Zita Wenzel and James Seng, "Requirements of Internationalized Domain Names", draft-ietf-idn-requirements. (Current version is -04, October 2000.)

[ISO10646] ISO/IEC 10646-1:1993. International Standard -- Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane. Five amendments and a technical corrigendum have been published up to now. UTF-16 is described in Annex Q, published as Amendment 1. 17 other amendments are currently at various stages of standardization. [[[THIS REFERENCE NEEDS TO BE UPDATED AFTER DETERMINING ACCEPTABLE WORDING]]]

[RACE] Paul Hoffman, "RACE: Row-based ASCII Compatible Encoding for IDN", Work in Progress, November 2000, draft-ietf-idn-race-03.txt.

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

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

[Unicode3] The Unicode Consortium, "The Unicode Standard -- Version 3.0", ISBN 0-201-61633-5. Described at <http://www.unicode.org/unicode/standard/versions/Unicode3.0.html>.

5. Acknowledgements

This document is shamelessly copied and extracted from Paul Hoffman's RACE encoding document [RACE], but is intended to provide a reference point for a completely unencumbered and unencumberable encoding. The acknowledgements in the RACE document apply here as well and will be incorporated if the document is published as an RFC. Harald Alvestrand suggested a name for the protocol after the author made a rude suggestion about another name. However, neither Paul Hoffman nor anyone else besides the author bears the blame for

the stupid techniques described herein.

6. IANA Considerations

This document does not require IANA action, registration, or considerations.

7. Author Contact Information

John C Klensin
AT&T Labs
99 Bedford St, 4th floor
Boston, MA 02111
+1 617 574 3076
klensin@att.com

Expires October 2001