Internet Draft
draft-ietf-idn-compare-01.txt
July 11, 2000
Expires in six months

Paul Hoffman
IMC & VPNC

Comparison of Internationalized Domain Name Proposals

Status of this memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC 2026.

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 http://www.ietf.org/shadow.html.

Abstract

The IDN Working Group is working on proposals for internationalized domain names that might become a standard in the IETF. Before a single full proposal can be made, competing proposals must be compared on a wide range of requirements and desired features. This document compares the many parts of a comprehensive protocol that have been proposed. It is the companion document to "Requirements of Internationalized Domain Names" [IDN-REQ], which lays out the requirements for the internationalized domain name protocol.

1. Introduction

As the IDN Working Group has discussed the requirements for IDN, suggestions have been made for various candidate protocols that might meet the requirements. These proposals have been somewhat helpful in bringing up real-world needs for the requirements.

It became clear no single proposal had wide agreement from the working group. In fact, the authors of various proposals found themselves taking some features from other proposals as they revised their drafts. At the same time, working group participants were making suggestions for incremental changes that might affect more than one proposal.

Because of this mixing and matching, it was decided that this IDN comparisons document should compare features that might end up in the final protocol, not full protocol suggestions themselves. The features that had been discussed in the working group were divided by function, and appear in this document in separate sections. For each function, there are multiple suggestions for protocol elements that might meet the requirements that are described in [IDN-REQ].

This document is being discussed on the "idn" mailing list. To join the list, send a message to <majordomo@ops.ietf.org> with the words "subscribe idn" in the body of the message. Archives of the mailing list can also be found at ftp://ops.ietf.org/pub/lists/idn*.

1.1 Format of this document

Each section covers one feature that has been discussed as being part of the final IDN solution. Within each section, alternate proposals are listed with the major perceived pros and cons of the proposal. Also, each proposal is given a label to make discussion of this document (and of the proposals themselves) easier.

References to the numbered requirements in [IDN-REQ] are from version -02 of that document. These numbers are expected to change and the requirements document evolves. In this draft, the requirements are show as "[#n-02]", where "n" is the requirement number from draft -02 of [IDN-REQ]. This document only lists where particular proposals don't meet particular requirmenents from [IDN-REQ], not the ones that they fulfill.

Note that this document is supposed to reflect the discussion of all proposed alternatives, not just the ones that fully match the requirements in [IDN-REQ]. It will serve as a summary of the discussion in the IDN WG for readers in the future who may want to know why certain alternatives were not chosen for the eventual protocol.

The proposal drafts covered in this document are:

[DUERST] Character Normalization in IETF Protocols,
draft-duerst-i18n-norm-03

[IDNE] Internationalized domain names using EDNS (IDNE),
draft-ietf-idn-idne-01

[KWAN] Using the UTF-8 Character Set in the Domain Name System,
draft-skwan-utf8-dns-03

[RACE] RACE: Row-based ASCII Compatible Encoding for IDN,
draft-ietf-idn-race-00

[SENG] UTF-5, a transformation format of Unicode and ISO 10646,
draft-jseng-utf5-01

[UDNS] Using the Universal Character Set in the Domain Name System (UDNS), draft-ietf-idn-udns-00

2. Architecture

One of the biggest questions raised early in the IDN discussion was what the format of internationalized name parts would be on the wire, that is, between the user's computer and the DNS resolvers. It was agreed that the DNS protocols certainly allow non-ASCII octets in domain name parts and resource records, but there was also acknowledgement that many protocols that rely on the DNS could not handle non-ASCII names due to the design of the protocol. Section 3.1 of this document describes the proposed encodings for the non-ASCII name parts.

Because of requirement [#2-02], there were proposals for ASCII-compatible encodings (ACEs) of non-ASCII characters. Different ACEs were proposed (and are discussed in Section 4 of this document), but they all have the same goal: to allow non-ASCII characters to be represented in host names that conform to RFC 1034 [RFC1034].

2.1 arch-1: Just send binary

[KWAN] proposes beginning to send characters outside the range allowed in RFC 1034.

Pro: Easiest to describe. Only changes host name syntax, not any of the related DNS protocols.

Con: Doesn't work with many exiting protocols that relies on DNS. Violates requirement [#9-02].

2.2 arch-2: Send binary or ACE

[UDNS] (and, later, [IDNE]) proposes using both binary and ACE formats on the wire.

