Independent Submission
Request for Comments: 8324
Category: Informational
ISSN: 2070-1721

J. Klensin

February 2018

DNS Privacy, Authorization, Special Uses, Encoding, Characters, Matching, and Root Structure: Time for Another Look?

Abstract

The basic design of the Domain Name System was completed almost 30 years ago. The last half of that period has been characterized by significant changes in requirements and expectations, some of which either require changes to how the DNS is used or can be accommodated only poorly or not at all. This document asks the question of whether it is time to either redesign and replace the DNS to match contemporary requirements and expectations (rather than continuing to try to design and implement incremental patches that are not fully satisfactory) or draw some clear lines about functionality that is not really needed or that should be performed in some other way.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8324.

Table of Contents

1. Introduction

This document explores contemporary expectations of the Internet's domain system (DNS) and compares them to the assumptions and properties of the DNS design, including both those documented in the RFC Series, an important early paper by the principal author of the original RFCs [Mockapetris-1988], and a certain amount of oral tradition. It is primarily intended to ask the question of whether the differences are causing enough stresses on the system, stresses that cannot be resolved satisfactorily by further patching, that the Internet community should be considering designing a new system, one that is better adapted to current needs and expectations, and developing a deployment and transition strategy for it. For those (perhaps the majority of us) for whom actually replacing the DNS is too radical to be realistic, the document may be useful in two other ways. It may provide a foundation for discussing what functions the DNS should not be expected to support and how those functions can be supported in other ways, perhaps via an intermediate system that then calls on the DNS or by using some other type of database technology for some set of functions while leaving the basic DNS functions intact. Or it may provide a basis for "better just get used to that and the way it works" discussions to replace fantasies about what the DNS might do in some alternate reality.

There is a key design or philosophical question associated with the analysis in this document that the document does not address. It is whether changes to perceived requirements to DNS functionality as described here are, in most respects, evolutionary or whether many of them are instances of trying to utilize the DNS for new requirements because it exists and is already deployed independent of whether the DNS is really appropriate or not. The latter might be an instance of a problem often described in the IETF as "when all you have is a hammer, everything looks like a nail".

Other recent work, including a short article by Vint Cerf [Cerf2017], has discussed an overlapping set of considerations from a different perspective, reinforcing the view that it may be time to ask fundamental questions about the evolution and future of the DNS.

While this document does not assume deep technical or operational knowledge of the DNS, it does assume some knowledge and at least general familiarity with the concepts of RFC 1034 [RFC1034] and RFC 1035 [RFC1035] and the terminology discussed in RFC 7719 [RFC7719] and elsewhere. Although some of the comments it contains might be taken as hints or examples of different ways to think about the design issues, it makes no attempt to explore, much less offer a tutorial on, alternate naming systems or database technologies.

It is perhaps worth noting that, while the perspective is different and more than a dozen years have passed, many of the issues discussed in this document were analyzed and described (most of them with more extensive explanations) in a 2005 US National Research Council report [NRC-Signposts].

Readers should note that several references are to obsolete documents. That was done because they are intended to show the documents and dates that introduced particular features or concepts. When current versions are intended, they are referenced.

2. Background and Hypothesis

The Domain Name System (DNS) [RFC1034] was designed starting in the early 1980s [RFC0799] [RFC0881] [RFC0882] [RFC0883] with the main goal of replacing the flat, centrally administered, host table system [RFC0810] [RFC0952] [RFC0953] with a hierarchical, administratively distributed, system. The DNS design included some features that, after initial implementation and deployment, were judged to be unworkable and either replaced (e.g., the mail destination (MD) and mail forwarder (MF) approach [RFC0882] that were replaced by the MX approach [RFC0974]), abandoned (e.g., the mechanism for using email local parts as labels described in RFC 1034, Section 3.3), or deprecated (e.g., the WKS RR TYPE [RFC1123]). Newer ideas and requirements have identified a number of other features, some of which were less developed than others. Of course the original designers could not anticipate everything that has come to be expected of the DNS in the last 30 years.

In recent years, demand for new and extended services and uses of the DNS have, in turn, led to proposals for DNS extensions or changes of various sorts. Some have been adopted, including a model for negotiating extended functionality [RFC2671] (commonly known as EDNS(0)) and to support IPv6 [RFC3596], others were found to be impracticable, and still others continue to be under consideration. Some examples of the latter two categories are discussed below. A few features of the original DNS specification, such as the CLASS property and label types, have also been suggested to be so badly specified that they should be deprecated [Sullivan-Class].

Unlike earlier changes such as the Internationalized Domain Names for Applications (IDNA) mechanisms for better incorporating non-ASCII labels without modifying the DNS structure itself [RFC3490] [RFC5890], some recent proposals require or strongly suggest changes to APIs, formats, or interfaces by programs that need to retrieve information from the DNS or interpret that information. Differences between the DNS architecture and the requirements that imply those proposals suggest that it may be time to stop patching the DNS or

trying to extend it in small increments. Instead, we should be considering moving some current or proposed functionality elsewhere or developing a new system that better meets today's needs and a transition strategy to it.

The next section of this document discusses a number of issues with the current DNS design that could appropriately be addressed by a different and newer design model. In at least some cases, changing the model and protocols could bring significant benefits to the Internet and/or its administration.

This document is not a proposal for a new protocol. It is intended to stimulate thought about how far we want to try to push the existing DNS, to examine whether expectations of it are already exceeding its plausible capabilities, and to start discussion of a redesign or alternatives to one if the time for that decision has come.

3. Warts and Tensions with the Current DNS

As suggested above, there are many signs that the DNS is incapable of meeting contemporary expectations of how it should work and functionality it should support. Some of those expectations are unrealistic under any imaginable circumstances; others are impossible (or merely problematic) in the current DNS structure but could be accommodated in a redesign. These are examples, rather than a comprehensive list, and do not appear in any particular order.

