Internet Draft
<draft-duerst-dns-i18n-02.txt>
Expires in six months

M. Duerst
Keio University
July 1998

Internationalization of Domain Names

Status of this Memo

Abstract

Internet domain names are currently limited to a very restricted character set. This document proposes the introduction of a new "zero-level" domain (ZLD) to allow the use of arbitrary characters from the Universal Character Set (ISO 10646/Unicode) in domain names. The proposal is fully backwards compatible and does not need any changes to DNS. Version 02 is reissued without changes just to keep this draft available.

Table of contents

0. Change History

0.8 Changes Made from Version 01 to Version 02

No significant changes; reissued to make it available officially.
Changed author's address.

Changes deferred to future versions (if ever):
- Decide on ZLD name (.i or .i18n.int or something else)
- Decide on casing solution
- Decide on exact syntax
- Proposals for experimental setup

0.9 Changes Made from Version 00 to Version 01

- Minor rewrites and clarifications
- Added the following references: [RFC1730], [Kle96], [ISO3166], [iNORM]
- Slightly expanded discussion about casing
- Added some variant proposals for syntax
- Added some explanations about different kinds of name parallelism
- Added some explanation about independent addition of internationalized names in subdomains without bothering higher-level domains
- Added some explanations about tools needed for support, and the MX/CNAME problem
- Change to RFC1123 (numbers allowed at beginning of labels)

1. Introduction

1.1 Motivation

The lower layers of the Internet do not discriminate any language or script. On the application level, however, the historical dominance of the US and the ASCII character set [ASCII] as a lowest common denominator have led to limitations. The process of removing these limitations is called internationalization (abbreviated i18n). One example of the abovementioned limitations are domain names [RFC1034, RFC1035], where only the letters of the basic Latin alphabet (case-insensitive), the decimal digits, and the hyphen are allowed.

While such restrictions are convenient if a domain name is intended to be used by arbitrary people around the globe, there may be very good reasons for using aliases that are more easy to remember or type in a local context. This is similar to traditional mail addresses, where both local scripts and conventions and the Latin script can be used.

There are many good reasons for domain name i18n, and some arguments that are brought forward against such an extension. This document, however, does not discuss the pros and cons of domain name i18n. It proposes and discusses a solution and therefore eliminates one of the

most often heard arguments agains, namely "it cannot be done".

The solution proposed in this document consists of the introduction of a new "zero-level" domain building the root of a new domain branch, and an encoding of the Universal Character Set (UCS) [ISO10646] into the limited character set of domain names.

1.2 Notational Conventions

In the domain name examples in this document, characters of the basic Latin alphabet (expressible in ASCII) are denoted with lower case letters. Upper case letters are used to represent characters outside ASCII, such as accented characters of the Latin alphabet, characters of other alphabets and syllabaries, ideographic characters, and various signs.

2. The Hidden Zero Level Domain

The domain name system uses the domain "in-addr.arpa" to convert internet addresses back to domain names. One way to view this is to say that in-addr.arpa forms the root of a separate hierarchy. This hierarchy has been made part of the main domain name hierarchy just for implementation convenience. While syntactically, in-addr.arpa is a second level domain (SLD), functionally it is a zero level domain (ZLD) in the same way as "." is a ZLD. A similar example of a ZLD is the domain tpc.int, which provides a hierarchy of the global phone numbering system [RFC1530] for services such as paging and printing to fax machines.

For domain name i18n to work inside the tight restrictions of domain name syntax, one has to define an encoding that maps strings of UCS characters to strings of characters allowable in domain names, and a means to distinguish domain names that are the result of such an encoding from ordinary domain names.

This document proposes to create a new ZLD to distinguish encoded i18n domain names from traditional domain names. This domain would be hidden from the user in the same way as a user does not see in-addr.arpa. This domain could be called "i18n.arpa" (although the use of arpa in this context is definitely not appropriate), simply "i18n", or even just "i". Below, we are using "i" for shortness, while we leave the decision on the actual name to further= discussion.

3. Encoding International Characters

3.1 Encoding Requirements

Until quite recently, the thought of going beyond ASCII for something such as domain names failed because of the lack of a single encompassing character set for the scripts and languages of the world. Tagging techniques such as those used in MIME headers [RFC1522] would be much too clumsy for domain names.

The definition of ISO 10646 [ISO10646], codepoint by codepoint identical with Unicode [Unicode], provides a single Universal Character Set (UCS). A recent report [RFCIAB] clearly recommends to base the i18n of the Internet on these standards.

