Network Working Group
Request for Comments: 4185
Category: Informational

J. Klensin

October 2005

National and Local Characters for DNS Top Level Domain (TLD) Names

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

IESG Note

This RFC is not a candidate for any level of Internet Standard. The IETF disclaims any knowledge of the fitness of this RFC for any purpose and notes that the decision to publish is not based on IETF review apart from IESG review for conflict with IETF work. The RFC Editor has chosen to publish this document at its discretion. See RFC 3932 [RFC3932] for more information.

Abstract

In the context of work on internationalizing the Domain Name System (DNS), there have been extensive discussions about "multilingual" or "internationalized" top level domain names (TLDs), especially for countries whose predominant language is not written in a Roman-based script. This document reviews some of the motivations for such domains, several suggestions that have been made to provide needed functionality, and the constraints that the DNS imposes. It then suggests an alternative, local translation, that may solve a superset of the problem while avoiding protocol changes, serious deployment delays, and other difficulties. The suggestion utilizes a localization technique in applications to permit any TLD to be accessed using the vocabulary and characters of any language. It is not restricted to language- or country-specific "multilingual" TLDs in the language(s) and script(s) of that country.

Table of Contents

1. Introduction

1.1. Terminology

This document assumes the conventional terminology used to discuss the domain name system (DNS) and its hierarchical arrangements. Terms such as "top level domain" (or just "TLD"), "subdomain", "subtree", and "zone file" are used without further explanation. In addition, the term "ccTLD" is used to denote a "country code top level domain" and "gTLD" is used to denote a "generic top level domain" as described in [RFC1591] and in common usage.

1.2. Background on the "Multilingual Name" Problem

People who share a language usually prefer to communicate in it, using whatever characters are normally used to write that language, rather than in some "foreign" one. There have been standards for using mutually-agreed characters and languages in electronic mail message bodies and selected headers since the introduction of MIME in 1992 [MIME] and the Web has permitted multilingual text since its inception, also using MIME. Actual use of non-Roman-character content came even earlier, using private conventions. However, domain names are exposed to users in email addresses and URLs. Corresponding arrangements, typically also exposing domain names, are made for other application protocols. The combination of exposed domain names with internationalization requirements led rapidly to demands to permit domain names in applications that used characters other than those of the very restrictive, ASCII-subset, "hostname" (or "letter-digit-hyphen" ("LDH")) conventions recommended in the DNS specifications [RFC1035]. The effort to do this soon became known as "multilingual domain names". That was actually a misnomer, since the DNS deals only with characters and identifier strings, and not, except by accident or local registration conventions, with what people usually think of as "names". There has also been little interest in what would actually be a "multilingual name", i.e., a name that contains components from more than one language. Instead, interest has focused on the use, in the context of the DNS, of strings that conform to specific individual languages.

1.2.1. Approaches to the Requirement

When the requirement was seen, not as "modifying the DNS", but as "providing users with access to the DNS from a variety of languages and character sets", three sets of proposals emerged in the IETF and elsewhere. They were:

1. Perform processing in client software that recodes a user-visible string into an ASCII-compatible form that can safely be passed through the DNS protocols and stored in the DNS. This is the approach used, for example, in the IETF's "IDNA" protocol [RFC3490].

2. Modify the DNS to be more hospitable to non-ASCII names and strings. There have been a variety of proposals to do this, using several different techniques. Some of these have been implemented on a proprietary basis by various vendors. None of them have gained acceptance in the IETF community, primarily because they would take a long time to deploy, would leave many problems unsolved, and have been shown to cause problems with deployed approaches that had not yet been upgraded.

3. Move the problem out of the DNS entirely, relying instead on a "directory" or "presentation" layer to handle internationalization. The rationale for this approach is discussed in [RFC3467].

This document proposes a fourth approach, applicable to the top level domains (TLDs) only (see Section 1.3.1 for a discussion of the special issues that make TLDs both problematic and a special opportunity). That approach involves having the user interface of applications map non-ASCII names for TLDs to existing TLDs and could be used as an alternate or supplement to the strategies summarized above.