3.1. Multi-type Queries

The DNS does not gracefully support multi-type queries. The current case where this problem rears its head involves attempts at solutions that return both TYPE A (IPv4) and type AAA (IPv6) addresses collectively. The problem was originally seen with "QTYPE=MAILA" [RFC0882] for the original MA and MD RRTYPEs, an experience that strongly suggests that some very careful thinking about cache effects (and possibly additional DNS changes) would be needed. Other solutions might seem equally or more plausible. What they, including "two types of addresses", probably have in common is that they illustrate stresses on the system and that changing the DNS to deal with those stresses is not straightforward or likely to be problem-free.

3.2. Matching Part I: Case Sensitivity in Labels and Other Anomalies

The DNS specifications assume that labels are octet strings and octets with the high bit zero have seven-bit ASCII codes in the remaining bits. They require that, when a domain name used in a query is matched to one stored in the database, those ASCII characters be interpreted in a case-independent way, i.e., upper- and lower-case letters are treated as equivalent (digits and symbols are not affected) [RFC4343]. For non-ASCII octets, i.e., octets in labels with the first bit turned on, there are no assumptions about the character coding used, much less any rules about character case equivalence -- strings must be compared by matching bits in sequence. Even though the current model for handling non-ASCII (i.e., "internationalized") domain name labels (IDNs) [RFC5890] (see Section 3.3 below) encodes information so the DNS is not directly affected, the notion that some characters in labels are handled in a case-insensitive way and that others are case sensitive (or that upper case must be prohibited entirely as IDNA does) has caused a good deal of confusion and resentment. Those concerns and complaints about inconsistent behavior and mishandling (or suboptimal handling) of case relationships for some languages have not been mitigated by repeated explanations that the relationships between "decorated" lower-case characters and their upper-case equivalents are often sensitive to language and locality and therefore not deterministic with information available to DNS servers.

3.3. Matching Part II: Non-ASCII ("Internationalized") Domain Name Labels

Quite independent of the case-sensitivity problem, one of the fundamental properties of Unicode [Unicode] is that some abstract characters can be represented in multiple ways, such as by a single, precomposed, code point or by a base code point followed by one or more code points that specify combining characters. While Unicode Normalization can be used to eliminate many (but not all) of those distinctions for comparison (matching) purposes, it is best applied during matching rather than by changing one string into another. The first version of IDNA ("IDNA2003") made the choice to change strings during processing for either storage or retrieval [RFC3490] [RFC3491]; the second ("IDNA2008") required that all strings be normalized and that upper-case characters are not allowed at all [RFC5891]. Neither is optimal, if only because, independent of where they are changed if they are changed at all, transforming the strings themselves implies that the input string in an application may not be the same as the string used in processing and perhaps later display.

It would almost certainly be preferable, and more consistent with Unicode recommendations, to use normalization (and perhaps other techniques if they are appropriate) at matching time rather than altering the strings at all, even if there were still only a single matching algorithm, i.e., normalization were added to the existing ASCII-only case folding. However, even Unicode's discussion of normalization [Unicode-UAX15] indicates that there are special, language-dependent, cases (the most commonly cited example is the dotless "i" (U+0131)). Not only does the DNS lack any information about languages that could be used in a mapping algorithm, but, as long as there is a requirement that there be only one mapping algorithm for the entire system, that information could not be used even if it were available. One could imagine a successor system that would use information stored at nodes in the hierarchy to specify different matching rules for subsidiary nodes (or equivalent arrangements for non-hierarchical systems). It is not clear whether that would be a good idea, but it certainly is not possible with the DNS as we know it.

3.4. Matching Part III: Label Synonyms, Equivalent Names, and Variants

As the initial phases of work on IDNs started to conclude, it became obvious that the nature and evolution of human language and writing systems required treating some names as "the same as" others. The first important example of this involved the relatively recent effort to simplify the Chinese writing system, thereby creating a distinction between "Simplified" and "Traditional" Chinese even though the meaning of the characters remained the same in almost all cases (in so-called ideographic character sets, characters have meaning rather than exclusively representing sounds). A joint effort among the relevant Country Code Top-Level Domain (ccTLD) registries and some other interested parties produced a set of recommendations for dealing with the issues with that script [RFC3743] and introduced the concept of "variant" characters and domain names.

However, when names are seen as having meanings, rather than merely being mnemonics, especially when they represent brands or the equivalent, or when spelling for a particular written language is not completely standardized, demands to treat different strings as exact equivalents are obvious and inevitable. As a trivial English-language example, it is widely understood that "colour" and "color" represent the same word, so does that imply that, if they are used as DNS labels in domain names all of whose other labels are identical, the two domain names should be treated as identical? Examples for other languages or writing systems, especially ones in which some or all markings that distinguish characters or words by sound or tone or that change the pronunciation of words are optional, are often more numerous and more problematic than national spelling differences in

English, but they are harder to explain to those unfamiliar with those other languages or writing systems (and hard to illustrate in ASCII-only Internet-Drafts and RFCs). Although approximations are possible, the DNS cannot handle that requirement: not only do its aliasing mechanisms (CNAME, DNAME, and various proposals for newer and different types of aliasing [DNS-Aliases] [DNS-BNAME]) not provide a strong enough binding, but the ability to use those aliases from a subtree controlled by one administrative entity to that of another one implies that there is little or no possibility of the owner (in either the DNS sense or the registrar-registrant one) of a particular name to control the synonyms for it. Some of that issue can be dealt with at the application level, e.g., by redirects in web protocols, but taking that approach, which is the essential characteristic of "if both names belong to the same owner, everything is OK" approaches, results in names being handled in inconsistent ways in different protocols.

A different way of looking at part of this issue (and, to some degree, of the one discussed above in Section 3.3) is that these perceived equivalences and desired transformations are context-dependent, but the DNS resolution process is not [RFC6912].

Similar problems arise as people notice that some characters are easily mistaken for others and that might be an opportunity for user confusion and attacks. Commonly cited examples include the Latin and Cyrillic script "a" characters, which are identical [CACM-Homograph], the characters in many scripts that look like open circles or vertical or horizontal lines, and even the Latin script letter "l" and the European digit "1", but examples abound in other scripts and combinations of scripts as well. The most common proposed solution within the DNS context has been to treat these cases, as well as those involving orthographic variations, as "variants" (but variants different from the system for Chinese characters mentioned above) and either ban all but one (or a few) of the possible labels from the DNS (possibly on a first come, first served basis) or ensure that any collection of such strings that are delegated as assigned to the same ownership (see above). Neither solution is completely satisfactory: if all but one string is excluded, users who guess at a different form, perhaps in trying to transcribe characters from written or printed form, don’t find what they are looking for and, as pointed out above, "same ownership" is sufficient only with carefully designed and administered applications protocol support, and sometimes not then.