Pro: Allows protocols that can handle binary name parts to use them directly, while allowing protocols that cannot use binary name parts to also handle names without conversion. Allows domain names in free text to be displayed in binary even in systems that require ACE-formatted names on the wire.

Con: Requires all software that uses domain names to handle both formats. Requires processing time for conversion of ACE formats into the format must likely used internally to the software.

2.3 arch-3: Just send ACE

[RACE] and [SENG] propose that host naming rules remain the same and that all internationalize domain names be sent in ACE format.

Pro: No changes at all to current DNS protocols.

Con: Requires all software to recognize ACE domain names and convert them to human-readable for display. This is true not only in domain names used on the wire but also domain names used in free text.

3. Names in binary

Both arch-1 and arch-2 include domain name parts that are represented on the wire in a binary format. This section describes some of the features of such names.

3.1 bin-1: Format

There are many different charsets and encodings for the scripts of the world. The WG has discussed which binary encoding should be used on the wire.

3.1.1 bin-1.1: UTF-8

The IETF policy on character sets [RFC2277] states that UTF-8 [RFC2279] is the preferred charset for IETF protocols. UTF-8 encodes all characters in the ISO 10646 repertoire.

Pro: Well-supported in other IETF protocols. Compact for most scripts. Wide implementation in programming languages. US-ASCII characters have the same encoding in UTF-8 as they do in US-ASCII. Because it is based on ISO 10646, expansion of the repertoire comes from respected international standards bodies.

Con: Asian scripts require three octets per character.

3.1.2 bin-1.2: Labelled charsets

Mailing list discussion mentioned using multiple charsets for the binary representation. Each name part would be labelled with the charset used.

Pro: Allows users to specify names in the charsets they are most familiar with.

Con: All resolvers would have to know all charsets. Thus, the number of charsets would probably have to be limited and never expand. Mapping of characters between charsets would have to be exact and not change over time.

3.2 bin-2: Distinguishing binary from current format

Software built for current domain names might give unexpected results when dealing with non-ASCII characters in domain names. For example, it was reported on the mailing list that some software crashes when a

non-ASCII domain name is returned for in-addr.arpa requests. Thus, there may be a need for IDN to prevent software that is not binary-aware from receiving domain names with binary parts. This would only apply to an IDN that used arch-2, not arch-1.

3.2.1 bin-2.1: Don't mark binary

[KWAN] does not specify any way of changing requests to prevent binary name parts from being transmitted.

Pro: No changes to current DNS requests and responses.

Con: Likely to cause disruption in software that is not binary-aware. Likely to cause systems to misread names and possibly (and incorrectly) convert them to ASCII names by stripping off the high bit in octets; this in turn would lead to security problems due to mistaken identities. Returning binary host names to DNS queries is known to break some current software.

3.2.2 bin-2.2: Mark binary with IN bit

[UDNS] describes using a bit from the header of DNS queries to mark the query as possibly containing a binary name part and indicating that the response to the query can contain binary name parts.

Pro: This bit is currently unused and must be set to zero, so current software won't use it accidentally. No changes to any other part of the query or RRs.

Con: It's the last unused bit in the header and DNS folks have indicated that they are very hesitant to give it up.

3.2.3 bin-2.3: Mark binary with new QTYPEs

[UDNS] using new QTYPEs to mark the query as possibly containing a binary name part and indicating that the response to the query can contain binary name parts. QTYPEs are two octets long, and no QTYPEs to date use more than the lower eight bits, so one of the bits from the upper octet could be used to indicate binary names.

Pro: These bits are currently unused and must be set to zero, so current software won't use them accidentally. No changes to any other part of the query or RRs. Uses a bit that isn't as prized as the IN bit.

Con: Software must pay more attention to the QTYPEs than it might have previously.

3.2.4 bin-2.4: Mark binary with EDNS

[IDNE] uses EDNS [RFC2671] to mark the query and response as containing a binary name part.

Pro: There is little use of EDNS at this point, so it is very unlikely to have bad interactions with old software. EDNS allows longer name parts, and allows additional information (such as IDN version number) in each name part.

Con: There is little use of EDNS and this might make implementation harder.

4. Names in ASCII-compatible encoding (ACE)

Both arch-2 and arch-3 include domain name parts that are represented on the wire in an ASCII-compatible encoding (ACE). This section describes some of the features of such names.

4.1 ace-1: Format

A variety of proposals for the format of ACE have been proposed. Each proposal has different features, such as how many characters can be encoded within the 63 octet limit for each name part. The length descriptions in this section assume that there is no distinguishing of ACE from current names; this is not a likely outcome of the WG work.

