Network Working Group
Request for Comments: 3467
Category: Informational

J. Klensin

February 2003

# Role of the Domain Name System (DNS)

## Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

## Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

## Abstract

This document reviews the original function and purpose of the domain name system (DNS). It contrasts that history with some of the purposes for which the DNS has recently been applied and some of the newer demands being placed upon it or suggested for it. A framework for an alternative to placing these additional stresses on the DNS is then outlined. This document and that framework are not a proposed solution, only a strong suggestion that the time has come to begin thinking more broadly about the problems we are encountering and possible approaches to solving them.

## Table of Contents

## 1. Introduction and History

The DNS was designed as a replacement for the older "host table" system. Both were intended to provide names for network resources at a more abstract level than network (IP) addresses (see, e.g., [RFC625], [RFC811], [RFC819], [RFC830], [RFC882]). In recent years, the DNS has become a database of convenience for the Internet, with many proposals to add new features. Only some of these proposals have been successful. Often the main (or only) motivation for using the DNS is because it exists and is widely deployed, not because its existing structure, facilities, and content are appropriate for the particular application of data involved. This document reviews the history of the DNS, including examination of some of those newer applications. It then argues that the overloading process is often inappropriate. Instead, it suggests that the DNS should be supplemented by systems better matched to the intended applications and outlines a framework and rationale for one such system.

Several of the comments that follow are somewhat revisionist. Good design and engineering often requires a level of intuition by the designers about things that will be necessary in the future; the reasons for some of these design decisions are not made explicit at the time because no one is able to articulate them. The discussion below reconstructs some of the decisions about the Internet's primary namespace (the "Class=IN" DNS) in the light of subsequent development and experience. In addition, the historical reasons for particular decisions about the Internet were often severely underdocumented contemporaneously and, not surprisingly, different participants have different recollections about what happened and what was considered important. Consequently, the quasi-historical story below is just one story. There may be (indeed, almost certainly are) other stories about how the DNS evolved to its present state, but those variants do not invalidate the inferences and conclusions.

This document presumes a general understanding of the terminology of RFC 1034 [RFC1034] or of any good DNS tutorial (see, e.g., [Albitz]).

## 1.1 Context for DNS Development

During the entire post-startup-period life of the ARPANET and nearly the first decade or so of operation of the Internet, the list of host names and their mapping to and from addresses was maintained in a frequently-updated "host table" [RFC625], [RFC811], [RFC952]. The names themselves were restricted to a subset of ASCII [ASCII] chosen to avoid ambiguities in printed form, to permit interoperation with systems using other character codings (notably EBCDIC), and to avoid the "national use" code positions of ISO 646 [IS646]. These restrictions later became collectively known as the "LDH" rules for "letter-digit-hyphen", the permitted characters. The table was just a list with a common format that was eventually agreed upon; sites were expected to frequently obtain copies of, and install, new versions. The host tables themselves were introduced to:

- Eliminate the requirement for people to remember host numbers (addresses). Despite apparent experience to the contrary in the conventional telephone system, numeric numbering systems, including the numeric host number strategy, did not (and do not) work well for more than a (large) handful of hosts.

- Provide stability when addresses changed. Since addresses -- to some degree in the ARPANET and more importantly in the contemporary Internet -- are a function of network topology and routing, they often had to be changed when connectivity or topology changed. The names could be kept stable even as addresses changed.

- Provide the capability to have multiple addresses associated with a given host to reflect different types of connectivity and topology. Use of names, rather than explicit addresses, avoided the requirement that would otherwise exist for users and other hosts to track these multiple host numbers and addresses and the topological considerations for selecting one over others.

After several years of using the host table approach, the community concluded that model did not scale adequately and that it would not adequately support new service variations. A number of discussions and meetings were held which drew several ideas and incomplete proposals together. The DNS was the result of that effort. It continued to evolve during the design and initial implementation period, with a number of documents recording the changes (see [RFC819], [RFC830], and [RFC1034]).

The goals for the DNS included:

- Preservation of the capabilities of the host table arrangements (especially unique, unambiguous, host names),
- Provision for addition of additional services (e.g., the special record types for electronic mail routing which quickly followed introduction of the DNS), and
- Creation of a robust, hierarchical, distributed, name lookup system to accomplish the other goals.

The DNS design also permitted distribution of name administration, rather than requiring that each host be entered into a single, central, table by a central administration.

## 1.2 Review of the DNS and Its Role as Designed

The DNS was designed to identify network resources. Although there was speculation about including, e.g., personal names and email addresses, it was not designed primarily to identify people, brands, etc. At the same time, the system was designed with the flexibility to accommodate new data types and structures, both through the addition of new record types to the initial "INternet" class, and, potentially, through the introduction of new classes. Since the appropriate identifiers and content of those future extensions could not be anticipated, the design provided that these fields could contain any (binary) information, not just the restricted text forms of the host table.

However, the DNS, as it is actually used, is intimately tied to the applications and application protocols that utilize it, often at a fairly low level.

In particular, despite the ability of the protocols and data structures themselves to accommodate any binary representation, DNS names as used were historically not even unrestricted ASCII, but a very restricted subset of it, a subset that derives from the original host table naming rules. Selection of that subset was driven in part by human factors considerations, including a desire to eliminate possible ambiguities in an international context. Hence character codes that had international variations in interpretation were excluded, the underscore character and case distinctions were eliminated as being confusing (in the underscore’s case, with the hyphen character) when written or read by people, and so on. These considerations appear to be very similar to those that resulted in similarly restricted character sets being used as protocol elements in many ITU and ISO protocols (cf. [X29]).

Another assumption was that there would be a high ratio of physical hosts to second level domains and, more generally, that the system would be deeply hierarchical, with most systems (and names) at the third level or below and a very large percentage of the total names representing physical hosts. There are domains that follow this model: many university and corporate domains use fairly deep hierarchies, as do a few country-oriented top level domains ("ccTLDs"). Historically, the "US." domain has been an excellent example of the deeply hierarchical approach. However, by 1998, comparison of several efforts to survey the DNS showed a count of SOA records that approached (and may have passed) the number of distinct hosts. Looked at differently, we appear to be moving toward a situation in which the number of delegated domains on the Internet is approaching or exceeding the number of hosts, or at least the number of hosts able to provide services to others on the network. This presumably results from synonyms or aliases that map a great many names onto a smaller number of hosts. While experience up to this time has shown that the DNS is robust enough -- given contemporary machines as servers and current bandwidth norms -- to be able to continue to operate reasonably well when those historical assumptions are not met (e.g., with a flat, structure under ".COM" containing well over ten million delegated subdomains [COMSIZE]), it is still useful to remember that the system could have been designed to work optimally with a flat structure (and very large zones) rather than a deeply hierarchical one, and was not.