1.2.2. Writing the Name of One's Country in its Own Characters

An early focus of the "multilingual domain name" efforts was expressed in statements such as "users in my country, in which ASCII is rarely used, should be able to write an entire domain name in their own character set". In particular, since all top-level domain names, at present, follow the LDH rules, the modified naming rules discussed in [RFC1123], and the coding conventions specified in [RFC1591], all fully-qualified DNS names were effectively required to contain at least one ASCII label (the TLD name). Some advocates for internationalized names have considered the presence of any ASCII labels inappropriate. One should, instead, be able to write the name of the ccTLD for China in Chinese, the name of the ccTLD for Saudi Arabia in Arabic, the name for Spain in Spanish, and so on.

That much could be accomplished, given updated applications, by using a new TLD name with IDNA encoding. Of course, adding such a TLD would raise new questions: what to do about gTLDs, how to handle countries with several official languages (perhaps even using different scripts), how should name strings be chosen, and whether

there should be an attempt to coordinate the contents of the local-language TLD zone and the traditional ISO 3166-coded one. A few of these issues are addressed below. But, if one examines (or even thinks about) user behavior and preferences, it is almost as important that one be able to write the name of the ccTLD for China in Arabic and that of Saudi Arabia in Chinese: true internationalization implies that, at least to the extent to which ambiguity and conflicts can be avoided, people should be able to use the languages and character sets they prefer. For the same reasons that one would like to have all-Chinese domain names available in China, it is important to have the capability to have an apparent Chinese-language TLD for a domain whose second level and beyond are Chinese characters, even when the TLD itself serves predominantly non-Chinese-speaking registrants and users.

1.2.3. Countries with Multiple Languages and Countries with Multiple Names

From a user interface standpoint, writing ccTLD names in local characters is a problem. As discussed below in Section 1.3.2, the DNS itself does not easily permit a domain to be referred to by more than one name (or spelling or translation of a name). Countries with more than one official language would require that the country name be represented in each of those languages. And, just as it is important that a user in China be able to represent the name of the Chinese ccTLD in Chinese characters, she should be able to access a Chinese-language site in France using Chinese characters. That would require that she be able to write the name of the French ccTLD in Chinese characters rather than in a form based on a Roman character set.

1.2.4. Availability of Non-ASCII Characters in Programs

Over the years, computer users have gotten used to the fact that not every computer has a full set of characters available to every program. An extreme example is an Arabic speaker using a public kiosk computer in an airport in the United States: there is only a small chance that the web browser there will be able to input and render Arabic correctly. This has a direct effect on the multilingual TLD problem in that it is not possible to simply change a name of the ccTLDs in the DNS to be one of a given country’s non-ASCII names without possibly preventing people from entering those names throughout the world.

1.3. Domain Name System Constraints

1.3.1. Administrative Hierarchy

The domain name system is firmly rooted in the idea of an "administrative hierarchy", with the entity responsible for a given node of the hierarchy responsible for policies applicable to its subhierarchies (Cf. [RFC1034], [RFC1035], and [RFC1591]). The model works quite well for the domain and subdomains of a particular enterprise. In an enterprise situation, the hierarchy can be organized to match the organizational structure; there are established ways to set policies; and there are, at least presumably, shared assumptions about overall goals and objectives among all registrants in the domain. It is more problematic when a domain is shared by unrelated entities that lack common policy assumptions because it is difficult to reach agreement on rules that should apply to all of the entities and subdomains of such a domain. In general, the unrelated entities situation always prevails for the labels registered in a TLD (second-level names). Exceptions occur in those TLDs for which the second level is structural (e.g., the .CO, .AC, .GOV conventions in many ccTLDs or in the historical geographical organization of .US [RFC1480]). In those cases, it exists for the labels within that structural level.

TLDs may, but need not, have consistent registration policies for those second (or third) level names. Countries (or ccTLD administrators) have often adopted rules about what entities may register in their ccTLDs, and what forms the names may take. RFC 1591 outlined registration norms for most of the then-extant gTLDs; however, those norms have been largely ignored in recent years. Some recent "sponsored" and purpose-specific domains are based on quite specific rules about appropriate registrations. Homogeneous registration rules for the root are, by contrast, impossible: almost by definition, the subdomains registered in the root (TLDs) are diverse, and no single policy about types and formats of names applying to all root subdomains is feasible.