The descriptions of lengths is based on script block names from [BLOCK-NAMES].

4.1.1 ace-1.1: UTF-5

[SENG] Describes UTF-5, which is a fairly direct encoding of ISO 10646 characters using a system similar to UTF-8. Characters from Basic Latin and Latin-1 Supplement take 2 octets; Latin Extended-A through Tibetan take 3 octets; Myanmar through the end of BMP take 4 octets; non-BMP characters take 5 octets. This means that names using all characters in the Myanmar through the end of BMP are limited to 15 characters.

Pro: Extremely simple.

Con: Poor compression, particularly for Asian scripts.

4.1.2 ace-1.2: RACE

[RACE] describes RACE, which is a two-step algorithm that first compresses the name part, then converts the compressed string into and ACE. Name parts in all scripts other than Han, Yi, Hangul syllables, Ethiopic, and non-BMP take up ceil(1.6*(n+1)) octets; name parts in those scripts and any name that mixes characters from different rows in ISO 10646 take up ceil(3.2*(n+1)) octets. This means that names using Han, Yi, Hangul syllables, or Ethiopic, are limited to 18 characters. (Note: this document used to be called CIDNUC.)

Pro: Best compression for most scripts, and similar compression for the scripts where it is not the best.

Con: More complicated than UTF-5. Not well optimized for names that have mixed scripts, such as non-Latin names that use hyphen or ASCII digits.

4.1.3 ace-1.3: Hex of UTF-8

An early draft described "hex of UTF-8", which is a straight-forward hexadecimal encoding of UTF-8. Characters in Basic Latin (other than non-US-ASCII and hyphen) take 3 octets; Latin Extended-A through Tibetan take 5 octets; Myanmar through end of BMP take 7 octets; non-BMP characters take 9 octets. This means that names using all characters in the Myanmar through the end of BMP are limited to 9 characters.

Pros: Very simple to describe.

Cons: Very poor compression for all scripts.

4.1.4 ace-1.5: SACE

A message on the mailing list pointed to code for SACE, an ASCII encoding that purports to compact to about the same size as UTF-8.

Pros: Similar compression to UTF-8.

Cons: No description of how the algorithm works.

4.2 ace-2: Distinguishing ACE from current names

Software that finds ACE name parts in free text probably should display the name part using the actual characters, not the ACE equivalent. Thus, software must be able to identify which ASCII name parts are ACE and which are non-ACE ASCII parts (such as current names). This would only apply to an IDN proposal that used arch-2, not arch-3.

4.2.1 ace-2.1: Currently legal names

Name parts that are currently legal in RFC 1034 can be tagged to indicate the part is encoded with ACE.

4.2.1.1 ace-2.1.1: Add hopefully-unique legal tag

[RACE] proposes adding a hopefully-unique legal tag to the beginning of the name. The proposal would also work with such a tag at the end of the name part, but it is easier for most people to recognize at the beginning of name parts.

Pros: Easy for software (and humans) to recognize.

Cons: There is no way to prevent people from beginning non-ACE names with the tag. Unless the tag is very unlikely to appear in any name in any human language, non-ACE names that begin with the tag will display oddly or be rejected by some systems.

4.2.1.2 ace-2.1.2: Add a checksum

Off-list discussion has mentioned the possibility of creating a checksum mechanism where the checksum would be added to the beginning (or end) of ACE name parts.

4.2.2 ace-2.2: Currently illegal names

Instead of creating names that are currently legal, another proposal is to create names that use the current ASCII characters but are illegal.

4.2.2.1 ace-2.2.1: Add trailing hyphen

An earlier draft described using a trailing hyphen as a signifier of an ACE name.

Pros: It is surmised that most current software does not reject names that are illegal in this fashion. Thus, there would be little disruption to current systems. This mechanism takes up fewer characters than any proposed in ace-2.1.

Cons: Some current software is will probably break with this mechanism. It goes against some current protocols that match the rules in RFC 1034.

5. Prohibited characters

There was a short but active discussion on the mailing list about which characters from the ISO 10646 character set should never appear in host names. To date, there are no Internet Drafts on the subject. This section summarizes some of the suggestions.

5.1 prohib-1: Identical and near-identical characters

Some characters are visually identical or incredibly similar to other characters, thus making it impossible to accurately enter host names that are seen in print.