Similarly, despite some early speculation about entering people's names and email addresses into the DNS directly (e.g., see [RFC1034]), electronic mail addresses in the Internet have preserved the original, pre-DNS, "user (or mailbox) at location" conceptual format rather than a flatter or strictly dot-separated one. Location, in that instance, is a reference to a host. The sole exception, at least in the "IN" class, has been one field of the SOA record.

Both the DNS architecture itself and the two-level (host name and mailbox name) provisions for email and similar functions (e.g., see the finger protocol [FINGER]), also anticipated a relatively high ratio of users to actual hosts. Despite the observation in RFC 1034 that the DNS was expected to grow to be proportional to the number of users (section 2.3), it has never been clear that the DNS was seriously designed for, or could, scale to the order of magnitude of number of users (or, more recently, products or document objects), rather than that of physical hosts.

Just as was the case for the host table before it, the DNS provided critical uniqueness for names, and universal accessibility to them, as part of overall "single internet" and "end to end" models (cf.

[RFC2826]). However, there are many signs that, as new uses evolved and original assumptions were abused (if not violated outright), the system was being stretched to, or beyond, its practical limits.

The original design effort that led to the DNS included examination of the directory technologies available at the time. The design group concluded that the DNS design, with its simplifying assumptions and restricted capabilities, would be feasible to deploy and make adequately robust, which the more comprehensive directory approaches were not. At the same time, some of the participants feared that the limitations might cause future problems; this document essentially takes the position that they were probably correct. On the other hand, directory technology and implementations have evolved significantly in the ensuing years: it may be time to revisit the assumptions, either in the context of the two- (or more) level mechanism contemplated by the rest of this document or, even more radically, as a path toward a DNS replacement.

### 1.3 The Web and User-visible Domain Names

From the standpoint of the integrity of the domain name system -- and scaling of the Internet, including optimal accessibility to content -- the web design decision to use "A record" domain names directly in URLs, rather than some system of indirection, has proven to be a serious mistake in several respects. Convenience of typing, and the desire to make domain names out of easily-remembered product names, has led to a flattening of the DNS, with many people now perceiving that second-level names under COM (or in some countries, second- or third-level names under the relevant ccTLD) are all that is meaningful. This perception has been reinforced by some domain name registrars [REGISTRAR] who have been anxious to "sell" additional names. And, of course, the perception that one needed a second-level (or even top-level) domain per product, rather than having names associated with a (usually organizational) collection of network resources, has led to a rapid acceleration in the number of names being registered. That acceleration has, in turn, clearly benefited registrars charging on a per-name basis, "cybersquatters", and others in the business of "selling" names, but it has not obviously benefited the Internet as a whole.

This emphasis on second-level domain names has also created a problem for the trademark community. Since the Internet is international, and names are being populated in a flat and unqualified space, similarly-named entities are in conflict even if there would ordinarily be no chance of confusing them in the marketplace. The problem appears to be unsolvable except by a choice between draconian measures. These might include significant changes to the legislation and conventions that govern disputes over "names" and "marks". Or

they might result in a situation in which the "rights" to a name are typically not settled using the subtle and traditional product (or industry) type and geopolitical scope rules of the trademark system. Instead they have depended largely on political or economic power, e.g., the organization with the greatest resources to invest in defending (or attacking) names will ultimately win out. The latter raises not only important issues of equity, but also the risk of backlash as the numerous small players are forced to relinquish names they find attractive and to adopt less-desirable naming conventions.

Independent of these sociopolitical problems, content distribution issues have made it clear that it should be possible for an organization to have copies of data it wishes to make available distributed around the network, with a user who asks for the information by name getting the topologically-closest copy. This is not possible with simple, as-designed, use of the DNS: DNS names identify target resources or, in the case of email "MX" records, a preferentially-ordered list of resources "closest" to a target (not to the source/user). Several technologies (and, in some cases, corresponding business models) have arisen to work around these problems, including intercepting and altering DNS requests so as to point to other locations.

Additional implications are still being discovered and evaluated.

Approaches that involve interception of DNS queries and rewriting of DNS names (or otherwise altering the resolution process based on the topological location of the user) seem, however, to risk disrupting end-to-end applications in the general case and raise many of the issues discussed by the IAB in [IAB-OPES]. These problems occur even if the rewriting machinery is accompanied by additional workarounds for particular applications. For example, security associations and applications that need to identify "the same host" often run into problems if DNS names or other references are changed in the network without participation of the applications that are trying to invoke the associated services.

## 1.4 Internet Applications Protocols and Their Evolution

At the applications level, few of the protocols in active, widespread, use on the Internet reflect either contemporary knowledge in computer science or human factors or experience accumulated through deployment and use. Instead, protocols tend to be deployed at a just-past-prototype level, typically including the types of expedient compromises typical with prototypes. If they prove useful, the nature of the network permits very rapid dissemination (i.e., they fill a vacuum, even if a vacuum that no one previously knew existed). But, once the vacuum is filled, the installed base

provides its own inertia: unless the design is so seriously faulty as to prevent effective use (or there is a widely-perceived sense of impending disaster unless the protocol is replaced), future developments must maintain backward compatibility and workarounds for problematic characteristics rather than benefiting from redesign in the light of experience. Applications that are "almost good enough" prevent development and deployment of high-quality replacements.

The DNS is both an illustration of, and an exception to, parts of this pessimistic interpretation. It was a second-generation development, with the host table system being seen as at the end of its useful life. There was a serious attempt made to reflect the computing state of the art at the time. However, deployment was much slower than expected (and very painful for many sites) and some fixed (although relaxed several times) deadlines from a central network administration were necessary for deployment to occur at all. Replacing it now, in order to add functionality, while it continues to perform its core functions at least reasonably well, would presumably be extremely difficult.

There are many, perhaps obvious, examples of this. Despite many known deficiencies and weaknesses of definition, the "finger" and "whois" [WHOIS] protocols have not been replaced (despite many efforts to update or replace the latter [WHOIS-UPDATE]). The Telnet protocol and its many options drove out the SUPDUP [RFC734] one, which was arguably much better designed for a diverse collection of network hosts. A number of efforts to replace the email or file transfer protocols with models which their advocates considered much better have failed. And, more recently and below the applications level, there is some reason to believe that this resistance to change has been one of the factors impeding IPv6 deployment.