Some of these issues are discussed at more length in an ICANN report [ICANN-VIP].

3.5. Query Privacy

There has been growing concern in recent years that DNS queries occur in cleartext on the public Internet and that, if those queries can be intercepted, they can expose a good deal of information about interests and contacts that could compromise individual privacy. While a number of proposals, including query name minimization [RFC7816] and running DNS over an encrypted tunnel [RFC7858], have been made to mitigate that problem, they all appear to share the common properties of security patches rather than designed-in security or privacy mechanisms. While experience may prove otherwise once (and if) they are widely deployed, it does not appear that any of them are as satisfactory as a system with query privacy designed in might be. More general tutorials on this issue have appeared recently [Huston2017a].

3.6. Alternate Namespaces for Public Use in the DNS Framework: The CLASS Problem

The DNS standards include specification of a CLASS value, which "identifies a protocol family or instance of a protocol" (RFC 1034, Section 3.6, and elsewhere). While CLASS was used effectively in the early days of the DNS to manage different protocol families within the same administrative environment, recent attempts to use it to either partition the DNS namespace in other ways such as for non-ASCII names (partially to address the issues in Sections 3.2 and 3.3) or use DNS mechanisms for entirely different namespaces have exposed fundamental problems with the mechanism [Sullivan-Class]. Perhaps the most fundamental of those problems is disagreement about whether multiple CLASSes were intended to exist within a given zone (with records within RRSETs) or whether different CLASSes implied different zones. Different implementations make different assumptions [Faltstrom-2004] [Vixie-20170704]. These problems have led to recommendations that it be dropped entirely [Sullivan-Class], but discussions on the IETF list and in WGs in mid-2017 made it clear that there is no clear consensus on that matter.

3.7. Loose Synchronization

The DNS model of master and slave servers, with the latter initiating updates based on expiration interval values, and local caches with updates based on TTL values, depends heavily on an approach that has come to be called "loose synchronization", i.e., that there can be no expectation that all of the servers that might reasonably answer a query will have exactly the same data unless those data have been unchanged for a rather long period. Put differently, if some or all of the records associated with a particular node in the DNS

(informally, a fully qualified domain name (FQDN)) change, one cannot expect those changes to be propagated immediately.

That model has worked rather well since the DNS was first deployed, protecting the system from requirements for mechanisms that are typical where a simultaneous update of multiple systems is needed. Such mechanisms include elaborate locking, complex update procedures and handshaking, or journaling. As has often been pointed out with the Internet, implementation and operational complexity are often the enemy of stability, security, and robustness. Loose synchronization has helped keep the DNS as simple and robust as possible.

A number of recent ideas about using the DNS to store data for which important changes occur very rapidly are, however, largely incompatible with loose synchronization. Efforts to use very short (or zero) refresh times (in SOA records for slave updates from masters) and TTLs (for caches) to simulate nearly simultaneous updating may work up to a point but appear to impose very heavy loads on servers and distribution mechanisms that were not designed to accommodate that style of working. Similar observations can be made about attempts to use the NOTIFY extension [RFC1996] or dynamic, "server-push", updating rather than the traditional DNS mechanisms. While the NOTIFY and push mechanisms normally provide refresh times and update mechanisms faster than those specified in RFCs 1034 and 1035, they imply that a "master" server must know the identities of (and have good connectivity to all of) its slaves. That defeats at least some of the advantages associated with stealth slaves, particularly those associated with reduction of query traffic across the Internet. Those mechanisms do nothing for cache updates: unless servers keep track of the source of every query for names associated with a specific zone and then somehow notify the query source systems, the only alternative to having information that might be obsolete stored in caches is to use very short or zero TTLs so the cached data time out almost immediately after being stored (or are not stored at all), requiring a new query to an authoritative server each time a resolver attempts to look up a name.

3.8. Private Namespaces and Special Names

Almost since the DNS was first deployed, there have been situations in which it is desirable to use DNS-like names, and often DNS resolution mechanisms or modifications of them, with namespaces for which globally available and consistent resolution using the public DNS is either unfeasible or undesirable (and for which the use of CLASS is not an appropriate mechanism). The need to isolate names and addresses on LANs from the public Internet, typically via "split horizon" approaches, is one example of this requirement although often not recognized as such. Another example that has generated a

good deal of controversy involves "special names" -- labels or pseudo-labels, often in TLD positions, that signal that the full name should not be subject to normal DNS resolution or other processing [RFC6761] [RFC8244].

Independent of troublesome policy questions about who should allocate such names and the procedures to be used, they almost inherently require either a syntax convention to identify them (there actually was such a convention, but it was abandoned many years ago and there is no plausible way to reinstitute it) or tables of such names that are known to, and kept updated on, every resolver on the Internet, at least if spurious queries to the root servers are to be avoided.

If the DNS were to be redesigned and replaced, we could recognize this requirement as part of the design and handle it much better than it is possible to handle it today.

3.9. Alternate Query or Response Encodings

The DNS specifies formats for queries and data responses, based on the state of the art and best practices at the time it was designed. Recent work has suggested that there would be significant advantages to supporting at least a description of the DNS messages in one or more alternate formats, such as JSON [Hoffman-DNS-JSON] [Hoffman-SimpleDNS-JSON]. While that work has been carefully done to avoid requiring changes to the DNS, much of the argument for having such a JSON-based description format could easily be turned into an argument that, if the DNS were being revised, that format might be preferable as a more direct alternative to having DNS queries and responses in the original form.

3.10. Distribution and Management of Root Servers

The DNS model requires a collection of root servers that hold, at minimum, information about top-level domains. Over the years, that requirement has evolved from a technically fairly minor function, normally carried out as a service to the broader Internet community and its users and systems, to a subject that is intensely controversial with regard to control of those servers, including how they should be distributed and where they should be located. While a number of mechanisms, most recently including making the information more local [RFC7706], have been proposed and one (anycast [RFC7094]) is in very active use to mitigate some of the real and perceived problems, it seems obvious that a DNS successor, designed for today’s global Internet and perceived requirements, could handle these problems in a technically more appropriate and less controversial way. Some additional discussion of the issues involved appears in a recent paper [Huston2017b].