An encoding for i18n domain names therefore has to take the characters of ISO 10646/Unicode as a starting point. The full four-byte (31 bit) form of UCS, called UCS4, should be used. A limitation to the two-byte form (UCS2), which allows only for the encoding of the Base Multilingual Plane, is too restricting.

For the mapping between UCS4 and the strongly limited character set of domain names, the following constraints have to be considered:

- The structure of domain names, and therefore the "dot", have to be conserved. Encoding is done for individual labels.

- Individual labels in domain names allow the basic Latin alphabet (monocase, 26 letters), decimal digits, and the "-" inside the label. The capacity per octet is therefore limited to somewhat above 5 bits.

- There is no need nor possibility to preserve any characters.

- Frequent characters (i.e. ASCII, alphabetic, UCS2, in that order) should be encoded relatively compactly. A variable-length encoding (similar to UTF-8) seems desirable.

3.2 Encoding Definition

Several encodings for UCS, so called UCS Transform Formats, exist

already, namely UTF-8 [RFC2044], UTF-7 [RFC1642], and UTF-16 [Unicode]. Unfortunately, none of them is suitable for our purposes. We therefore use the following encoding:

- To accommodate the slanted probability distribution of characters in UCS4, a variable-length encoding is used.

- Each target letter encodes 5 bits of information. Four bits of information encode character data, the fifth bit is used to indicate continuation of the variable-length encoding.

- Continuation is indicated by distinguishing the initial letter from the subsequent letter.

- Leading four-bit groups of binary value 0000 of UCS4 characters are discarded, except for the last TWO groups (i.e. the last octet). This means that ASCII and Latin-1 characters need two target letters, the main alphabets up to and including Tibetan need three target letters, the rest of the characters in the BMP need four target letters, all except the last (private) plane in the UTF-16/Surrogates area [Unicode] need five target letters, and so on.

- The letters representing the various bit groups in the various positions are chosen according to the following table:

Nibble Value		Initial	Subsequent
Hex	Binary		
0	0000	G	0
1	0001	H	1
2	0010	I	2
3	0011	J	3
4	0100	K	4
5	0101	L	5
6	0110	M	6
7	0111	N	7
8	1000	O	8
9	1001	P	9
A	1010	Q	A
B	1011	R	B
C	1100	S	C
D	1101	T	D
E	1110	U	E
F	1111	V	F

[Should we try to eliminate "I" and "O" from initial? "I" might be

eliminated because then an algorithm can more easily detect ".i". "O" could lead to some confusion with "0". What other protocols are there that might be able to use a similar solution, but that might have other restrictions for the initial letters? Proposal to run initial range from H to X. Extracting the initial bits then becomes ^ 'H'. Proposal to have a special convention for all-ASCII labels (start label with one of the letters not used above).]

Please note that this solution has the following interesting properties:

- For subsequent positions, there is an equivalence between the hexadecimal value of the character code and the target letter used. This assures easy conversion and checking.

- The absence of digits from the "initial" column, and the fact that the hyphen is not used, assures that the resulting string conforms to domain name syntax.

- Raw sorting of encoded and unencoded domain names is equivalent.

- The boundaries of characters can always be detected easily. (While this is important for representations that are used internally for text editing, it is actually not very important here, because tools for editing can be assumed to use a more straightforward representation internally.)

- Unless control characters are allowed, the target string will never actually contain a G.

3.3 Encoding Example

As an example, the current domain

    is.s.u-tokyo.ac.jp

with the components standing for information science, science, the University of Tokyo, academic, and Japan, might in future be represented by

    JOUHOU.RI.TOUDAI.GAKU.NIHON

(a transliteration of the kanji that might probably be chosen to represent the same domain). Writing each character in U+HHHH notation as in [Unicode], this results in the following (given for reference

only, not the actual encoding or something being typed in by the user):

U+60c5U+5831.U+7406.U+6771U+5927.U+5b66.U+65e5U+672c

The software handling internationalized domain names will translate this, according to the above specifications, before submitting it to the DNS resolver, to:

M0C5L831.N406.M771L927.LB66.M5E5M72C.i

3.4 Length Considerations

DNS allows for a maximum of 63 positions in each part, and for 255 positions for the overall domain name including dots. This allows up to 15 ideographs, or up to 21 letters e.g. from the Hebrew or Arabic alphabet, in a label. While this does not allow for the same margin as in the case of ASCII domain names, it should still be quite sufficient. [Problems could only surface for languages that use very long words or terms and don't know any kind of abbreviations or similar shortening devices. Do these exist? Islandic expert asserted Islandic is not a problem.] DNS contains a compression scheme that avoids sending the same trailing portion of a domain name twice in the same transmission. Long domain names are therefore not that much of a concern.