## 2. Signs of DNS Overloading

Parts of the historical discussion above identify areas in which the DNS has become overloaded (semantically if not in the mechanical ability to resolve names). Despite this overloading, it appears that DNS performance and reliability are still within an acceptable range: there is little evidence of serious performance degradation. Recent proposals and mechanisms to better respond to overloading and scaling issues have all focused on patching or working around limitations that develop when the DNS is utilized for out-of-design functions, rather than on dramatic rethinking of either DNS design or those uses. The number of these issues that have arisen at much the same time may argue for just that type of rethinking, and not just for adding complexity and attempting to incrementally alter the design (see, for example, the discussion of simplicity in section 2 of [RFC3439]).

For example:

- While technical approaches such as larger and higher-powered servers and more bandwidth, and legal/political mechanisms such as dispute resolution policies, have arguably kept the problems from becoming critical, the DNS has not proven adequately responsive to business and individual needs to describe or identify things (such as product names and names of individuals) other than strict network resources.

- While stacks have been modified to better handle multiple addresses on a physical interface and some protocols have been extended to include DNS names for determining context, the DNS does not deal especially well with many names associated with a given host (e.g., web hosting facilities with multiple domains on a server).

- Efforts to add names deriving from languages or character sets based on other than simple ASCII and English-like names (see below), or even to utilize complex company or product names without the use of hierarchy, have created apparent requirements for names (labels) that are over 63 octets long. This requirement will undoubtedly increase over time; while there are workarounds to accommodate longer names, they impose their own restrictions and cause their own problems.

- Increasing commercialization of the Internet, and visibility of domain names that are assumed to match names of companies or products, has turned the DNS and DNS names into a trademark battleground. The traditional trademark system in (at least) most countries makes careful distinctions about fields of applicability. When the space is flattened, without differentiation by either geography or industry sector, not only are there likely conflicts between "Joe’s Pizza" (of Boston) and "Joe’s Pizza" (of San Francisco) but between both and "Joe’s Auto Repair" (of Los Angeles). All three would like to control "Joes.com" (and would prefer, if it were permitted by DNS naming rules, to also spell it as "Joe’s.com" and have both resolve the same way) and may claim trademark rights to do so, even though conflict or confusion would not occur with traditional trademark principles.

- Many organizations wish to have different web sites under the same URL and domain name. Sometimes this is to create local variations -- the Widget Company might want to present different material to a UK user relative to a US one -- and sometimes it is to provide higher performance by supplying information from the server topologically closest to the user. If the name resolution

mechanism is expected to provide this functionality, there are three possible models (which might be combined):

- supply information about multiple sites (or locations or references). Those sites would, in turn, provide information associated with the name and sufficient site-specific attributes to permit the application to make a sensible choice of destination, or

- accept client-site attributes and utilize them in the search process, or

- return different answers based on the location or identity of the requestor.

While there are some tricks that can provide partial simulations of these types of function, DNS responses cannot be reliably conditioned in this way.

These, and similar, issues of performance or content choices can, of course, be thought of as not involving the DNS at all. For example, the commonly-cited alternate approach of coupling these issues to HTTP content negotiation (cf. [RFC2295]), requires that an HTTP connection first be opened to some "common" or "primary" host so that preferences can be negotiated and then the client redirected or sent alternate data. At least from the standpoint of improving performance by accessing a "closer" location, both initially and thereafter, this approach sacrifices the desired result before the client initiates any action. It could even be argued that some of the characteristics of common content negotiation approaches are workarounds for the non-optimal use of the DNS in web URLs.

o Many existing and proposed systems for "finding things on the Internet" require a true search capability in which near matches can be reported to the user (or to some user agent with an appropriate rule-set) and to which queries may be ambiguous or fuzzy. The DNS, by contrast, can accommodate only one set of (quite rigid) matching rules. Proposals to permit different rules in different localities (e.g., matching rules that are TLD- or zone-specific) help to identify the problem. But they cannot be applied directly to the DNS without either abandoning the desired level of flexibility or isolating different parts of the Internet from each other (or both). Fuzzy or ambiguous searches are desirable for resolution of names that might have spelling variations and for names that can be resolved into different sets of glyphs depending on context. Especially when internationalization is considered, variant name problems go beyond simple differences in representation of a character or

ordering of a string. Instead, avoiding user astonishment and confusion requires consideration of relationships such as languages that can be written with different alphabets, Kanji-Hiragana relationships, Simplified and Traditional Chinese, etc. See [Seng] for a discussion and suggestions for addressing a subset of these issues in the context of characters based on Chinese ones. But that document essentially illustrates the difficulty of providing the type of flexible matching that would be anticipated by users; instead, it tries to protect against the worst types of confusion (and opportunities for fraud).

- The historical DNS, and applications that make assumptions about how it works, impose significant risk (or forces technical kludges and consequent odd restrictions), when one considers adding mechanisms for use with various multi-character-set and multilingual "internationalization" systems. See the IAB's discussion of some of these issues [RFC2825] for more information.

- In order to provide proper functionality to the Internet, the DNS must have a single unique root (the IAB provides more discussion of this issue [RFC2826]). There are many desires for local treatment of names or character sets that cannot be accommodated without either multiple roots (e.g., a separate root for multilingual names, proposed at various times by MINC [MINC] and others), or mechanisms that would have similar effects in terms of Internet fragmentation and isolation.

- For some purposes, it is desirable to be able to search not only an index entry (labels or fully-qualified names in the DNS case), but their values or targets (DNS data). One might, for example, want to locate all of the host (and virtual host) names which cause mail to be directed to a given server via MX records. The DNS does not support this capability (see the discussion in [IQUERY]) and it can be simulated only by extracting all of the relevant records (perhaps by zone transfer if the source permits doing so, but that permission is becoming less frequently available) and then searching a file built from those records.

- Finally, as additional types of personal or identifying information are added to the DNS, issues arise with protection of that information. There are increasing calls to make different information available based on the credentials and authorization of the source of the inquiry. As with information keyed to site locations or proximity (as discussed above), the DNS protocols make providing these differentiated services quite difficult if not impossible.

In each of these cases, it is, or might be, possible to devise ways to trick the DNS system into supporting mechanisms that were not designed into it. Several ingenious solutions have been proposed in many of these areas already, and some have been deployed into the marketplace with some success. But the price of each of these changes is added complexity and, with it, added risk of unexpected and destabilizing problems.