5.2 prohib-2: Separators

Horizontal and vertical spacing characters would make it unclear where a host name begins and ends. Also, allowing periods and period-like characters as characters within a name part would also cause similar confusion.

5.3 prohib-3: Non-displaying and non-spacing characters

There are many characters that cannot be seen in the ISO 10646 character set. These include control characters, non-breaking spaces, formatting characters, and tagging characters. These characters would certainly cause confusion if allowed in host names.

5.4 prohib-4: Private use characters

Private use characters from ISO 10646 inherently have no specified visual form (and in fact can be used for non-displaying characters). Thus, there could be no visual interoperability for characters in the private use areas.

5.5 prohib-5: Punctuation

Some punctuation characters are disallowed in URLs because they are used in URL syntax.

5.6 prohib-6: Symbols

Some mailing list discussion stated that characters that do not normally appear in human or company names should not be allowed in host names. This includes symbols and non-name punctuation.

6. Canonicalization

The working group has a spirited discussion on the need for canonicalization. [IDN-REQ] describes many requirements for when and what type of canonicalization might be performed.

6.1 canon-1: Type of canonicalization

The Unicode Consortium's recommendations and definitions of canonicalization [UTR-15] describes many forms of canonicalization that can be performed on character strings. [DUERST] covers much of the same ground but makes more focused requirements for canonicalization on the Internet.

6.1.1 canon-1.1: Normalization Form C

[DUERST] recommends Normalization Form C, as described in [UTR-15], for use on the Internet. This form is a canonical decomposition, followed by canonical composition.

6.1.2 canon-1.2: Normalization Form KC

Discussion on the mailing list recommended Normalization Form KC. This form is a compatibility decomposition, followed by canonical composition. Compatibility decomposition makes characters that have compatibility equivalence the same after decomposing.

6.2 canon-2: Other canonicalization

Host names may have special canonicalization needs that can be added to those given in canon-1.

6.2.1 canon-2.1: Case folding in ASCII

RFC 1034 specifies that there is no difference between host names that have the same letters but the letters have different case. Thus, the name part "example" is considered the same as "Example" and "EXamPLe". Neither uppercase nor lowercase is specified as being canonical.

6.2.2 canon-2.2: Case folding in non-ASCII

Discussion on the mailing list has raised the issue of whether or not non-ASCII Latin characters should have the same case-folding rules as ASCII. Such rules would match the expectations of native speakers of some languages, but would go counter to the expectations of native speakers of other languages.

6.2.3 canon-2.3: Han folding

Discussion on the mailing list has raised the issue of equivalences in some languages use of Han characters. For example, in Chinese, there are many traditional characters that have equivalent simplified characters. Similarly, there are some Han ideographs for which there are multiple representations in ISO 10646. There are no well-established rules for such folding, and some of the proposed folding would be locale-specific.

6.3 canon-3: Location of canonicalization

Canonicalization can be performed in any system in the DNS. Because it is not a trivial operation and can require large tables, the location of where canonicalization is performed is important.

6.3.1 canon-3.1: Canonicalize only in the application

Early canonicalization is a cleaner architecture design. Spending the cycles on the end systems puts less burden on resolvers or servers in the DNS service. When IDN is first adopted, the applications need to be updated anyway to handle the new format for the names. It is easier for people to upgrade their applications than their resolvers if they need a new IDN feature.

6.3.2 canon-3.2: Canonicalize only in the resolver

Updating a single resolver provides new service to large number of applications and (possibly) users. It is easier to find canonicalization bugs in resolvers than in applications because the resolver has predictable programmatic interfaces. IDN will probably be revised often as new characters are added to ISO 10646, so updating smaller number of resolvers is better than revising more applications. When an end user has a problem with resolving an IDN name, it is much easier to test if the problem is in the resolver than in the user's application.

6.3.3 canon-3.3: Canonicalize in the DNS service

Canonicalization should happen as late as possible so that changes in

the canonicalization algorithm don't orphan all applications and resolvers. Some canonicalization discards information and so should be delayed as long as possible. Canonicalization is practically free, computationally (although it involves some large tables). Because adding IDN to the DNS will happen over time, canonicalizing at the server will minimize the number of things that need to be changed, and simplify and centralize the process of change.

7. Transitions

Early in the working group discussion, there was active debate about how the transition from the current host name rules to IDN would be handled. Given requirement [#1-02], this transition is quite important to deciding which proposals might be feasible.

7.1 trans-1: Always do current plus new architecture

In this proposal, IDN will be used at the same time as the current DNS forever. That is, IDN will be in addition to the current DNS.