1.3.2. Aliases

In an environment different from the DNS, a rational way to permit assigning local-language names to a country code (or other) domain would be to set up an alias for the name, or to use some sort of "see instead" reference. But the DNS does not have facilities for either. Instead, it supports a "CNAME" record, whose label can refer only to a particular label and not to a subtree. For example, if A.B.C is a fully-qualified name, then a CNAME reference in B.C from X to A would make X.B.C appear to have the same values as A.B.C. However, a CNAME reference from Y to C in the root would not make A.B.Y referenceable

(or even defined) at all. A second record type, DNAME [RFC2672], can provide an alias for a portion of the tree. But many believe that it is problematic technically. At a minimum, it can cause synchronization issues when references across zones occur, and its use has been discouraged within the IETF, except as a means of enabling a transition from one domain to another. Even if the design of yet another alias-type record type were contemplated, DNS technical constraints of query-response integrity and DNSSec zone signing (cf. [RFC4033], [RFC4034], and [RFC4035]) make it extremely unlikely that one could be defined that would meet the desired requirements for "see instead" or true synonym references.

1.4. Internationalization and Localization

It has often been observed that, while many people talk about "internationalization", they often really mean, and want, "localization". "Internationalization", in this context, suggests making something globally accessible while incorporating a broad-range "universal" character set and conventions appropriate to all languages and cultures. "Localization", by contrast, involves having things work well in a particular locality or for a broad range of localities, although aspects of the style of operation might differ for each locality. Anything that actually involves the DNS must be global, and hence internationalized, since the DNS cannot meaningfully support different responses or query and matching models based, e.g., on the location of the user making a query. While the DNS cannot support localization internally, many of the features discussed earlier in this section are much more easily thought about in local terms -- whether localized to a geographical area, users of a language, or using some other criteria -- than in global ones.

2. Client-Side Solutions

Traditionally, the IETF avoided becoming involved in standardization for actions that take place strictly on individual hosts on the network, instead confining itself to behavior that is observable "on the wire", i.e., in protocols between network hosts. Exceptions to this general principle have been made when different clients were required to utilize data or interpret values in compatible ways to preserve interoperability: the standards for email and web body formats, and IDNA itself, are examples of these exceptions. Regardless of what is required to be standardized, it is almost never required, and often unwise, that a user interface present "on the wire" formats to the user, at least by default (debugging options that show the wire formats are common and often quite useful). However, in most cases when the presentation format and the wire format differ, the client program must take precautions to ensure that the wire format can be reconstructed from user input, or to keep

the wire format, while hidden, bound to the presentation mechanism so that it can be reconstructed. While it is rarely a goal in itself, it is often necessary that the user be at least vaguely aware that the wire ("real") format is different from the presentation one and that the wire format be available for debugging.

In fact, the DNS itself is an excellent example of the difference between the wire format and the user presentation format. Most Internet users do not realize that the wire format for DNS queries and responses does not include the "." character. Instead, each label is represented by a length in bytes of the label, followed by the label itself.

2.1. IDNA and the Client

As mentioned above, IDNA itself is entirely a client-side protocol. It works by performing some mappings and then encoding labels to be placed into the DNS in a special format called "punycode" [RFC3492]. When labels in that format are encountered, they are transformed, by the client, back into internationalized (normally Unicode [ISO10646]) characters. In the context of this document, the important observation about IDNA is that any application program that supports it is already doing considerable transformation work in the client; it is not simply presenting the "on the wire" formats to the user. It is also the case that, if an application implementation makes different mappings than those called for by IDNA, it is likely to be detected only when, and if, users complain about unexpected behavior. As long as the punycode strings sent to it are valid, the server cannot tell what mappings were applied to develop those strings.

2.2. Local Translation Tables for TLD Names

We suggest that, in addition to maintaining the code and tables required to support IDNA, authors of application programs may want to maintain a table that contains a list of TLDs and locally-desirable names for each one. For ccTLDs, these might be the names (or locally-standard abbreviations) by which the relevant countries are known locally (whether in ASCII characters or others). With some care on the part of the application designer (e.g., to ensure that local forms do not conflict with the actual TLD names), a particular TLD name input from the user could be either in local or standard form without special tagging or problems. When DNS names are received by these client programs, the TLD labels would be mapped to local form before IDNA is applied to the rest of the name; when names are received from users, local TLD names would be mapped to the global ones before applying IDNA or being used in other DNS processing.