Several of the above problems are addressed well by a good directory system (supported by the LDAP protocol or some protocol more precisely suited to these specific applications) or searching environment (such as common web search engines) although not by the DNS. Given the difficulty of deploying new applications discussed above, an important question is whether the tricks and kludges are bad enough, or will become bad enough as usage grows, that new solutions are needed and can be deployed.

## 3. Searching, Directories, and the DNS

### 3.1 Overview

The constraints of the DNS and the discussion above suggest the introduction of an intermediate protocol mechanism, referred to below as a "search layer" or "searchable system". The terms "directory" and "directory system" are used interchangeably with "searchable system" in this document, although the latter is far more precise. Search layer proposals would use a two (or more) stage lookup, not unlike several of the proposals for internationalized names in the DNS (see section 4), but all operations but the final one would involve searching other systems, rather than looking up identifiers in the DNS itself. As explained below, this would permit relaxation of several constraints, leading to a more capable and comprehensive overall system.

Ultimately, many of the issues with domain names arise as the result of efforts to use the DNS as a directory. While, at the time this document was written, sufficient pressure or demand had not occurred to justify a change, it was already quite clear that, as a directory system, the DNS is a good deal less than ideal. This document suggests that there actually is a requirement for a directory system, and that the right solution to a searchable system requirement is a searchable system, not a series of DNS patches, kludges, or workarounds.

The following points illustrate particular aspects of this conclusion.

- A directory system would not require imposition of particular length limits on names.

- A directory system could permit explicit association of attributes, e.g., language and country, with a name, without having to utilize trick encodings to incorporate that information in DNS labels (or creating artificial hierarchy for doing so).

- There is considerable experience (albeit not much of it very successful) in doing fuzzy and "sonex" (similar-sounding) matching in directory systems. Moreover, it is plausible to think about different matching rules for different areas and sets of names so that these can be adapted to local cultural requirements. Specifically, it might be possible to have a single form of a name in a directory, but to have great flexibility about what queries matched that name (and even have different variations in different areas). Of course, the more flexibility that a system provides, the greater the possibility of real or imagined trademark conflicts. But the opportunity would exist to design a directory structure that dealt with those issues in an intelligent way, while DNS constraints almost certainly make a general and equitable DNS-only solution impossible.

- If a directory system is used to translate to DNS names, and then DNS names are looked up in the normal fashion, it may be possible to relax several of the constraints that have been traditional (and perhaps necessary) with the DNS. For example, reverse-mapping of addresses to directory names may not be a requirement even if mapping of addresses to DNS names continues to be, since the DNS name(s) would (continue to) uniquely identify the host.

- Solutions to multilingual transcription problems that are common in "normal life" (e.g., two-sided business cards to be sure that recipients trying to contact a person can access romanized spellings and numbers if the original language is not comprehensible to them) can be easily handled in a directory system by inserting both sets of entries.

- A directory system could be designed that would return, not a single name, but a set of names paired with network-locational information or other context-establishing attributes. This type of information might be of considerable use in resolving the "nearest (or best) server for a particular named resource"

problems that are a significant concern for organizations hosting web and other sites that are accessed from a wide range of locations and subnets.

- Names bound to countries and languages might help to manage trademark realities, while, as discussed in section 1.3 above, use of the DNS in trademark-significant contexts tends to require worldwide "flattening" of the trademark system.

Many of these issues are a consequence of another property of the DNS: names must be unique across the Internet. The need to have a system of unique identifiers is fairly obvious (see [RFC2826]). However, if that requirement were to be eliminated in a search or directory system that was visible to users instead of the DNS, many difficult problems -- of both an engineering and a policy nature -- would be likely to vanish.

## 3.2 Some Details and Comments

Almost any internationalization proposal for names that are in, or map into, the DNS will require changing DNS resolver API calls ("gethostbyname" or equivalent), or adding some pre-resolution preparation mechanism, in almost all Internet applications -- whether to cause the API to take a different character set (no matter how it is then mapped into the bits used in the DNS or another system), to accept or return more arguments with qualifying or identifying information, or otherwise. Once applications must be opened to make such changes, it is a relatively small matter to switch from calling into the DNS to calling a directory service and then the DNS (in many situations, both actions could be accomplished in a single API call).

A directory approach can be consistent both with "flat" models and multi-attribute ones. The DNS requires strict hierarchies, limiting its ability to differentiate among names by their properties. By contrast, modern directories can utilize independently-searched attributes and other structured schema to provide flexibilities not present in a strictly hierarchical system.

There is a strong historical argument for a single directory structure (implying a need for mechanisms for registration, delegation, etc.). But a single structure is not a strict requirement, especially if in-depth case analysis and design work leads to the conclusion that reverse-mapping to directory names is not a requirement (see section 5). If a single structure is not needed, then, unlike the DNS, there would be no requirement for a global organization to authorize or delegate operation of portions of the structure.

The "no single structure" concept could be taken further by moving away from simple "names" in favor of, e.g., multiattribute, multihierarchical, faceted systems in which most of the facets use restricted vocabularies. (These terms are fairly standard in the information retrieval and classification system literature, see, e.g., [IS5127].) Such systems could be designed to avoid the need for procedures to ensure uniqueness across, or even within, providers and databases of the faceted entities for which the search is to be performed. (See [DNS-Search] for further discussion.)

While the discussion above includes very general comments about attributes, it appears that only a very small number of attributes would be needed. The list would almost certainly include country and language for internationalization purposes. It might require "charset" if we cannot agree on a character set and encoding, although there are strong arguments for simply using ISO 10646 (also known as Unicode or "UCS" (for Universal Character Set) [UNICODE], [IS10646] coding in interchange. Trademark issues might motivate "commercial" and "non-commercial" (or other) attributes if they would be helpful in bypassing trademark problems. And applications to resource location, such as those contemplated for Uniform Resource Identifiers (URIs) [RFC2396, RFC3305] or the Service Location Protocol [RFC2608], might argue for a few other attributes (as outlined above).

## 4. Internationalization

Much of the thinking underlying this document was driven by considerations of internationalizing the DNS or, more specifically, providing access to the functions of the DNS from languages and naming systems that cannot be accurately expressed in the traditional DNS subset of ASCII. Much of the relevant work was done in the IETF's "Internationalized Domain Names" Working Group (IDN-WG), although this document also draws on extensive parallel discussions in other forums. This section contains an evaluation of what was learned as an "internationalized DNS" or "multilingual DNS" was explored and suggests future steps based on that evaluation.

When the IDN-WG was initiated, it was obvious to several of the participants that its first important task was an undocumented one: to increase the understanding of the complexities of the problem sufficiently that naive solutions could be rejected and people could go to work on the harder problems. The IDN-WG clearly accomplished that task. The beliefs that the problems were simple, and in the corresponding simplistic approaches and their promises of quick and painless deployment, effectively disappeared as the WG's efforts matured.