3.11. Identifiers versus Brands and Other Convenience Names

A key design element of the original network object naming systems for the ARPANET, largely inherited by the DNS, was that the names, while expected to be mnemonic, were identifiers and their being highly distinguishable and not prone to ambiguity was important. That led to restrictive rules about what could appear in a name. Those restrictions originated with the host table and even earlier [RFC0236] [RFC0247] and came to the DNS (largely via SMTP) as the "preferred syntax" (RFC 1034, Section 3.5) or what we now often call the letter-digit-hyphen (LDH) rule. Similar rules to make identifiers easier to use, less prone to ambiguity, or less likely to interfere with syntax occur frequently in more formal languages. For example, almost every programming language has restrictions on what can appear in an identifier, and Unicode provides general recommendations about identifier composition [Unicode-USA31]. Both are quite restrictive as compared to the number of characters and total number of strings that can be written using that character coding system.

That model, which originally prohibited labels starting with digits in order to avoid any possible confusion with IP addresses, began to break down in 1987 or 1988 when a company named 3Com wanted to use its corporate name as a label within the COM TLD, and the rule was relaxed [RFC1123].

In the last decade or two, the perspective that company names should be supported if possible has expanded and done so largely without its limits, if any, being explicitly understood or acknowledged. In the current form, the DNS is really (and primarily) a system for expressing thoughts and concepts. Those include free expression of ideas in as close to natural language as possible as well as representation of product names and brands. That view requires letter-like characters that might not be reasonable in identifiers along with a variety of symbols and punctuation. It may also require indicators of preferred type styles to provide information in a form that exactly matches personal or legal preferences. At least if carried to an extreme, that perspective would argue for standardizing word and sentence separators, removing the limit of 63 octets per label and probably the limit of 255 octets on the total length of a domain name, and perhaps even eliminating the hierarchy or allowing separators for labels in presentation form (now fixed at "." for the DNS) to be different according to context. It suggests that, at least, the original design was defective in not prioritizing those uses over the more restrictive approach associated with prioritizing unique and unambiguous identifiers.

So we have two or, depending on how one counts, three very different use cases. The historical one is support for unique identifiers. The other is expression of ideas and, if one considers them separate, presentation of brand and product names. Because they inherently involve different constraints, priorities, and success criteria, these perspectives are, at best, only loosely compatible.

We cannot simultaneously optimize both the identifier perspective and either or both of the others in the same system. At best, there are some complex trade-offs involved. Even then, it is not clear that the same DNS (or other system) can accommodate all of them. Until we come to terms with that, the differences manifest themselves with friction among communities, most often with tension between "we want to do (or use or sell) these types of labels" and "not good for the operational Internet or the DNS".

3.12. A Single Hierarchy with a Centrally Controlled Root

A good many Internet policy discussions in the last two decades have revolved around such questions of how many top-level domains there should be, what they should be, who should control them and how, how (or if) their individual operations and policy decisions should be accountable to others, and what processes should be used (and by what entities or organizational structures) to make those decisions. Several people have pointed out that, if we were designing a next-generation DNS using today’s technology, it should be possible to remove the technical requirement for a central authority over the root (some people have suggested that blockchain approaches would be helpful for this purpose; others believe they just would not scale adequately, at least at acceptable cost, but that other options are possible). Whether elimination of a single, centrally controlled, root would be desirable or not is fairly obviously a question of perspective and priorities.

3.13. Newer Application Protocols, New Requirements, and DNS Evolution

New work done in other areas has led to demands for new DNS features, many of them involving data values that require recursively referencing the DNS. Early record types that did that were accompanied by restrictions that reduced the risk of looping references or other difficulties. For example, while the MX RRTYPE has a fully qualified domain name as its data, SMTP imposes "primary name" restrictions that prevent the name used from being, e.g., a CNAME. While loops with CNAMEs are possible, Section 3.6 of RFC 1034 includes a discussion about ways to avoid problems and how they should be handled. Some newer protocols and conventions can cause more stress. There are separate issues with additions and with how the DNS has been extended to try to deal with them.

3.13.1. The Extensions

Some examples of DNS extensions for new protocol demands that illustrate, or have led to, increased stress include:

NAPTR: Requires far more complex data in the DNS for ENUM (e.g., Voice over IP (VoIP), specifically SIP) support, including URI information and hence recursive or repeated lookups, than any of the RRTYPEs originally supported. The RRSET associated with these records can become quite large because the separator between the various records is part of the RDATA, and not the {owner, class, type} triple (a problem slightly related to the problem with overloading of TXT RRTYPE discussed in Section 3.13.2). This problem, and similar ones for some of the cases below. may suggest that any future design is in need of a different TYPE model such as systematic arrangements for subtypes or some explicit hierarchy in the TYPEs.

URI: Has a URI as its data, typically also requiring recursive or repeated lookups.

Service location (SRV) and credential information (including Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM)): Require structured data and, especially for the latter two, significantly more data than most original RRTYPEs.

URI/URL: The early design decision for the World Wide Web that its mechanism for identifying digital web content (now known as Uniform Resource Identifiers [RFC3986]) did so by using domain names and hence the network location of the information or other material. That, in turn, has required systems intended to improve web performance by locating and retrieving a "nearest copy" (rather than the single copy designated by the URL) to intercept DNS queries and respond with values that are not precisely those stored for the designated domain name in the DNS or to otherwise access information in a way not supported by the DNS itself.

3.13.2. Extensions and Deployment Pressures -- The TXT RRTYPE

Unfortunately (but unsurprisingly), and despite IETF efforts to make things easier [RFC6895], DNS support libraries have often been slow to add full support for new RRTYPEs. This has impeded deployment of applications that depend on those types and that must ask (query) explicitly for them. Both to get faster deployment and, at least until recently, to avoid burdensome IETF approval procedures, many application designers have chosen to push protocol-critical

information into records with TXT RRTYPE, a record type that was originally intended to include only information equivalent to comments.