3. Advantages and Disadvantages of Local Translation

3.1. Every TLD Appears in the Local Language and Character Set

The notion of a top-level domain whose name matches, e.g., the name that is used for a country in that country or the name of a language in that language as, as mentioned above, is immediately appealing. But most of the reasons for it argue equally strongly for other TLDs being accessible from that language. A user in Korea who can access the national ccTLD in the Korean language and character set has every reason to expect that both generic top level domains and domains associated with other countries would be similarly accessible, especially if the second-level domains bear Korean names. A user native to Spain or Portugal, or in Latin America, would presumably have similar expectations, but would expect to use Spanish or Portuguese names, not Korean ones.

That level of local optimization is not realistic -- some would argue not possible -- with the DNS since it would ultimately require that every top level domain be replicated for each of the world's languages. That replication process would involve not just the top level domain itself; in principle, all of its subtrees would need to be completely replicated as well. Perhaps in practice, not all subtrees would require replication, but only those for which a language variation or translation was significant. But, while that restriction would change the scale of the problem, it would not alter its basic nature. The administrative hierarchy characteristics of the DNS (see Section 1.3.1) turn the replication process into an administrative nightmare: every administrator of a second-level domain in the world would be forced to maintain dozens, probably hundreds, of similar zone files for the replicates of the domain. Even if only the zones relevant to a particular country or language were replicated, the administrative and tracking problems to bind these to the appropriate top-level domain and keep all of the replicas synchronized would be extremely difficult at best. And many administrators of third- and fourth-level domains, and beyond, would be faced with similar problems.

By contrast, dealing with the names of TLDs as a localization problem, using local translation, is fairly simple, although it places some burden of understanding on the user (see Section 4). Each function represented by a TLD -- a country, generic registrations, or purpose-specific registrations -- could be represented in the local language and character set as needed. And, for countries with many languages -- or users living, working in, or visiting countries where their language is not dominant -- "local" could be defined in terms of the needs or wishes of each particular user.

An additional benefit is that, if two countries called themselves by the same name in their local languages -- if, e.g., Western Slobbovia and Eastern Slobbovia both called themselves "Slobland" -- local conventions could be followed as long as users understood that only internal forms (in this case, the ISO 3166-based ccTLD name) could be exported outside the country (see Section 3.3).

Note that this proposal is to allow mapping of native-language strings to existing TLDs. It would almost certainly be ill-advised to stretch this idea too far and try to map strings that local users would be unlikely to guess into TLDs. For example, there are probably no languages in which the country known in English as "Finland" is called "FI". Thus, one would not want to create a mapping from two characters that look or sound like a Roman "F" and a Roman "I" to the ccTLD ".fi".

3.2. Unification of Country Code Domains

It follows from some of the comments above that, while there appears to be some immediate appeal from having (at least) two domains for each country, one using the ISO 3166-1 code [ISO3166] and another one using a name based on the national name in the national language, such a situation would create considerable problems for registrants in both domains. For registrants maintaining enterprise or organizational subdomains, ease of administration of a single family of zone files will usually make a registration in a single top-level domain preferable to replicated sets of them, at least as long as their functional requirements (such a local-language access) are met by the unified structure. For those registrants with no interest in any Internet function or protocols other than use of the HTTP/HTTPS-based web, this problem can be dealt with at the applications level by the use of redirects but, in the general case, that is not a feasible solution.

For countries with multiple national languages that are considered equal and legally equivalent, the advantages of a translation-based approach, rather than multiple registrations and replicated trees, would be even more significant. Actually installing and maintaining a separate TLD for each language would be an administrative nightmare, especially if it was intended that the associated zones be kept synchronized. The oft-suggested proposal to adopt an "exactly one extra domain for each country" rule would essentially require some of the multiple-official-language countries to violate their own constitutions. Conversely, having multiple domains for a given country, based on the number of official languages and without any expectation of synchronization, would give some countries an additional allocation of TLDs that others would certainly consider unfair.