Some of the lessons learned from increased understanding and the dissipation of naive beliefs should be taken as cautions by the wider community: the problems are not simple. Specifically, extracting small elements for solution rather than looking at whole systems, may result in obscuring the problems but not solving any problem that is worth the trouble.

## 4.1 ASCII Isn't Just Because of English

The hostname rules chosen in the mid-70s weren't just "ASCII because English uses ASCII", although that was a starting point. We have discovered that almost every other script (and even ASCII if we permit the rest of the characters specified in the ISO 646 International Reference Version) is more complex than hostname-restricted-ASCII (the "LDH" form, see section 1.1). And ASCII isn't sufficient to completely represent English -- there are several words in the language that are correctly spelled only with characters or diacritical marks that do not appear in ASCII. With a broader selection of scripts, in some examples, case mapping works from one case to the other but is not reversible. In others, there are conventions about alternate ways to represent characters (in the language, not [only] in character coding) that work most of the time, but not always. And there are issues in coding, with Unicode/10646 providing different ways to represent the same character ("character", rather than "glyph", is used deliberately here). And, in still others, there are questions as to whether two glyphs "match", which may be a distance-function question, not one with a binary answer. The IETF approach to these problems is to require pre-matching canonicalization (see the "stringprep" discussion below).

The IETF has resisted the temptations to either try to specify an entirely new coded character set, or to pick and choose Unicode/10646 characters on a per-character basis rather than by using well-defined blocks. While it may appear that a character set designed to meet Internet-specific needs would be very attractive, the IETF has never had the expertise, resources, and representation from critically-important communities to actually take on that job. Perhaps more important, a new effort might have chosen to make some of the many complex tradeoffs differently than the Unicode committee did, producing a code with somewhat different characteristics. But there is no evidence that doing so would produce a code with fewer problems and side-effects. It is much more likely that making tradeoffs differently would simply result in a different set of problems, which would be equally or more difficult.

### 4.2 The "ASCII Encoding" Approaches

While the DNS can handle arbitrary binary strings without known internal problems (see [RFC2181]), some restrictions are imposed by the requirement that text be interpreted in a case-independent way ([RFC1034], [RFC1035]). More important, most internet applications assume the hostname-restricted "LDH" syntax that is specified in the host table RFCs and as "prudent" in RFC 1035. If those assumptions are not met, many conforming implementations of those applications may exhibit behavior that would surprise implementors and users. To avoid these potential problems, IETF internationalization work has focused on "ASCII-Compatible Encodings" (ACE). These encodings preserve the LDH conventions in the DNS itself. Implementations of applications that have not been upgraded utilize the encoded forms, while newer ones can be written to recognize the special codings and map them into non-ASCII characters. These approaches are, however, not problem-free even if human interface issues are ignored. Among other issues, they rely on what is ultimately a heuristic to determine whether a DNS label is to be considered as an internationalized name (i.e., encoded Unicode) or interpreted as an actual LDH name in its own right. And, while all determinations of whether a particular query matches a stored object are traditionally made by DNS servers, the ACE systems, when combined with the complexities of international scripts and names, require that much of the matching work be separated into a separate, client-side, canonicalization or "preparation" process before the DNS matching mechanisms are invoked [STRINGPREP].

### 4.3 "Stringprep" and Its Complexities

As outlined above, the model for avoiding problems associated with putting non-ASCII names in the DNS and elsewhere evolved into the principle that strings are to be placed into the DNS only after being passed through a string preparation function that eliminates or rejects spurious character codes, maps some characters onto others, performs some sequence canonicalization, and generally creates forms that can be accurately compared. The impact of this process on hostname-restricted ASCII (i.e., "LDH") strings is trivial and essentially adds only overhead. For other scripts, the impact is, of necessity, quite significant.

Although the general notion underlying stringprep is simple, the many details are quite subtle and the associated tradeoffs are complex. A design team worked on it for months, with considerable effort placed into clarifying and fine-tuning the protocol and tables. Despite general agreement that the IETF would avoid getting into the business of defining character sets, character codings, and the associated conventions, the group several times considered and rejected special

treatment of code positions to more nearly match the distinctions made by Unicode with user perceptions about similarities and differences between characters. But there were intense temptations (and pressures) to incorporate language-specific or country-specific rules. Those temptations, even when resisted, were indicative of parts of the ongoing controversy or of the basic unsuitability of the DNS for fully internationalized names that are visible, comprehensible, and predictable for end users.

There have also been controversies about how far one should go in these processes of preparation and transformation and, ultimately, about the validity of various analogies. For example, each of the following operations has been claimed to be similar to case-mapping in ASCII:

- stripping of vowels in Arabic or Hebrew
- matching of "look-alike" characters such as upper-case Alpha in Greek and upper-case A in Roman-based alphabets
- matching of Traditional and Simplified Chinese characters that represent the same words,
- matching of Serbo-Croatian words whether written in Roman-derived or Cyrillic characters

A decision to support any of these operations would have implications for other scripts or languages and would increase the overall complexity of the process. For example, unless language-specific information is somehow available, performing matching between Traditional and Simplified Chinese has impacts on Japanese and Korean uses of the same "traditional" characters (e.g., it would not be appropriate to map Kanji into Simplified Chinese).

Even were the IDN-WG's other work to have been abandoned completely or if it were to fail in the marketplace, the stringprep and nameprep work will continue to be extremely useful, both in identifying issues and problem code points and in providing a reasonable set of basic rules. Where problems remain, they are arguably not with nameprep, but with the DNS-imposed requirement that its results, as with all other parts of the matching and comparison process, yield a binary "match or no match" answer, rather than, e.g., a value on a similarity scale that can be evaluated by the user or by user-driven heuristic functions.

### 4.4 The Unicode Stability Problem

ISO 10646 basically defines only code points, and not rules for using or comparing the characters. This is part of a long-standing tradition with the work of what is now ISO/IEC JTC1/SC2: they have performed code point assignments and have typically treated the ways in which characters are used as beyond their scope. Consequently, they have not dealt effectively with the broader range of internationalization issues. By contrast, the Unicode Technical Committee (UTC) has defined, in annexes and technical reports (see, e.g., [UTR15]), some additional rules for canonicalization and comparison. Many of those rules and conventions have been factored into the "stringprep" and "nameprep" work, but it is not straightforward to make or define them in a fashion that is sufficiently precise and permanent to be relied on by the DNS.

