

Domain Name System (DNS) Case Insensitivity Clarification

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

Domain Name System (DNS) names are "case insensitive". This document explains exactly what that means and provides a clear specification of the rules. This clarification updates RFCs 1034, 1035, and 2181.

Table of Contents

1. Introduction	2
2. Case Insensitivity of DNS Labels	2
2.1. Escaping Unusual DNS Label Octets	2
2.2. Example Labels with Escapes	3
3. Name Lookup, Label Types, and CLASS	3
3.1. Original DNS Label Types	4
3.2. Extended Label Type Case Insensitivity Considerations	4
3.3. CLASS Case Insensitivity Considerations	4
4. Case on Input and Output	5
4.1. DNS Output Case Preservation	5
4.2. DNS Input Case Preservation	5
5. Internationalized Domain Names	6
6. Security Considerations	6
7. Acknowledgements	7
Normative References.....	7
Informative References.....	8

1. Introduction

The Domain Name System (DNS) is the global hierarchical replicated distributed database system for Internet addressing, mail proxy, and other information. Each node in the DNS tree has a name consisting of zero or more labels [STD13, RFC1591, RFC2606] that are treated in a case insensitive fashion. This document clarifies the meaning of "case insensitive" for the DNS. This clarification updates RFCs 1034, 1035 [STD13], and [RFC2181].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Case Insensitivity of DNS Labels

DNS was specified in the era of [ASCII]. DNS names were expected to look like most host names or Internet email address right halves (the part after the at-sign, "@") or to be numeric, as in the in-addr.arpa part of the DNS name space. For example,

```
foo.example.net.  
aol.com.  
www.gnu.ai.mit.edu.  
or 69.2.0.192.in-addr.arpa.
```

Case-varied alternatives to the above [RFC3092] would be DNS names like

```
Foo.Example.net.  
AOL.COM.  
WWW.gnu.AI.mit.EDU.  
or 69.2.0.192.in-ADDR.ARPA.
```

However, the individual octets of which DNS names consist are not limited to valid ASCII character codes. They are 8-bit bytes, and all values are allowed. Many applications, however, interpret them as ASCII characters.

2.1. Escaping Unusual DNS Label Octets

In Master Files [STD13] and other human-readable and -writable ASCII contexts, an escape is needed for the byte value for period (0x2E, ".") and all octet values outside of the inclusive range from 0x21 ("!") to 0x7E ("~"). That is to say, 0x2E and all octet values in the two inclusive ranges from 0x00 to 0x20 and from 0x7F to 0xFF.

One typographic convention for octets that do not correspond to an ASCII printing graphic is to use a back-slash followed by the value of the octet as an unsigned integer represented by exactly three decimal digits.

The same convention can be used for printing ASCII characters so that they will be treated as a normal label character. This includes the back-slash character used in this convention itself, which can be expressed as `\092` or `\\`, and the special label separator period (`."`), which can be expressed as `\046` or `\\.` It is advisable to avoid using a backslash to quote an immediately following non-printing ASCII character code to avoid implementation difficulties.

A back-slash followed by only one or two decimal digits is undefined. A back-slash followed by four decimal digits produces two octets, the first octet having the value of the first three digits considered as a decimal number, and the second octet being the character code for the fourth decimal digit.

2.2. Example Labels with Escapes

The first example below shows embedded spaces and a period (`."`) within a label. The second one shows a 5-octet label where the second octet has all bits zero, the third is a backslash, and the fourth octet has all bits one.

```
Donald\032E\.\032Eastlake\0323rd.example.  
and  a\000\\\255z.example.
```

3. Name Lookup, Label Types, and CLASS

According to the original DNS design decision, comparisons on name lookup for DNS queries should be case insensitive [STD13]. That is to say, a lookup string octet with a value in the inclusive range from `0x41` to `0x5A`, the uppercase ASCII letters, MUST match the identical value and also match the corresponding value in the inclusive range from `0x61` to `0x7A`, the lowercase ASCII letters. A lookup string octet with a lowercase ASCII letter value MUST similarly match the identical value and also match the corresponding value in the uppercase ASCII letter range.

(Historical note: The terms "uppercase" and "lowercase" were invented after movable type. The terms originally referred to the two font trays for storing, in partitioned areas, the different physical type elements. Before movable type, the nearest equivalent terms were "majuscule" and "minuscule".)