Of course, having replicated domains might be popular with some registries and registrars, since replication would almost inevitably increase the total number of domains to be registered. Helping that group of registries and registrars, while hurting Internet users by adding administrative overhead and confusion, is not a goal of this document.

3.3. User Understanding of Local and Global References

While the IDNA tables (actually Nameprep [RFC3491] and Stringprep [RFC3454]) must be identical globally for IDNA to work reliably, the tables for mapping between local names and TLD names could be locally determined, and differ from one locale to another, as long as users understood that international interchange of names required using the standard forms. That understanding puts some additional burden of learning on users, although part of it could be assisted by software (see Section 4).

In any event, at least in the foreseeable future, it is likely that DNS names being passed among users in different countries, or using different languages, will be forced to be in punycode form to guarantee compatibility, since those users would not, in general, have the ability to read each other's scripts or have appropriate input facilities (keyboards, etc.) for then. So the marginal knowledge or effort needed to put TLD names into standard form and transmit them in that way would actually be fairly small.

3.4. Limits on Expansion of the Number of TLDs

The concept of using local translation does have one side effect that some portions of the Internet community might consider undesirable. The size and complexity of translation tables, and maintaining those tables, will be, to a considerable extent, a function of the number of top-level domains of interest, the frequency with which new domains are added, and the number of domains added at a time. A country or other locale that wished to maintain a complete set of translations (i.e., so that every TLD had a representation in the local language) would presumably find setting up a table for the current collection of a few hundred domains to be a task that would take some days. If the number of TLDs were relatively stable, with a relatively small number being added at infrequent intervals, the updates could probably be dealt with on an ad hoc basis. But, if large numbers of domains were added frequently, or if the total number of TLDs became very large, maintaining the table might require dedicated staff if each new TLD is to be accommodated. Worse, updating the tables stored on client machines might require update

and synchronization protocols and all of the complexities that tend to go with them (see [RFC3696] for a discussion of some related issues in applications).

In practice, there will be little requirement to translate every TLD into a local language. There are already existing TLDs for which there is no obvious translations in many languages (most notably, ".arpa") or where the translation will be far from obvious to typical users (for example, ".int" and ".aero"). Of course, these could be translated by function: ".arpa" to the local term for "infrastructure", ".int" with "international" or "international organization", ".aero" with "aeronautical" or "airlines", and so on; but it is not clear whether doing so would have significant value. For almost every language, there are dozens of ccTLDs for which there are no translations of the country names into the local language that would be known by anyone other than geographers. If new TLDs are added, there might not be a strong need (or even capability) to have language-specific equivalents for each.

3.5. Standardization of the Translations

An immediate question when proposals such as this one are considered is whether the names for the various TLDs that do not match the strings that are actually in the DNS should be standardized and, if so, by what mechanism. Standardization would promote communication within a country or among people sharing a language. However, it is likely to be very difficult to reach appropriate international agreements to which wide conformance could be expected. Exceptions might arise within particular countries or language groups but, even then, there might be advantages to users being able to specify additional synonymous names that are easy for them to remember. As with IDNA-based IDNs, users who wish to transmit information about domain names to people whose exact capabilities and software are unknown, and to do so with minimal risk of confusion, will probably confine themselves to the names that actually appear in the DNS, i.e., the "punycode" representations.

In any event, neither standardization nor uniform use of either the system outlined here or of a specific collection of names is required to make the system work for those who would find it useful. Similarly, mechanisms for country-wide coordination, and examination of the appropriateness or inappropriateness of such mechanisms, is beyond the scope of this document.

3.6. Implications for Future New Domain Names

Applications that implement the proposal in this document are likely to make the subsequent creation and acceptance of new IDNA-based TLDs significantly more difficult. If this proposal becomes widely adopted, local language names mapped as it suggests will be generally expected by users of those languages to mean the same as a current TLD. Creating a new, stand-alone IDNA-based TLD will then require more deliberation and care to avoid conflicts and, when executed, will require all the application software that maps the name to the existing TLD to change the mapping tables.