Perhaps more important, the discussions leading to nameprep also identified several areas in which the UTC definitions are inadequate, at least without additional information, to make matching precise and unambiguous. In some of these cases, the Unicode Standard permits several alternate approaches, none of which are an exact and obvious match to DNS needs. That has left these sensitive choices up to IETF, which lacks sufficient in-depth expertise, much less any mechanism for deciding to optimize one language at the expense of another.

For example, it is tempting to define some rules on the basis of membership in particular scripts, or for punctuation characters, but there is no precise definition of what characters belong to which script or which ones are, or are not, punctuation. The existence of these areas of vagueness raises two issues: whether trying to do precise matching at the character set level is actually possible (addressed below) and whether driving toward more precision could create issues that cause instability in the implementation and resolution models for the DNS.

The Unicode definition also evolves. Version 3.2 appeared shortly after work on this document was initiated. It added some characters and functionality and included a few minor incompatible code point changes. IETF has secured an agreement about constraints on future changes, but it remains to be seen how that agreement will work out in practice. The prognosis actually appears poor at this stage, since UTC chose to ballot a recent possible change which should have been prohibited by the agreement (the outcome of the ballot is not relevant, only that the ballot was issued rather than having the result be a foregone conclusion). However, some members of the community consider some of the changes between Unicode 3.0 and 3.1 and between 3.1 and 3.2, as well as this recent ballot, to be

evidence of instability and that these instabilities are better handled in a system that can be more flexible about handling of characters, scripts, and ancillary information than the DNS.

In addition, because the systems implications of internationalization are considered out of scope in SC2, ISO/IEC JTC1 has assigned some of those issues to its SC22/WG20 (the Internationalization working group within the subcommittee that deals with programming languages, systems, and environments). WG20 has historically dealt with internationalization issues thoughtfully and in depth, but its status has several times been in doubt in recent years. However, assignment of these matters to WG20 increases the risk of eventual ISO internationalization standards that specify different behavior than the UTC specifications.

### 4.5 Audiences, End Users, and the User Interface Problem

Part of what has "caused" the DNS internationalization problem, as well as the DNS trademark problem and several others, is that we have stopped thinking about "identifiers for objects" -- which normal people are not expected to see -- and started thinking about "names" -- strings that are expected not only to be readable, but to have linguistically-sensible and culturally-dependent meaning to non-specialist users.

Within the IETF, the IDN-WG, and sometimes other groups, avoided addressing the implications of that transition by taking "outside our scope -- someone else’s problem" approaches or by suggesting that people will just become accustomed to whatever conventions are adopted. The realities of user and vendor behavior suggest that these approaches will not serve the Internet community well in the long term:

- If we want to make it a problem in a different part of the user interface structure, we need to figure out where it goes in order to have proof of concept of our solution. Unlike vendors whose sole [business] model is the selling or registering of names, the IETF must produce solutions that actually work, in the applications context as seen by the end user.

- The principle that "they will get used to our conventions and adapt" is fine if we are writing rules for programming languages or an API. But the conventions under discussion are not part of a semi-mathematical system, they are deeply ingrained in culture. No matter how often an English-speaking American is told that the Internet requires that the correct spelling of "colour" be used, he or she isn’t going to be convinced. Getting a French-speaker in Lyon to use exactly the same lexical conventions as a French-

> speaker in Quebec in order to accommodate the decisions of the IETF or of a registrar or registry is just not likely. "Montreal" is either a misspelling or an anglicization of a similar word with an acute accent mark over the "e" (i.e., using the Unicode character U+00E9 or one of its equivalents). But global agreement on a rule that will determine whether the two forms should match -- and that won't astonish end users and speakers of one language or the other -- is as unlikely as agreement on whether "misspelling" or "anglicization" is the greater travesty.

More generally, it is not clear that the outcome of any conceivable nameprep-like process is going to be good enough for practical, user-level, use. In the use of human languages by humans, there are many cases in which things that do not match are nonetheless interpreted as matching. The Norwegian/Danish character that appears in U+00F8 (visually, a lower case 'o' overstruck with a forward slash) and the "o-umlaut" German character that appears in U+00F6 (visually, a lower case 'o' with diaeresis (or umlaut)) are clearly different and no matching program should yield an "equal" comparison. But they are more similar to each other than either of them is to, e.g., "e". Humans are able to mentally make the correction in context, and do so easily, and they can be surprised if computers cannot do so. Worse, there is a Swedish character whose appearance is identical to the German o-umlaut, and which shares code point U+00F6, but that, if the languages are known and the sounds of the letters or meanings of words including the character are considered, actually should match the Norwegian/Danish use of U+00F8.

This text uses examples in Roman scripts because it is being written in English and those examples are relatively easy to render. But one of the important lessons of the discussions about domain name internationalization in recent years is that problems similar to those described above exist in almost every language and script. Each one has its idiosyncrasies, and each set of idiosyncracies is tied to common usage and cultural issues that are very familiar in the relevant group, and often deeply held as cultural values. As long as a schoolchild in the US can get a bad grade on a spelling test for using a perfectly valid British spelling, or one in France or Germany can get a poor grade for leaving off a diacritical mark, there are issues with the relevant language. Similarly, if children in Egypt or Israel are taught that it is acceptable to write a word with or without vowels or stress marks, but that, if those marks are included, they must be the correct ones, or a user in Korea is potentially offended or astonished by out-of-order sequences of Jamo, systems based on character-at-a-time processing and simplistic matching, with no contextual information, are not going to satisfy user needs.

Users are demanding solutions that deal with language and culture. Systems of identifier symbol-strings that serve specialists or computers are, at best, a solution to a rather different (and, at the time this document was written, somewhat ill-defined), problem. The recent efforts have made it ever more clear that, if we ignore the distinction between the user requirements and narrowly-defined identifiers, we are solving an insufficient problem. And, conversely, the approaches that have been proposed to approximate solutions to the user requirement may be far more complex than simple identifiers require.

### 4.6 Business Cards and Other Natural Uses of Natural Languages

Over the last few centuries, local conventions have been established in various parts of the world for dealing with multilingual situations. It may be helpful to examine some of these. For example, if one visits a country where the language is different from ones own, business cards are often printed on two sides, one side in each language. The conventions are not completely consistent and the technique assumes that recipients will be tolerant. Translations of names or places are attempted in some situations and transliterations in others. Since it is widely understood that exact translations or transliterations are often not possible, people typically smile at errors, appreciate the effort, and move on.