This causes two problems. First, TXT records used this way tend to get long and complex, which is a problem in itself if one is trying to minimize TCP connections. Second, applications that are attempting to obtain data cannot merely ask for the relevant QTYPE; they must obtain all of the records with QTYPE TXT and parse them to determine which ones are of interest. That would be easier if there was some standard for how to do that parsing, but, at least in part because the clear preference in the DNS design is for distinct RRTYPEs for different kinds of information, there is no such standard. (There was a proposal in 1993 to structure the TXT DATA in a way that would have addressed the issue [RFC1464], but it apparently never went anywhere.)

On the other hand, this issue is somewhat different from most of the others described in this document because (as the IETF has recommended several times) the problem is easily solved within the current DNS design by allocating and supporting new RRTYPEs when needed rather than using TXT as a workaround (that does not mean that other solutions are impossible, either with the current DNS or with some other design). The problem then lies in the implementations and/or mechanisms that deter or impede rapid deployment of support for new RRTYPEs.

3.13.3. Periods and Zone Cut Issues

One of the DNS characteristics that is poorly understood by non-experts is that the period (".", U+002E) character can be used in four different ways:

- As a label separator in the presentation form that also designates a "zone break" (delegation boundary). For example, foo.bar.example.com indicates the owner, "foo", of records in the "bar.example.com" zone.

- As a label separator in the presentation form that does not designate a zone break. For example, foo.bar.example.com indicates the owner, "foo.bar", of records in the "example.com" zone.

- As a character within a label, including as a substitute for an at-sign ("@") when an email address appears in an SOA record or in a label that denotes such an address (see Section 2 above). The ability to embed periods in labels in this way has also led to attacks in which, e.g., a domain name consisting of the labels

"example" followed by "com" is deliberately confused with the single label "example.com" with an embedded period.

- At the end of a fully qualified domain name to designate the root zone, e.g., "example.com." (RFC 1034, Section 3.1).

In general, these cases cannot be distinguished by looking at them. The third is problematic for non-DNS reasons, e.g., "john.doe.example.net" can be interpreted as either a simple FQDN or as a notation for john@doe.example.net, john.doe@example.net, or even (at least in principle) john.doe.example@net.

The distinction between the FQDN interpretation and the first email-like one was probably not important as the DNS was originally intended to be used. However, as soon as RRTYPEs (other than NS records that define the zone cut) are used that are sensitive to the boundaries between zones, the distinctions become important to people other than the relevant zone administrators. DNSSEC [RFC4033] involves one such set of relationships. It increases the importance of questions about what should go in a parent zone and what should go in child zones and how much difference it makes if NS records in a parent zone for a child zone are consistent with the records and data in the child zone. This also causes application issues and may raise questions about relationships between registrars and one or more registries or, if they are separate, DNS operators.

3.14. Scaling of Reputation and Other Ancillary Information

The original design for DNS administration, reflected in RFC 1591 [RFC1591] and elsewhere, assumed that all domains would exhibit a very high level of responsibility toward and for the community and that level of responsibility would be enforced if necessary.

More recent decisions, many of them associated with commercialization of the DNS, have eroded those very strong assumptions of registry responsibility and accountability to the point that many consider decisions about delegation of names, identification of registrants, and relationships among names to be matters of "registrant beware" and even "user and applications beware". While some recent protocols and proposals at least partially reflect that original model of a high level of responsibility (see, e.g., IDNA [RFC5890] and a more recent discussion [Klensin-5891bis]), other decisions and actions tend to shift responsibility to the registrant or try to avoid accountability entirely. One possible approach to the problems, especially security problems, that are enabled by those new trends and the associated environment is to establish reputation systems associated with clearly defined administrative boundaries and with

warnings to users, even if those reputation systems are managed by parties not directly associated with the DNS.

The IETF DBOUND WG [IETF-DBOUND] addressed ways to establish and document boundaries more precise than simple dependencies on TLDs, but it was not successful in producing a standard.

A TLD reputation-based approach was adopted by some web browsers after IDNs and a growing number of Generic Top-Level Domains (gTLDs) were introduced; that approach was based on a simple list and does not scale to the current size of the DNS or even the DNS root.

3.15. Tensions among Transport, Scaling, and Content

The original design for the DNS envisaged a simple query and response protocol where both the command and the response could be readily mapped into a single IP packet. The host requirements specification [RFC1123] required all DNS applications to accept a UDP query or response over UDP with up to 512 octets of DNS payload. TCP was seen as a fallback when the response was greater than this 512-octet limit, and this fallback to use TCP as the transport protocol was considered to be the exception rather than the rule.

Over the intervening years, we have seen the rise of a common assumption of an Internet-wide Maximum Transmission Unit (MTU) size of 1,500 octets, accompanied with an assumption that UDP fragmentation is generally viable. This underpins the adoption of the Extension Mechanisms for DNS (EDNS(0)) [RFC6891] to, among other things, specify a UDP buffer size larger than 512 octets and a suggestion within that specification to use 4,096 as a suitable compromise for the UDP payload size. This has proved to be fortuitous for the DNSSEC security extensions where the addition of DNSSEC security credentials in DNS responses [RFC4034] can lead to the use of large DNS responses. However, this exposes some tensions over the handling of fragmentation in IP, where UDP fragments have been observed to be filtered by various firewalls. Additionally for IPv6, there are the factors of filtering the ICMPv6 Packet Too Big diagnostic messages and discarding the IPv6 packets that contain extension headers [RFC7872]. More generally, fragmented UDP packets appear to have a lower level of reliability than unfragmented TCP packets.

Behind this observation about relative reliability of delivery is the tension between the lightweight load of UDP and the downside of elevated probability of discarding of packet fragments as compared to TCP, which offers increased levels of assurance of content delivery, but with the associated imposition of TCP session state and the downside of reduced DNS scalability and increased operational cost.