One way to implement this rule would be to subtract 0x20 from all octets in the inclusive range from 0x61 to 0x7A before comparing octets. Such an operation is commonly known as "case folding", but implementation via case folding is not required. Note that the DNS case insensitivity does NOT correspond to the case folding specified in [ISO-8859-1] or [ISO-8859-2]. For example, the octets 0xDD (\221) and 0xFD (\253) do NOT match, although in other contexts, where they are interpreted as the upper- and lower-case version of "Y" with an acute accent, they might.

3.1. Original DNS Label Types

DNS labels in wire-encoded names have a type associated with them. The original DNS standard [STD13] had only two types: ASCII labels, with a length from zero to 63 octets, and indirect (or compression) labels, which consist of an offset pointer to a name location elsewhere in the wire encoding on a DNS message. (The ASCII label of length zero is reserved for use as the name of the root node of the name tree.) ASCII labels follow the ASCII case conventions described herein and, as stated above, can actually contain arbitrary byte values. Indirect labels are, in effect, replaced by the name to which they point, which is then treated with the case insensitivity rules in this document.

3.2. Extended Label Type Case Insensitivity Considerations

DNS was extended by [RFC2671] so that additional label type numbers would be available. (The only such type defined so far is the BINARY type [RFC2673], which is now Experimental [RFC3363].)

The ASCII case insensitivity conventions only apply to ASCII labels; that is to say, label type 0x0, whether appearing directly or invoked by indirect labels.

3.3. CLASS Case Insensitivity Considerations

As described in [STD13] and [RFC2929], DNS has an additional axis for data location called CLASS. The only CLASS in global use at this time is the "IN" (Internet) CLASS.

The handling of DNS label case is not CLASS dependent. With the original design of DNS, it was intended that a recursive DNS resolver be able to handle new CLASSES that were unknown at the time of its implementation. This requires uniform handling of label case insensitivity. Should it become desirable, for example, to allocate a CLASS with "case sensitive ASCII labels", it would be necessary to allocate a new label type for these labels.

4. Case on Input and Output

While ASCII label comparisons are case insensitive, [STD13] says case MUST be preserved on output and preserved when convenient on input. However, this means less than it would appear, since the preservation of case on output is NOT required when output is optimized by the use of indirect labels, as explained below.

4.1. DNS Output Case Preservation

[STD13] views the DNS namespace as a node tree. ASCII output is as if a name were marshaled by taking the label on the node whose name is to be output, converting it to a typographically encoded ASCII string, walking up the tree outputting each label encountered, and preceding all labels but the first with a period ("."). Wire output follows the same sequence, but each label is wire encoded, and no periods are inserted. No "case conversion" or "case folding" is done during such output operations, thus "preserving" case. However, to optimize output, indirect labels may be used to point to names elsewhere in the DNS answer. In determining whether the name to be pointed to (for example, the QNAME) is the "same" as the remainder of the name being optimized, the case insensitive comparison specified above is done. Thus, such optimization may easily destroy the output preservation of case. This type of optimization is commonly called "name compression".

4.2. DNS Input Case Preservation

Originally, DNS data came from an ASCII Master File as defined in [STD13] or a zone transfer. DNS Dynamic update and incremental zone transfers [RFC1995] have been added as a source of DNS data [RFC2136, RFC3007]. When a node in the DNS name tree is created by any of such inputs, no case conversion is done. Thus, the case of ASCII labels is preserved if they are for nodes being created. However, when a name label is input for a node that already exists in DNS data being held, the situation is more complex. Implementations are free to retain the case first loaded for such a label, to allow new input to override the old case, or even to maintain separate copies preserving the input case.

For example, if data with owner name "foo.bar.example" [RFC3092] is loaded and then later data with owner name "xyz.BAR.example" is input, the name of the label on the "bar.example" node (i.e., "bar") might or might not be changed to "BAR" in the DNS stored data. Thus, later retrieval of data stored under "xyz.bar.example" in this case can use "xyz.BAR.example" in all returned data, use "xyz.bar.example" in all returned data, or even, when more than one RR is being returned, use a mixture of these two capitalizations. This last case

is unlikely, as optimization of answer length through indirect labels tends to cause only one copy of the name tail ("bar.example" or "BAR.example") to be used for all returned RRs. Note that none of this has any effect on the number or completeness of the RR set returned, only on the case of the names in the RR set returned.