For several reasons, this problem may not be as serious in practice as it might first appear. For ccTLDs allocated according to the ISO 3166-1 list, there will presumably be no problem at all: not only are the 3166-1 alpha-2 codes strictly in ASCII, but general trends, such as those embodied in ICANN's "GAC Recommendations" against using country names or codes for any purpose not associated with those specific countries, make conflicts with internationalized names extremely unlikely. Because the DNS does not currently have a usable aliasing function (see Section 1.3.2), it is likely that new IDNA-based TLDs will be allocated only after there is considerable opportunity for countries and other individual entities to identify any problems they see with proposed new names.

3.7. Mapping for TLDs, Not Domain Names or Keywords

It should be clear to anyone who has read this far that the mapping described in this document is limited to TLDs, not full domain names or keywords. In particular, nothing here should be construed as applying to anything other than TLDs, due at least in part to the limitations described in Section 3.1. Further, this document is only about the domain name system (DNS), not about any keyword system. The interactions between particular keyword systems and the proposals here are left as a (possibly very difficult) exercise for the reader or implementer of such systems. However, for the subset of such systems whose intent is to entirely hide DNS names or URIs from the user, their output would presumably be the LDH names that actually appeared in the DNS, i.e., in punycode form for IDNA names and without any application processing of the type contemplated here.

4. Information Interchange, IDNs, Comparisons, and Translations

This specification is based on a pair of fairly explicit assumptions. The first is that the greatest and most important impact and value of any internationalization or localization technique is to permit users who share a language or culture to communicate with others who also share that language or culture. Communication among users from

different cultures, using different languages or different scripts is inherently more difficult, and still more difficult if they cannot easily identify languages and scripts in common. The reason for those difficulties are age-old issues in language translation and differences among languages and scripts, not problems associated with the DNS or IDNs, however they are represented. That is the second assumption: when communication across language or cultural groups is required, the users who need to do it -- typically a much smaller number than those communicating within the same language and culture -- are going to need to rely on commonly-understood languages and scripts and will need to exert somewhat more care and effort than within their own groups.

As outlined in the sections above, the suggestions made in this document could clearly be turned into major problems by misuse or misunderstanding. For example, if two applications on the same host used different translation tables, a situation could easily result that would be very confusing to the user. However, in some cases, this would be only slightly worse than some of the alternatives. For example, if, on a given system, IDNs are expressed in native script, but ASCII TLD names are used, cutting and pasting from one application to another may not work as expected, unless both applications and the underlying operating system are all Unicode-based and use the same encoding model for Unicode. Some applications writers have already discovered, even without significant use of IDNs, that they need to support separate "copy string" and "copy link location", and the corresponding "paste" operations. Any use of IDNs or Internationalized Resource Identifiers (IRIs, see [RFC3987]) may require similar operations, or extensions to those operations, to force strings into internal ("punycode" or URI) form on the copy operation and to translate them back on paste. Were that done, the appropriate translations could be performed as part of the same process. If this author’s hypothesis is correct -- that these operations are likely to be required on many systems whether this proposal is adopted or not -- then the additional translation operations are likely to be invisible to the user.

In particular, precisely because the translated names proposed here are part of a presentation form, rather than the internal form names, they are inappropriate in a number of circumstances in which a globally-unique, internal-form name is actually required. It would be a poor, indeed dangerous, idea to use these names in security contexts such as names in certificates, access lists, or other contexts in which accurate comparisons are necessary.

A more general issue exists when DNS or IRI references are transferred among users whose systems may be localized for different languages or conventions. In general, a user in one part of the

world will not actually know how another user's systems are set up, precisely what software is being used, etc., nor should users be expected or forced to learn that information. But, if the user transmitting an internationalized reference doesn't know that the receiving system supports the same characters and fonts, and that the receiving user is prepared to deal with them, the prudent user will transmit the internal form of the reference in addition to, or even instead of, the native-character form. And, of course, if the reference is transmitted on paper, on a sign, in some coded character set other than Unicode, or even as an image, rather than as a Unicode string, the importance of supplementing it with the internal form becomes even more important. The addition of a translation requirement for TLD labels makes availability of internal forms in interchange significantly more important, but does not actually change the requirement to do so.