4. Usage Considerations

4.1 General Usage

To implement this proposal, neither DNS servers nor resolvers need changes. These programs will only deal with the encoded form of the domain name with the .i suffix. Software that wants to offer an internationalized user interface (for example a web browser) is responsible for the necessary conversions. It will analyze the domain name, call the resolver directly if the domain name conforms to the domain name syntax restrictions, and otherwise encode the name according to the specifications of Section 3.2 and append the .i suffix before calling the resolver. New implementations of resolvers will of course offer a companion function to gethostbyname accepting a ISO10646/Unicode string as input.

For domain name administrators, them main tool that will be needed is a program to compile files configuring zones from an UTF-8 notation (or any other suitable encoding) to the encoding described in Section 3.3. Utility tools will include a corresponding decompiler, checkers for various kinds of internationalization-related errors, and tools for managing syntactic parallelism (see Section 4.3).

4.2 Usage Restrictions

While this proposal in theory allows to have control characters such as BEL or NUL or symbols such as arrows and smilies in domain names, such characters should clearly be excluded from domain names. Whether this has to be explicitly specified or whether the difficulty to type these characters on any keyboard of the world will limit their use has to be discussed. One approach is to start with a very restricted subset and gradually relax it; the other is to allow almost anything and to rely on common sense. Anyway, such specifications should go into a separate document to allow easy updates.

A related point is the question of equivalence. For historical reasons, ISO 10646/Unicode contain considerable number of compatibility characters and allow more than one representation for characters with diacritics. To guarantee smooth interoperability in these and related cases, additional restrictions or the definition of some form of normalization seem necessary. However, this is a general problem affecting all areas where ISO 10646/Unicode is used in identifiers, and should therefore be addressed in a generic way. See [iNORM] for an initial proposal.

Equally related is the problem of case equivalence. Users can very well distinguish between upper case and lower case. Also, casing in an i18n context is not as straightforward as for ASCII, so that case equivalence is best avoided. Problems therefore result not from the fact that case is distinguished for i18n domain names, but from the fact that existing domain names do not distinguish case. Where it is impossible to distinguish between next.com and NeXT.com, the same two subdomains would easily be distinguishable if subordinate to a i18n domain. There are several possible solutions. One is to try to gradually migrate from a case-insensitive solution to a case-sensitive solution even for ASCII. Another is to allow case-sensitivity only beyond ASCII. Another is to restrict anything beyond ASCII to lowercase only (lowercase distinguishes better than uppercase, and is also generally used for ASCII domain names).

A problem that also has to be discussed and solved is bidirectionality. Arabic and Hebrew characters are written right-to-left, and the

mixture with other characters results in a divergence between logical and graphical sequence. See [HTML-I18N] for more explanations. The proposal of [Yer96] for dealing with bidirectionality in URLs could probably be applied to domain names. Anyway, there should be a general solution for identifiers, not a DNS-specific solution.

4.3 Domain Name Creation

The ".i" ZLD should be created as such to allow the internationalization of domain names. Rules for creating subdomains inside ".i" should follow the established rules for the creation of functionally equivalent domains in the existing domain hierarchy, and should evolve in parallel.

For the actual domain hierarchy, the amount of parallelism between the current ASCII-oriented hierarchy and some internationalized hierarchy depends on various factors. In some cases, two fully parallel hierarchies may emerge. In other cases, if more than one script or language is used locally, more than two parallel hierarchies may emerge. Some nodes, e.g. in intranets, may only appear in an i18n hierarchy, whereas others may only appear in the current hierarchy. In some cases, the pecularities of scripts, languages, cultures, and the local marketplace may lead to completely different hierarchies.

Also, one has to be aware that there may be several kinds of parallelisms. The first one is called syntactic parallelism. If there is a domain XXXX.yy.zz and a domain vvvv.yy.zz, then the domain yy.zz will have to exist both in the traditional DNS hierarchy as well as within the hierarchy starting at the .i ZLD, with appropriate encoding.

The second type of parallelism is called transcription parallelism. It results by transcribing or transliterating relations between ASCII domain names and domain names in other scripts.

The third type of parallelism is called semantic parallelism. It results from translating elements of a domain name from one language to another, possibly also changing the script or set of used characters.

On the host level, parallelism means that there are two names for the same host. Conventions should exist to decide whether the parallel names should have separate IP addresses or not (A record or CNAME record). With separate IP addresses, address to name lookup is easy, otherwise it needs special precautions to be able to find all names corresponding to a given host address. Another detail entering this

consideration is that MX records only work for hostnames/domains, not for CNAME aliases. This at least has the consequence that alias resolution for internationalized mail addresses has to occur before MX record lookup.