The DNS situation differs from these practices in at least two ways. Since a global solution is required, the business card would need a number of sides approximating the number of languages in the world, which is probably impossible without violating laws of physics. More important, the opportunities for tolerance don't exist: the DNS requires a exact match or the lookup fails.

### 4.7 ASCII Encodings and the Roman Keyboard Assumption

Part of the argument for ACE-based solutions is that they provide an escape for multilingual environments when applications have not been upgraded. When an older application encounters an ACE-based name, the assumption is that the (admittedly ugly) ASCII-coded string will be displayed and can be typed in. This argument is reasonable from the standpoint of mixtures of Roman-based alphabets, but may not be relevant if user-level systems and devices are involved that do not support the entry of Roman-based characters or which cannot conveniently render such characters. Such systems are few in the world today, but the number can reasonably be expected to rise as the Internet is increasingly used by populations whose primary concern is with local issues, local information, and local languages. It is,

for example, fairly easy to imagine populations who use Arabic or Thai scripts and who do not have routine access to scripts or input devices based on Roman-derived alphabets.

### 4.8 Intra-DNS Approaches for "Multilingual Names"

It appears, from the cases above and others, that none of the intra-DNS-based solutions for "multilingual names" are workable. They rest on too many assumptions that do not appear to be feasible -- that people will adapt deeply-entrenched language habits to conventions laid down to make the lives of computers easy; that we can make "freeze it now, no need for changes in these areas" decisions about Unicode and nameprep; that ACE will smooth over applications problems, even in environments without the ability to key or render Roman-based glyphs (or where user experience is such that such glyphs cannot easily be distinguished from each other); that the Unicode Consortium will never decide to repair an error in a way that creates a risk of DNS incompatibility; that we can either deploy EDNS [RFC2671] or that long names are not really important; that Japanese and Chinese computer users (and others) will either give up their local or IS 2022-based character coding solutions (for which addition of a large fraction of a million new code points to Unicode is almost certainly a necessary, but probably not sufficient, condition) or build leakproof and completely accurate boundary conversion mechanisms; that out of band or contextual information will always be sufficient for the "map glyph onto script" problem; and so on. In each case, it is likely that about 80% or 90% of cases will work satisfactorily, but it is unlikely that such partial solutions will be good enough. For example, suppose someone can spell her name 90% correctly, or a company name is matched correctly 80% of the time but the other 20% of attempts identify a competitor: are either likely to be considered adequate?

## 5. Search-based Systems: The Key Controversies

For many years, a common response to requirements to locate people or resources on the Internet has been to invoke the term "directory". While an in-depth analysis of the reasons would require a separate document, the history of failure of these invocations has given "directory" efforts a bad reputation. The effort proposed here is different from those predecessors for several reasons, perhaps the most important of which is that it focuses on a fairly-well-understood set of problems and needs, rather than on finding uses for a particular technology.

As suggested in some of the text above, it is an open question as to whether the needs of the community would be best served by a single (even if functionally, and perhaps administratively, distributed)

directory with universal applicability, a single directory that supports locally-tailored search (and, most important, matching) functions, or multiple, locally-determined, directories. Each has its attractions. Any but the first would essentially prevent reverse-mapping (determination of the user-visible name of the host or resource from target information such as an address or DNS name). But reverse mapping has become less useful over the years --at least to users -- as more and more names have been associated with many host addresses and as CIDR [CIDR] has proven problematic for mapping smaller address blocks to meaningful names.

Locally-tailored searches and mappings would permit national variations on interpretation of which strings matched which other ones, an arrangement that is especially important when different localities apply different rules to, e.g., matching of characters with and without diacriticals. But, of course, this implies that a URL may evaluate properly or not depending on either settings on a client machine or the network connectivity of the user. That is not, in general, a desirable situation, since it implies that users could not, in the general case, share URLs (or other host references) and that a particular user might not be able to carry references from one host or location to another.

And, of course, completely separate directories would permit translation and transliteration functions to be embedded in the directory, giving much of the Internet a different appearance depending on which directory was chosen. The attractions of this are obvious, but, unless things were very carefully designed to preserve uniqueness and precise identities at the right points (which may or may not be possible), such a system would have many of the difficulties associated with multiple DNS roots.

Finally, a system of separate directories and databases, if coupled with removal of the DNS-imposed requirement for unique names, would largely eliminate the need for a single worldwide authority to manage the top of the naming hierarchy.

## 6. Security Considerations

The set of proposals implied by this document suggests an interesting set of security issues (i.e., nothing important is ever easy). A directory system used for locating network resources would presumably need to be as carefully protected against unauthorized changes as the DNS itself. There also might be new opportunities for problems in an arrangement involving two or more (sub)layers, especially if such a system were designed without central authority or uniqueness of names. It is uncertain how much greater those risks would be as compared to a DNS lookup sequence that involved looking up one name,

getting back information, and then doing additional lookups potentially in different subtrees. That multistage lookup will often be the case with, e.g., NAPTR records [RFC 2915] unless additional restrictions are imposed. But additional steps, systems, and databases almost certainly involve some additional risks of compromise.

## 7. References

### 7.1 Normative References

None

### 7.2 Explanatory and Informative References

[Albitz] Any of the editions of Albitz, P. and C. Liu, DNS and BIND, O'Reilly and Associates, 1992, 1997, 1998, 2001.

[ASCII] American National Standards Institute (formerly United States of America Standards Institute), X3.4, 1968, "USA Code for Information Interchange". ANSI X3.4-1968 has been replaced by newer versions with slight modifications, but the 1968 version remains definitive for the Internet. Some time after ASCII was first formulated as a standard, ISO adopted international standard 646, which uses ASCII as a base. IS 646 actually contained two code tables: an "International Reference Version" (often referenced as ISO 646-IRV) which was essentially identical to the ASCII of the time, and a "Basic Version" (ISO 646-BV), which designates a number of character positions for national use.

[CIDR] Fuller, V., Li, T., Yu, J. and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, September 1993.

Eidnes, H., de Groot, G. and P. Vixie, "Classless IN-ADDR.ARPA delegation", RFC 2317, March 1998.

[COM-SIZE] Size information supplied by Verisign Global Registry Services (the zone administrator, or "registry operator", for COM, see [REGISTRAR], below) to ICANN, third quarter 2002.

[DNS-Search] Klensin, J., "A Search-based access model for the DNS", Work in Progress.

[FINGER] Zimmerman, D., "The Finger User Information Protocol", RFC 1288, December 1991.

Harrenstien, K., "NAME/FINGER Protocol", RFC 742, December 1977.