It may be helpful to note that, in a different networking model than that used in the Internet, both this proposal and IDNA itself are essentially "presentation layer" approaches rather than constructions that can be expected to work well in interchange.

5. Internationalization Considerations

This entire specification addresses issues in internationalization and especially the boundaries between internationalization and localization and between network protocols and client/user interface actions.

6. Security Considerations

IDNA provides a client-based mechanism for presenting Unicode names in applications while passing only ASCII-based names on the wire. As such, it constitutes a major step along the path of introducing a client-based presentation layer into the Internet. Client-based presentation layer transformations introduce risks from non-conforming tables that can change meaning without external protection. For example, if a mapping table normally maps A onto C, and that table is altered by an attacker so that A maps onto D instead, much mischief can be committed. On the other hand, these are not the usual sort of network attacks: they may be thought of as falling into the "users can always cause harm to themselves" category. The local translation model outlined here does not significantly increase the risks over those associated with IDNA, but may provide some new avenues for exploiting them.

Both this approach and IDNA rely on having updated programs present information to the user in a very different form than the one in which it is transmitted on the wire. Unless the internal (wire) form

is always used in interchange, or at least made available when DNS names are exchanged, there are possibilities for ambiguity and confusion about references. As with IDNA itself, if only the "wire" form is presented, the user will perceive that nothing of value has been done, i.e., that no internationalization or localization has occurred. So presentation of the "wire" form to eliminate the potential ambiguities is unlikely to be considered an acceptable solution, regardless of its security advantages.

If the translation tables associated with the technique suggested here are obtained from a server, or translations are obtained from a remote machine using some protocol, the mechanisms used should ensure that the values received are authentic, i.e., that neither they, nor the query for them, have been intercepted and tampered with in any way.

7. Acknowledgements

This document was inspired by a number of conversations in ICANN, IETF, MINC, and private contexts about the future evolution and internationalization of top level domains. Unknown to the author, but unsurprisingly (the general concept should be obvious to anyone even slightly skilled in the relevant technologies), the concept has been apparently developed independently in other groups but, as far as this author knows, not written up for general comment. Discussions within, and about, the ICANN IDN Committee were particularly helpful, although several of the participants in that committee may be surprised about where those discussions led. Email correspondence with several people after the first version of this document was posted, notably Richard Hill, Paul Hoffman, Lee XiaoDong, and Soobok Lee, led to considerable clarification in the subsequent versions. The author is particularly grateful to Paul Hoffman for extensive comments and additional text for the third version and to Patrik Faltstrom, Joel Halpern, Sam Hartman, and Russ Housley for suggestions incorporated into the final one.

The first version of this document was posted on October 21, 2002.

8. Informative References

[ISO10646] International Organization for Standardization, "Information Technology - Universal Multiple-octet coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane", ISO Standard 10646-1, May 1993.

[ISO3166] International Organization for Standardization, "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes", ISO Standard 3166-1:1977, 1997.

[MIME] Borenstein, N. and N. Freed, "MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, June 1992.

Updated and replaced by Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC2045, November 1996. Also, Moore, K., "Representation of Non-ASCII Text in Internet Message Headers", RFC 1342, June 1992. Updated and replaced by Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[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.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[RFC1480] Cooper, A. and J. Postel, "The US Domain", RFC 1480, June 1993.

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

[RFC2672] Crawford, M., "Non-Terminal DNS Name Redirection", RFC 2672, August 1999.

[RFC3454] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("stringprep")", RFC 3454, December 2002.

[RFC3467] Klensin, J., "Role of the Domain Name System (DNS)", RFC 3467, February 2003.

[RFC3490] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[RFC3491] Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)", RFC 3491, March 2003.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, March 2003.

[RFC3696] Klensin, J., "Application Techniques for Checking and Transformation of Names", RFC 3696, February 2004.

[RFC3932] Alvestrand, H., "The IESG and RFC Editor Documents: Procedures", BCP 92, RFC 3932, October 2004.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

Author's Address

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

Phone: +1 617 491 5735
EMail: john-ietf@jck.com

Full Copyright Statement

Intellectual Property

Acknowledgement

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