When discussing and applying the rules for creating domain names, some peculiarities of i18n domain names should be carefully considered:

- Depending on the script, reasonable lengths for domain name parts may differ greatly. For ideographic scripts, a part may often be only a one-letter code. Established rules for lengths may need adaptation. For example, a rule for country TLDs could read: one ideographic character or two other characters.

- If the number of generic TLDs (.com, .edu, .org, .net) is kept low, then it may be feasible to restrict i18n TLDs to country TLDs.

- There are no ISO 3166 [ISO3166] two-letter codes in scripts other than Latin. I18n domain names for countries will have to be designed from scratch.

- The names of some countries or regions may pose greater political problems when expressed in the native script than when expressed in 2-letter ISO 3166 codes.

- I18n country domain names should in principle only be created in those scripts that are used locally. There is probably little use in creating an Arabic domain name for China, for example.

- In those cases where domain names are open to a wide range of applicants, a special procedure for accepting applications should be used so that a reasonable-quality fit between ASCII domain names and i18n domain names results where desired. This would probably be done by establishing a period of about a month for applications inside a i18n domain newly created as a parallel for an existing domain, and resolving the detected conflicts. For syntactically parallel domain names, the owners should always be the same. Administration may be split in some cases to account for the necessary linguistic knowledge. For domain names with transcription parallelism and semantic parallelism, the question of owner identity should depend on the real-life situation (trademarks,...).

- It will be desirable to have internationalized subdomains in non-internationalized TLDs. As an example, many companies in France may want to register an accented version of their company name,

while remaining under the .fr TLD. For this, .fr would have to be reregistered as .M6N2.i. Accented and other internationalized subdomains would go below .M6N2.i, whereas unaccented ones would go below .fr in its plain form.

- To generalize the above case, one may need to create a requirement that any domain name registry would have to register and manage syntactically parallel domain names below the .i ZLD upon request to allow registration of i18n domain names in arbitrary subdomains. An alternative to this is to organize domain name search so that e.g. in a search for XXXXXX.fr, if M6N2.i is not found in .i, the name server for .fr is queried for XXXXXX.M6N2.i (with XXXXXX appropriately encoded). This convention would allow lower-level domains to introduce internationalized subdomains without depending on higher-level domains.

4.4 Usage in URLs

According to current definitions, URLs encode sequences of octets into a sequence of characters from a character set that is almost as limited as the character set of domain names [RFC1738]. This is clearly not satisfying for i18n.

Internationalizing URLs, i.e. assigning character semantics to the encoded octets, can either be done separately for each part and/or scheme, or in an uniform way. Doing it separately has the serious disadvantage that software providing user interfaces for URLs in general would have to know about all the different i18n solutions of the different parts and schemes. Many of these solutions may not even be known yet.

It is therefore definitely more advantageous to decide on a single and consistent solution for URL internationalization. The most valuable candidate [Yer96], for many reasons, is UTF-8 [RFC2044], an ASCII-compatible encoding of UCS4.

Therefore, an URL containing the domain name of the example of Section 3.3 should not be written as:

ftp://M0C5L831.N406.M771L927.LB66.M5E5M72C.i

(although this will also work) but rather

ftp://%e6%83%85%e5%a0%b1.%e7%90%86.%e6%9d%b1%e5%a4%a7.
%e5%ad%a6.%e6%97%a5%e6%9c%ac

In this canonical form, the trailing .i is absent, and the octets can be reconstructed from the %HH-encoding and interpreted as UTF-8 by generic URL software. The software part dealing with domain names will carry out the conversion to the .i form.

5. Alternate Proposals

5.1 The Dillon Proposal

The proposal of Michael Dillon [Dillon96] is also based on encoding Unicode into the limited character set of domain names. Distinction is done for each part, using the hyphen in initial position. Because this does not fully conform to the syntax of existing domain names, it is questionable whether it is backwards-compatible. On the other hand, this has the advantage that local i18n domain names can be installed easily without cooperation by the manager of the superdomain.

A variable-length scheme with base 36 is used that can encode up to 1610 characters, absolutely insufficient for Chinese or Japanese. Characters assumed not to be used in i18n domain names are excluded, i.e. only one case is allowed for basic Latin characters. This means that large tables have to be worked out carefully to convert between ISO 10646/Unicode and the actual number that is encoded with base=36.

5.2 Using a Separate Lookup Service

Instead of using a special encoding and burdening DNS with i18n, one could build and use a separate lookup service for i18n domain names. Instead of converting to UCS4 and encoding according to Section 3.2, and then calling the DNS resolver, a program would contact this new service when seeing a domain name with characters outside the allowed range.