[IAB-OPES] Floyd, S. and L. Daigle, "IAB Architectural and Policy Considerations for Open Pluggable Edge Services", RFC 3238, January 2002.

[IQUERY] Lawrence, D., "Obsoleting IQUERY", RFC 3425, November 2002.

[IS646] ISO/IEC 646:1991 Information technology -- ISO 7-bit coded character set for information interchange

[IS10646] ISO/IEC 10646-1:2000 Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane and ISO/IEC 10646-2:2001 Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 2: Supplementary Planes

[MINC] The Multilingual Internet Names Consortium, http://www.minc.org/ has been an early advocate for the importance of expansion of DNS names to accommodate non-ASCII characters. Some of their specific proposals, while helping people to understand the problems better, were not compatible with the design of the DNS.

[NAPTR] Mealling, M. and R. Daniel, "The Naming Authority Pointer (NAPTR) DNS Resource Record", RFC 2915, September 2000.

Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS", RFC 3401, October 2002.

Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Two: The Algorithm", RFC 3402, October 2002.

Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[REGISTRAR] In an early stage of the process that created the Internet Corporation for Assigned Names and Numbers (ICANN), a "Green Paper" was released by the US Government. That paper introduced new terminology and some concepts not needed by traditional DNS operations. The term "registry" was applied to the actual operator and database holder of a domain (typically at the top level, since the Green Paper was little concerned with anything else), while organizations that marketed names and made them available to "registrants" were known as "registrars". In the classic DNS model, the function of "zone administrator" encompassed both registry and registrar roles, although that model did not anticipate a commercial market in names.

[RFC625] Kudlick, M. and E. Feinler, "On-line hostnames service", RFC 625, March 1974.

[RFC734] Crispin, M., "SUPDUP Protocol", RFC 734, October 1977.

[RFC811] Harrenstien, K., White, V. and E. Feinler, "Hostnames Server", RFC 811, March 1982.

[RFC819] Su, Z. and J. Postel, "Domain naming convention for Internet user applications", RFC 819, August 1982.

[RFC830] Su, Z., "Distributed system for Internet name service", RFC 830, October 1982.

[RFC882] Mockapetris, P., "Domain names: Concepts and facilities", RFC 882, November 1983.

[RFC883] Mockapetris, P., "Domain names: Implementation specification", RFC 883, November 1983.

[RFC952] Harrenstien, K, Stahl, M. and E. Feinler, "DoD Internet host table specification", RFC 952, October 1985.

[RFC953] Harrenstien, K., Stahl, M. and E. Feinler, "HOSTNAME SERVER", RFC 953, October 1985.

[RFC1034] Mockapetris, P., "Domain names, Concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC1591] Postel, J., "Domain Name System Structure and Delegation", RFC 1591, March 1994.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2295] Holtman, K. and A. Mutz, "Transparent Content Negotiation in HTTP", RFC 2295, March 1998

[RFC2396] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[RFC2608] Guttman, E., Perkins, C., Veizades, J. and M. Day, "Service Location Protocol, Version 2", RFC 2608, June 1999.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[RFC2825] IAB, Daigle, L., Ed., "A Tangled Web: Issues of I18N, Domain Names, and the Other Internet protocols", RFC 2825, May 2000.

[RFC2826] IAB, "IAB Technical Comment on the Unique DNS Root", RFC 2826, May 2000.

[RFC2972] Popp, N., Mealling, M., Masinter, L. and K. Sollins, "Context and Goals for Common Name Resolution", RFC 2972, October 2000.

[RFC3305] Mealling, M. and R. Denenberg, Eds., "Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations", RFC 3305, August 2002.

[RFC3439] Bush, R. and D. Meyer, "Some Internet Architectural Guidelines and Philosophy", RFC 3439, December 2002.

[Seng] Seng, J., et al., Eds., "Internationalized Domain Names: Registration and Administration Guideline for Chinese, Japanese, and Korean", Work in Progress.

[STRINGPREP] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings (stringprep)", RFC 3454, December 2002.

The particular profile used for placing internationalized strings in the DNS is called "nameprep", described in Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names", Work in Progress.

[TELNET] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, RFC 854, May 1983.

Postel, J. and J. Reynolds, "Telnet Option Specifications", STD 8, RFC 855, May 1983.

[UNICODE] The Unicode Consortium, The Unicode Standard, Version 3.0, Addison-Wesley: Reading, MA, 2000. Update to version 3.1, 2001. Update to version 3.2, 2002.

[UTR15] Davis, M. and M. Duerst, "Unicode Standard Annex #15: Unicode Normalization Forms", Unicode Consortium, March 2002. An integral part of The Unicode Standard, Version 3.1.1. Available at (http://www.unicode.org/reports/tr15/tr15-21.html).

[WHOIS] Harrenstien, K, Stahl, M. and E. Feinler, "NICNAME/WHOIS", RFC 954, October 1985.

[WHOIS-UPDATE] Gargano, J. and K. Weiss, "Whois and Network Information Lookup Service, Whois++", RFC 1834, August 1995.

Weider, C., Fullton, J. and S. Spero, "Architecture of the Whois++ Index Service", RFC 1913, February 1996.

Williamson, S., Kosters, M., Blacka, D., Singh, J. and K. Zeilstra, "Referral Whois (RWhois) Protocol V1.5", RFC 2167, June 1997;

Daigle, L. and P. Faltstrom, "The application/whoispp-query Content-Type", RFC 2957, October 2000.

Daigle, L. and P. Falstrom, "The application/whoispp-response Content-type", RFC 2958, October 2000.

[X29] International Telecommuncations Union, "Recommendation X.29: Procedures for the exchange of control information and user data between a Packet Assembly/Disassembly (PAD) facility and a packet mode DTE or another PAD", December 1997.

## 8. Acknowledgements

Many people have contributed to versions of this document or the thinking that went into it. The author would particularly like to thank Harald Alvestrand, Rob Austein, Bob Braden, Vinton Cerf, Matt Crawford, Leslie Daigle, Patrik Faltstrom, Eric A. Hall, Ted Hardie, Paul Hoffman, Erik Nordmark, and Zita Wenzel for making specific suggestions and/or challenging the assumptions and presentation of earlier versions and suggesting ways to improve them.

## 9. Author's Address

John C. Klensin
1770 Massachusetts Ave, #322
Cambridge, MA 02140

EMail: klensin+srch@jck.com

A mailing list has been initiated for discussion of the topics discussed in this document, and closely-related issues, at ietf-irnss@lists.elistx.com. See http://lists.elistx.com/archives/ for subscription and archival information.

## 10. Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.