4. The Inverse Lookup Requirement

The requirement for an inverse lookup capability, i.e., the ability to find a domain name given an address and, in principle, to find the owner of a record by any of its data elements, was recognized in RFC 882. The feature was identified as optional but carried forward into RFCs 1034 and 1035 but was explicitly deprecated by RFC 1034 for address-to-hostname lookup (although RFC 1035 uses exactly that type of lookup in an example). Despite the discussion of inverted forms of the database in RFC 1035, inverse lookup has rarely, if ever, been implemented, at least in its general form. The fundamental difficulties with inverse lookup in either the form described in RFC 882 or the "in-addr.arpa" approach mentioned below are consistent with the problems described in fundamental papers on database management [Codd1970] but were not described in RFC 1035 or related contemporary IETF documents.

It is interesting to speculate on how many of the current requirements to treat aliases as an integrated set of synonyms (e.g., for variant handling) would have been addressed if inverse lookups could reliably produce the owners of CNAME records.

At the same time, it was obviously important to have some mechanism for address-to-name resolution. It was provided by PTR RRTYPE entries in the IN-ADDR.ARPA zone, with delegations on octet boundaries. However, that approach required that information be maintained in parallel, in separate zones, for the name-to-address and address-to-name mappings. That synchronization requirement for two copies of essentially the same data was another popular topic in the database management literature a decade or more before the DNS and, predictably, led to many inconsistencies and other failures.

The introduction of Classless Inter-Domain Routing (CIDR) [RFC1518] and Provider-Dependent addresses made the situation even more difficult, because it was no longer possible to delegate the administration of reverse mapping records for small networks to the actual operators of those networks. ISPs and other aggregators often had no incentive to maintain reverse mapping records consistent with network operator assignment of domain names. A proposal to use binary labels to work around that issue [RFC2673] was abandoned somewhat over three years later [RFC6891].

Independent of how much or little harm the absence of a general inverse lookup facility has caused and how effective the "in-addr.arpa" approach has been, inverse lookup remains a facility that was anticipated and known to be useful in the original DNS design but that has never been fully realized.

5. Internet Scale, Function Support, and Incremental Deployment

In addition to the stresses caused by the new functions, including those described in Section 3.13, incremental deployment of systems that utilize them means that some functions will work in some environments and not others. This has been especially problematic with complex, multi-record, capabilities like DNSSEC that provide or require special validation mechanisms and with some EDNS(0) extensions [RFC6891] that require both the client and server to accept particular extensions. When DNS functionality is required in embedded devices, deployment of new features across the entire Internet in a reasonable period of time is nearly impossible.

If one were redesigning the DNS, one could imagine ways to address these issues that would make them slightly more tractable, and, of course, the features that are known to be necessary today could become part of the baseline, "mandatory to implement", specification.

6. Searching and the DNS -- An Historical Note

Some of the issues identified above might reasonably be addressed, not by changing the DNS itself but by changing our model of what it is about and how it is used. Specifically, one key assumption when the DNS (and the host table system before it) was designed was that it was a naming system for network resources, not, e.g., digital content. As such, exact matching was important, it was reasonable to have labels treated as mnemonics that did not necessarily have linguistic or semantic meaning except to those using them, and so on. A return to that model, presumably by having user-facing applications call on an intermediate layer to disambiguate user-friendly names and map them to DNS names (or network object locators more generally), would significantly reduce stress on the DNS and would also allow dealing with types of matching and similar or synonymous strings that cannot be handled algorithmically no matter how much DNS matching rules were altered.

In some respects, search engines based on free-text analysis and linkages among information have come to serve many of the functions of such an intermediate layer. Many studies and sources have pointed out that few users actually understand, much less care about, the distinction between a DNS name and a search term. Recent versions of some web browsers have both recognized the failure of that distinction and reinforced it by eliminating the separation between "URL" and "search bar".

It is worth noting that, while that "search" approach, or some other approach that abstracted and separated several of the issues identified in Section 3 from the DNS protocol and database themselves, it does not address all of them. At least some elements of several of those issues, such as the synchronization ones described in Section 3.7 and the transport ones described in Section 3.15, are inherent in the DNS design, and, if we are not going to replace the DNS, we had best get used to them.

In the early part of the last decade, the IETF engaged in some preliminary exploration of the intermediate-layer approach in the context of IDNs and what were then called "Internet keywords" [DNS-search]. While that exploratory effort met several times informally, it never became an organized IETF activity, largely because of the choice of what became the IDNA approach but also in part by signs that the "Internet keywords" efforts were beginning to fall apart.

It may be time to reexamine intermediate-layer approaches. If so, the effort should examine use of those approaches by appropriate user-facing applications that might be used to address some of the issues identified above. The Internet and the DNS have changed considerably since the 2000-2003 period. Several of those changes are discussed elsewhere in this document; others, including repurposing of the DNAME RRTYPE from support for transitions [RFC2672] to a general-purpose mechanism for aliases of subtrees [RFC6672] and the addition of over a thousand new TLDs [IANA-TLD-registry], are not but nonetheless are part of the context for intermediate-layer work that did not exist in 2003.

7. Security Considerations

A wide range of security issues related to both securing the DNS and also to abilities to use namespaces for nefarious purposes have arisen. Issues of securing the DNS would obviously be essential to a replacement of the DNS. Issues of preventing nefarious use of the namespace (e.g. use of the name that appears or disappears as a signal to bots) would appear to be harder to solve within the naming system.

8. References

8.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

8.2. Informative References

[CACM-Homograph] Gabrilovich, E. and A. Gontmakher, "The Homograph Attack", Communications of the ACM, Volume 45, Issue 2, pp. 128, DOI 10.1145/503124.503156, February 2002, <http://www.cs.technion.ac.il/~gabr/papers/homograph_full.pdf>.

[Cerf2017] Cerf, V., "Desirable Properties of Internet Identifiers", IEEE Internet Computing, Volume 21, Issue 6, pp. 63-64, DOI 10.1109/MIC.2017.4180839, November/December 2017.

[Codd1970] Codd, E., "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM, Volume 13, Issue 6, pp. 377-387, DOI 10.1145/362384.362685, June 1970, <https://dl.acm.org/citation.cfm?id=362685>.

[DNS-Aliases] Woolf, S., Lee, X., and J. Yao, "Problem Statement: DNS Resolution of Aliased Names", Work in Progress, draft-ietf-dnsext-aliasing-requirements-01, March 2011.