7.2 trans-2: Transition period

In this proposal, IDN will be used at the same time as the current DNS for a specified period of time, after which only IDN will exist. That is, IDN will replace the current DNS.

8. Root server considerations

DNS root servers receive all requests for top-level domains that are not in the local DNS cache. They are critical to the Internet. Care must be taken to ensure that root servers will not be affected by new mechanisms introduced.

Any IDN proposal that includes a binary encoding will have an impact on the root servers. The binary requests will affect the root servers because the current root server software is designed to handle current host names. Further, the root zone files which contain ccTLDs and gTLDs would have to support binary domain names and possibly binary host names for NS records. Because all the root servers are equivalent, they would have to be synchronized to support the binary domain names at the same time.

Proposals that only use ACE and use tagging with currently-legal names would, by definition, not affect the root servers.

9. Security considerations

All security considerations listed in [IDN-REQ] apply to this document. Further, all security considerations listed in each of the IDN proposals

must be considered when comparing the proposals.

Some proposals described in this document may create new security considerations. However, these considerations will have to be addressed in the eventual protocol document. All the proposals described here are still incomplete and security considerations may be added to them as they are revised. All the proposals listed in this document use the ISO 10646 **character set, so the proposals inherit any security** characteristics of that character set.

Many protocols and applications rely on domain names to identify the parties involved in a network transaction. For example, a user who connects to a web site by entering or selecting a URL expects that their software will select the web site named in the URL. The uniqueness of domain names are crucial to ensure identification of Internet entities.

To make round-trip translation between local charsets and ISO 10646, the ISO 10646 specification has assigned multiple code points to individual glyphs. Moreover, some glyphs might look similar to some users, but look clearly different by other users. This means that it would be simple for an attacker to mimic a domain name by using similar-looking but different glyphs and guessing that some users will not see the difference in their user interface.

Some IDN protocols may have denial of service attacks, such as by using non-identified chars, exception characters, or under-specified behavior in using some special characters.

10. IANA considerations

This document does not create any new IANA registries. However, it is possible that a character property registry may need to be set up when the IDN protocol is created in order to list prohibited characters (section 5) and canonicalization mappings (section 6).

11. Acknowledgements

James Seng and Marc Blanchet gave many helpful suggestions on the pre-release versions of this document.

12. References

[BLOCK-NAMES] Unicode Consortium, <ftp://ftp.unicode.org/Public/UNIDATA/Blocks.txt>.

[DUERST] Character Normalization in IETF Protocols, draft-duerst-i18n-norm-03

[IDN-REQ] Requirements of Internationalized Domain Names,

draft-ietf-idn-requirements-02

[IDNE] Internationalized domain names using EDNS (IDNE), draft-ietf-idn-idne-01

[KWAN] Using the UTF-8 Character Set in the Domain Name System, draft-skwan-utf8-dns-03

[RACE] RACE: Row-based ASCII Compatible Encoding for IDN, draft-ietf-idn-race-00

[RFC2277] IETF Policy on Character Sets and Languages, RFC 2277

[RFC2279] UTF-8, a transformation format of ISO 10646, RFC 2279

[RFC2671] Extension Mechanisms for DNS (EDNS0), RFC 2671

[SENG] UTF-5, a transformation format of Unicode and ISO 10646, draft-jseng-utf5-01

[UDNS] Using the Universal Character Set in the Domain Name System (UDNS), draft-ietf-idn-udns-00

[UTR15] Unicode Normalization Forms, Unicode Technical Report #15

A. Differences Between -00 and -01 Drafts

Throughout: Changed references from [HOFFMAN] to [RACE].

Throughout: Changed references from [OSCARSSON] to [UDNS].

Throughout: Added [IDNE].

Removed section 1.2.

3.2.3: Updated to mention [UDNS].

3.2.4: Updated with [IDNE], changed "EDNS0" to "EDNS", and reworded.

4.1.2: Added Ethiopic to the list of scripts that require two octets per character.

4.1.3: Removed reference to [OSCARSSON] because that is no longer in the [UDNS] draft.

4.2.2.1: Removed reference to [OSCARSSON] because that is no longer in the [UDNS] draft.

6.1.1: Reworded first sentence.

6.3: Added entire section and subsections.

8: Fixed typo in first sentence.

B. Author Contact

Paul Hoffman
IMC & VPNC
127 Segre Place
Santa Cruz, CA 95060
phoffman@imc.org or paul.hoffman@vpnc.org