The same considerations apply when inputting multiple data records with owner names differing only in case. For example, if an "A" record is the first resource record stored under owner name "xyz.BAR.example" and then a second "A" record is stored under "XYZ.BAR.example", the second MAY be stored with the first (lower case initial label) name, the second MAY override the first so that only an uppercase initial label is retained, or both capitalizations MAY be kept in the DNS stored data. In any case, a retrieval with either capitalization will retrieve all RRs with either capitalization.

Note that the order of insertion into a server database of the DNS name tree nodes that appear in a Master File is not defined so that the results of inconsistent capitalization in a Master File are unpredictable output capitalization.

5. Internationalized Domain Names

A scheme has been adopted for "internationalized domain names" and "internationalized labels" as described in [RFC3490, RFC3454, RFC3491, and RFC3492]. It makes most of [UNICODE] available through a separate application level transformation from internationalized domain name to DNS domain name and from DNS domain name to internationalized domain name. Any case insensitivity that internationalized domain names and labels have varies depending on the script and is handled entirely as part of the transformation described in [RFC3454] and [RFC3491], which should be seen for further details. This is not a part of the DNS as standardized in STD 13.

6. Security Considerations

The equivalence of certain DNS label types with case differences, as clarified in this document, can lead to security problems. For example, a user could be confused by believing that two domain names differing only in case were actually different names.

Furthermore, a domain name may be used in contexts other than the DNS. It could be used as a case sensitive index into some database or file system. Or it could be interpreted as binary data by some integrity or authentication code system. These problems can usually be handled by using a standardized or "canonical" form of the DNS

ASCII type labels; that is, always mapping the ASCII letter value octets in ASCII labels to some specific pre-chosen case, either uppercase or lower case. An example of a canonical form for domain names (and also a canonical ordering for them) appears in Section 6 of [RFC4034]. See also [RFC3597].

Finally, a non-DNS name may be stored into DNS with the false expectation that case will always be preserved. For example, although this would be quite rare, on a system with case sensitive email address local parts, an attempt to store two Responsible Person (RP) [RFC1183] records that differed only in case would probably produce unexpected results that might have security implications. That is because the entire email address, including the possibly case sensitive local or left-hand part, is encoded into a DNS name in a readable fashion where the case of some letters might be changed on output as described above.

7. Acknowledgements

The contributions to this document by Rob Austein, Olafur Gudmundsson, Daniel J. Anderson, Alan Barrett, Marc Blanchet, Dana, Andreas Gustafsson, Andrew Main, Thomas Narten, and Scott Seligman are gratefully acknowledged.

Normative References

- [ASCII] ANSI, "USA Standard Code for Information Interchange", X3.4, American National Standards Institute: New York, 1968.
- [RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, August 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.
- [RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, November 2000.

- [RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, September 2003.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.
- [STD13] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

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

Informative References

- [ISO-8859-1] International Standards Organization, Standard for Character Encodings, Latin-1.
- [ISO-8859-2] International Standards Organization, Standard for Character Encodings, Latin-2.
- [RFC1183] Everhart, C., Mamakos, L., Ullmann, R., and P. Mockapetris, "New DNS RR Definitions", RFC 1183, October 1990.
- [RFC1591] Postel, J., "Domain Name System Structure and Delegation", RFC 1591, March 1994.
- [RFC2606] Eastlake 3rd, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, June 1999.
- [RFC2929] Eastlake 3rd, D., Brunner-Williams, E., and B. Manning, "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 2929, September 2000.
- [RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.
- [RFC2673] Crawford, M., "Binary Labels in the Domain Name System", RFC 2673, August 1999.
- [RFC3092] Eastlake 3rd, D., Manros, C., and E. Raymond, "Etymology of "Foo"", RFC 3092, 1 April 2001.
- [RFC3363] Bush, R., Durand, A., Fink, B., Gudmundsson, O., and T. Hain, "Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System (DNS)", RFC 3363, August 2002.

- [RFC3454] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("stringprep")", RFC 3454, December 2002.
- [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.
- [UNICODE] The Unicode Consortium, "The Unicode Standard", <<http://www.unicode.org/unicode/standard/standard.html>>.

Author's Address

Donald E. Eastlake 3rd
Motorola Laboratories
155 Beaver Street
Milford, MA 01757 USA

Phone: +1 508-786-7554 (w)
Email: Donald.Eastlake@motorola.com

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