[DNS-BNAME] Yao, J., Lee, X., and P. Vixie, "Bundled DNS Name Redirection", Work in Progress, draft-yao-dnsext-bname-06, May 2016.

[DNS-search] IETF, "Internet Resource Name Search Service (IRNSS)", 2003, <https://datatracker.ietf.org/wg/irnss/about/>.

[Faltstrom-2004] Faltstrom, P. and R. Austein, "Design Choices When Expanding DNS", Work in Progress, draft-ymbk-dns-choices-00, May 2004.

[Hoffman-DNS-JSON]
Hoffman, P., "Representing DNS Messages in JSON", Work in Progress, draft-hoffman-dns-in-json-13, October 2017.

[Hoffman-SimpleDNS-JSON]
Hoffman, P., "Simple DNS Queries and Responses in JSON", Work in Progress, draft-hoffman-simplednsjson-01, November 2017.

[Huston2017a]
Huston, G. and J. Silva Dama, "DNS Privacy", The Internet Protocol Journal, Vol. 20, No. 1, March 2017, <http://ipj.dreamhosters.com/wp-content/uploads/issues/2017/ipj20-1.pdf>.

[Huston2017b]
Huston, G., "The Root of the Domain Name System", The Internet Protocol Journal, Vol. 20, No. 2, pp. 15-25, June 2017, <http://ipj.dreamhosters.com/wp-content/uploads/2017/08/ipj20-2.pdf>.

[IANA-TLD-registry]
Internet Assigned Numbers Authority (IANA), "Root Zone Database", <https://www.iana.org/domains/root/db>.

[ICANN-VIP]
ICANN, "IDN Variant Issues Project: Final Integrated Issues Report Published and Proposed Project Plan for Next Steps is Now Open for Public Comment", February 2012, <https://www.icann.org/news/announcement-2012-02-20-en>.

[IETF-DBOUND]
IETF, "Domain Boundaries (dbound)", 2017, <https://datatracker.ietf.org/wg/dbound/about/>.

[Klensin-5891bis]
Klensin, J. and A. Freytag, "Internationalized Domain Names in Applications (IDNA): Registry Restrictions and Recommendations", Work in Progress, draft-klensin-idna-rfc5891bis-01, September 2017.

[Mockapetris-1988]
Mockapetris, P. and K. Dunlap, "Development of the Domain Name System", SIGCOMM '88 Symposium, pp. 123-133, available from ISI Reprint Series, ISI/RS-88-219 <ftp://ftp.isi.edu/isi-pubs/rs-88-219.pdf>, DOI 10.1145/52324.52338, August 1988, <http://dl.acm.org/citation.cfm?id=52338>.

[NRC-Signposts]
National Research Council, Signposts in Cyberspace: The Domain Name System and Internet Navigation, ISBN 0-309-54979-5, 2005, <https://www.nap.edu/catalog/11258/signposts-in-cyberspace-the-domain-name-system-and-internet-navigation>.

[RFC0236] Postel, J., "Standard host names", RFC 236, DOI 10.17487/RFC0236, September 1971, <https://www.rfc-editor.org/info/rfc236>.

[RFC0247] Karp, P., "Proffered set of standard host names", RFC 247, DOI 10.17487/RFC0247, October 1971, <https://www.rfc-editor.org/info/rfc247>.

[RFC0799] Mills, D., "Internet name domains", RFC 799, DOI 10.17487/RFC0799, September 1981, <https://www.rfc-editor.org/info/rfc799>.

[RFC0810] Feinler, E., Harrenstien, K., Su, Z., and V. White, "DoD Internet host table specification", RFC 810, DOI 10.17487/RFC0810, March 1982, <https://www.rfc-editor.org/info/rfc810>.

[RFC0881] Postel, J., "Domain names plan and schedule", RFC 881, DOI 10.17487/RFC0881, November 1983, <https://www.rfc-editor.org/info/rfc881>.

[RFC0882] Mockapetris, P., "Domain names: Concepts and facilities", RFC 882, DOI 10.17487/RFC0882, November 1983, <https://www.rfc-editor.org/info/rfc882>.

[RFC0883] Mockapetris, P., "Domain names: Implementation specification", RFC 883, DOI 10.17487/RFC0883, November 1983, <https://www.rfc-editor.org/info/rfc883>.

[RFC0952] Harrenstien, K., Stahl, M., and E. Feinler, "DoD Internet host table specification", RFC 952, DOI 10.17487/RFC0952, October 1985, <https://www.rfc-editor.org/info/rfc952>.

[RFC0953] Harrenstien, K., Stahl, M., and E. Feinler, "Hostname Server", RFC 953, DOI 10.17487/RFC0953, October 1985, <https://www.rfc-editor.org/info/rfc953>.

[RFC0974] Partridge, C., "Mail routing and the domain system", STD 10, RFC 974, DOI 10.17487/RFC0974, January 1986, <https://www.rfc-editor.org/info/rfc974>.

[RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <https://www.rfc-editor.org/info/rfc1123>.

[RFC1464] Rosenbaum, R., "Using the Domain Name System To Store Arbitrary String Attributes", RFC 1464, DOI 10.17487/RFC1464, May 1993, <https://www.rfc-editor.org/info/rfc1464>.

[RFC1518] Rekhter, Y. and T. Li, "An Architecture for IP Address Allocation with CIDR", RFC 1518, DOI 10.17487/RFC1518, September 1993, <https://www.rfc-editor.org/info/rfc1518>.

[RFC1591] Postel, J., "Domain Name System Structure and Delegation", RFC 1591, DOI 10.17487/RFC1591, March 1994, <https://www.rfc-editor.org/info/rfc1591>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/info/rfc1996>.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, DOI 10.17487/RFC2671, August 1999, <https://www.rfc-editor.org/info/rfc2671>.

[RFC2672] Crawford, M., "Non-Terminal DNS Name Redirection", RFC 2672, DOI 10.17487/RFC2672, August 1999, <https://www.rfc-editor.org/info/rfc2672>.

[RFC2673] Crawford, M., "Binary Labels in the Domain Name System", RFC 2673, DOI 10.17487/RFC2673, August 1999, <https://www.rfc-editor.org/info/rfc2673>.