Such solutions have various problems. There are many directory services and proposals for how to use them in a way similar to DNS. For an overview and a specific proposal, see [Kle96]. However, while there are many proposals, a real service containing the necessary data and providing the wide installed base and distributed updating is in DNS does not exist.

Most directory service proposals also do not offer uniqueness. Defining unique names again for a separate service will duplicate much of the work done for DNS. If uniqueness is not guaranteed, the

user is bundened with additional selection steps.

Using a separate lookup service for the internationalization of domain names also results in more complex implementations than the proposal made in this draft. Contrary to what some people might expect, the use of a separate lookup service also does not solve a capacity problem with DNS, because there is no such problem, nor will one be created with the introduction of i18n domain names.

6. Generic Considerations

6.1 Security Considerations

This proposal is believed not to raise any other security considerations than the current use of the domain name system.

6.2 Internationalization Considerations

This proposal addresses internationalization as such. The main additional consideration with respect to internationalization may be the indication of language. However, for concise identifiers such as domain names, language tagging would be too much of a burden and would create complex dependencies with semantics.

> NOTE -- This section is introduced based on a recommendation in [RFCIAB]. A similar section addressing internationalization should be included in all application level internet drafts and RFCs.

Acknowledgements

I am grateful in particular to the following persons for their advice or criticism: Bert Bos, Lori Brownell, Michael Dillon, Donald E. Eastlake 3rd, David Goldsmith, Larry Masinter, Ryan Moats, Keith Moore, Thorvardur Kari Olafson, Erik van der Poel, Jurgen Schwertl, Paul A. Vixie, Francois Yergeau, and others.

Bibliography

[ASCII] Coded Character Set -- 7-Bit American Standard Code for Information Interchange, ANSI X3.4-1986.

[Dillon96] M. Dillon, "Multilingual Domain Names", Memra Software Inc., November 1996 (circulated Dec. 6, 1996 on iahc-discuss@iahc.org).

[HTML-I18N] F. Yergeau, G. Nicol, G. Adams, and M. Duerst, "Internationalization of the Hypertext Markup Language", Work in progress (draft-ietf-html-i18n-05.txt), August 1996.

[iNORM] M. Duerst, "Normalization of Internationalized Identifiers", draft-duerst-i18n-norm-00.txt, July 1997.

[ISO3166] ISO 3166, "Code for the representation of names of countries", ISO 3166:1993.

[ISO10646] ISO/IEC 10646-1:1993. International standard -- Information technology -- Universal multiple-octet coded character Set (UCS) -- Part 1: Architecture and basic multilingual plane.

[Kle96] J. Klensin and T. Wolf, Jr., "Domain Names and Company Name Retrieval", Work in progress (draft-klensin-tld-whois-01.txt), November 1996.

[RFC1034] P. Mockapetris, "Domain Names - Concepts and Facilities", ISI, Nov. 1987.

[RFC1035] P. Mockapetris, "Domain Names - Implementation and Specification", ISI, Nov. 1987.

[RFC1522] K. Moore, "MIME (Multipurpose Internet Mail Extensions) Part Two: Message Header Extensions for Non-ASCII Text", University of Tennessee, September 1993.

[RFC1642] D. Goldsmith, M. Davis, "UTF-7: A Mail-safe Transformation Format of Unicode", Taligent Inc., July 1994.

[RFC1730] C. Malamud and M. Rose, "Principles of Operation for the TPC.INT Subdomain: General Principles and Policy", Internet Multicasting Service, October 1993.

[RFC1738] T. Berners-Lee, L. Masinter, and M. McCahill, "Uniform Resource Locators (URL)", CERN, Dec. 1994.

[RFC2044] F. Yergeau, "UTF-8, A Transformation Format of Unicode and ISO 10646", Alis Technologies, October 1996.

[RFCIAB] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R. Atkinson, M. Crispin, P. Svanberg, "Report from the IAB Character Set Workshop", October 1996 (currently available as draft-weider-iab-char-wrkshop-00.txt).

[Unicode] The Unicode Consortium, "The Unicode Standard, Version 2.0", Addison-Wesley, Reading, MA, 1996.

[Yer96] F. Yergeau, "Internationalization of URLs", Alis Technologies,
=
<http://www.alis.com:8085/~yergeau/url-00.html>.

Author's Address

Martin J. Duerst
World Wide Web Consortium
Keio Research Institute at SFC
Keio University
5322 Endo
Fujisawa
252-8520 Japan

Tel: +81 466 49 11 70
E-mail: mduerst@w3.org

NOTE -- Please write the author's name with u-Umlaut wherever possible, e.g. in HTML as Dürst.