[RFC3490] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, DOI 10.17487/RFC3490, March 2003, <https://www.rfc-editor.org/info/rfc3490>.

[RFC3491] Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)", RFC 3491, DOI 10.17487/RFC3491, March 2003, <https://www.rfc-editor.org/info/rfc3491>.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003, <https://www.rfc-editor.org/info/rfc3596>.

[RFC3743] Konishi, K., Huang, K., Qian, H., and Y. Ko, "Joint Engineering Team (JET) Guidelines for Internationalized Domain Names (IDN) Registration and Administration for Chinese, Japanese, and Korean", RFC 3743, DOI 10.17487/RFC3743, April 2004, <https://www.rfc-editor.org/info/rfc3743>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[RFC4343] Eastlake 3rd, D., "Domain Name System (DNS) Case Insensitivity Clarification", RFC 4343, DOI 10.17487/RFC4343, January 2006, <https://www.rfc-editor.org/info/rfc4343>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <https://www.rfc-editor.org/info/rfc5890>.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, DOI 10.17487/RFC5891, August 2010, <https://www.rfc-editor.org/info/rfc5891>.

[RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, DOI 10.17487/RFC6672, June 2012, <https://www.rfc-editor.org/info/rfc6672>.

[RFC6761] Cheshire, S. and M. Krochmal, "Special-Use Domain Names", RFC 6761, DOI 10.17487/RFC6761, February 2013, <https://www.rfc-editor.org/info/rfc6761>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

[RFC6895] Eastlake 3rd, D., "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 6895, DOI 10.17487/RFC6895, April 2013, <https://www.rfc-editor.org/info/rfc6895>.

[RFC6912] Sullivan, A., Thaler, D., Klensin, J., and O. Kolkman, "Principles for Unicode Code Point Inclusion in Labels in the DNS", RFC 6912, DOI 10.17487/RFC6912, April 2013, <https://www.rfc-editor.org/info/rfc6912>.

[RFC7094] McPherson, D., Oran, D., Thaler, D., and E. Osterweil, "Architectural Considerations of IP Anycast", RFC 7094, DOI 10.17487/RFC7094, January 2014, <https://www.rfc-editor.org/info/rfc7094>.

[RFC7706] Kumari, W. and P. Hoffman, "Decreasing Access Time to Root Servers by Running One on Loopback", RFC 7706, DOI 10.17487/RFC7706, November 2015, <https://www.rfc-editor.org/info/rfc7706>.

[RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", RFC 7719, DOI 10.17487/RFC7719, December 2015, <https://www.rfc-editor.org/info/rfc7719>.

[RFC7816] Bortzmeyer, S., "DNS Query Name Minimisation to Improve Privacy", RFC 7816, DOI 10.17487/RFC7816, March 2016, <https://www.rfc-editor.org/info/rfc7816>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", RFC 7872, DOI 10.17487/RFC7872, June 2016, <https://www.rfc-editor.org/info/rfc7872>.

[RFC8244] Lemon, T., Droms, R., and W. Kumari, "Special-Use Domain Names Problem Statement", RFC 8244, DOI 10.17487/RFC8244, October 2017, <https://www.rfc-editor.org/info/rfc8244>.

[Sullivan-Class] Sullivan, A., "The DNS Is Not Classy: DNS Classes Considered Useless", Work in Progress, draft-sullivan-dns-class-useless-03, July 2016.

[Unicode] The Unicode Consortium, The Unicode Standard, Version 9.0.0, (Mountain View, CA: The Unicode Consortium, 2016. ISBN 978-1-936213-13-9), <http://www.unicode.org/versions/Unicode9.0.0/>.

[Unicode-UAX15] Davis, M. and K. Whistler, "Unicode Standard Annex #15: Unicode Normalization Forms", February 2016, <http://unicode.org/reports/tr15/>.

[Unicode-USA31] Davis, M., "Unicode Standard Annex #31: Unicode Identifier and Pattern Syntax", May 2016, <http://unicode.org/reports/tr31/>.

[Vixie-20170704] Vixie, P., "Subject: Re: new DNS classes", message to the IETF dnsop mailing list, 4 July 2017, <https://www.ietf.org/mail-archive/web/ietf/current/msg103486.html>.

Acknowledgements

Many of the concerns and ideas described in this document reflect conversations over a period of many years, some rooted in DNS "keyword" and "search" discussions that paralleled the development of IDNs. Conversations with, or writings of, Rob Austein, Christine Borgman, Carolina Carvalho, Vint Cerf, Lyman Chapin, Nazli Choucri, Patrik Faltstrom, Geoff Huston, Xiaodong Lee, Karen Liu, Gervase Markham, Yaqub Mueller, Andrew Sullivan, Paul Twomey, Nico Williams, Suzanne Woolf, Jiankang Yao, other participants in the circa 2003 "DNS Search" effort and in the ICANN SSAC Working Party on IDNs, and some others whose names were sadly forgotten, were particularly important to either the content of this document or the motivation for writing it even though they may not agree with the conclusions I have reached and bear no responsibility for them.

Many of the subsections of Section 3 were extracted from comments first made in conjunction with recent email discussions. Comments from Suzanne Woolf about an earlier draft version were particularly important as was material developed with suggestions from Patrik Faltstrom, especially Section 3.13. Feedback and suggestions from several of the above and from Stephane Bortzmeyer, Tony Finch, Bob Harold, Warren Kumari, Craig Partridge, and George Sadowsky were extremely helpful for improving the clarity and accuracy of parts of the document, especially so for a broader audience. Craig Partridge also contributed much of the material about queries for multiple types. Geoff Huston made several useful comments and contributed most of Section 3.15, and Bill Manning pointed out some broader requirements about integrity of information and DNS management and operations.

Special thanks are due to Karen Moore of the RFC Production Center for her efforts, patience, and persistence in preparing this document for publication, a process that raised far more issues that required careful discussion than usual.

Author’s Address

John C. Klensin
1770 Massachusetts Ave, Ste 322
Cambridge, MA 02140
United States of America

Phone: +1 617 245 1457
Email: john-ietf@jck.com