

IPv6 Maintenance Working Group  
Internet-Draft  
Updates: [4291](#) (if approved)  
Intended status: Standards Track  
Expires: August 21, 2010

S. Kawamura  
NEC BIGLOBE, Ltd.  
M. Kawashima  
NEC AccessTechnica, Ltd.  
February 17, 2010

A Recommendation for IPv6 Address Text Representation  
draft-ietf-6man-text-addr-representation-05

## Abstract

As IPv6 network grows, there will be more engineers and also non-engineers who will have the need to use an IPv6 address in text. While the IPv6 address architecture [RFC 4291 section 2.2](#) depicts a flexible model for text representation of an IPv6 address, this flexibility has been causing problems for operators, system engineers, and users. This document will describe the problems that a flexible text representation has been causing. This document also recommends a canonical representation format that best avoids confusion. It is expected that the canonical format is followed by humans and systems when representing IPv6 addresses as text, but all implementations must accept and be able to handle any legitimate [RFC4291](#) format.

## Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on August 21, 2010.

---

Internet-Draft

IPv6 Text Representation

February 2010

## Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](http://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

Internet-Draft

IPv6 Text Representation

February 2010

## Table of Contents

<a href="#">1.</a>	<a href="#">Introduction . . . . .</a>	<a href="#">4</a>
<a href="#">1.1.</a>	<a href="#">Requirements Language . . . . .</a>	<a href="#">4</a>
<a href="#">2.</a>	<a href="#">Text Representation Flexibility of <a href="#">RFC4291</a> . . . . .</a>	<a href="#">4</a>
<a href="#">2.1.</a>	<a href="#">Leading Zeros in a 16 Bit Field . . . . .</a>	<a href="#">4</a>
<a href="#">2.2.</a>	<a href="#">Zero Compression . . . . .</a>	<a href="#">5</a>
<a href="#">2.3.</a>	<a href="#">Uppercase or Lowercase . . . . .</a>	<a href="#">5</a>
<a href="#">3.</a>	<a href="#">Problems Encountered with the Flexible Model . . . . .</a>	<a href="#">6</a>
<a href="#">3.1.</a>	<a href="#">Searching . . . . .</a>	<a href="#">6</a>
<a href="#">3.1.1.</a>	<a href="#">General Summary . . . . .</a>	<a href="#">6</a>
<a href="#">3.1.2.</a>	<a href="#">Searching Spreadsheets and Text Files . . . . .</a>	<a href="#">6</a>
<a href="#">3.1.3.</a>	<a href="#">Searching with Whois . . . . .</a>	<a href="#">6</a>
<a href="#">3.1.4.</a>	<a href="#">Searching for an Address in a Network Diagram . . . . .</a>	<a href="#">7</a>
<a href="#">3.2.</a>	<a href="#">Parsing and Modifying . . . . .</a>	<a href="#">7</a>
<a href="#">3.2.1.</a>	<a href="#">General Summary . . . . .</a>	<a href="#">7</a>
<a href="#">3.2.2.</a>	<a href="#">Logging . . . . .</a>	<a href="#">7</a>
<a href="#">3.2.3.</a>	<a href="#">Auditing: Case 1 . . . . .</a>	<a href="#">7</a>
<a href="#">3.2.4.</a>	<a href="#">Auditing: Case 2 . . . . .</a>	<a href="#">8</a>
<a href="#">3.2.5.</a>	<a href="#">Verification . . . . .</a>	<a href="#">8</a>
<a href="#">3.2.6.</a>	<a href="#">Unexpected Modifying . . . . .</a>	<a href="#">8</a>
<a href="#">3.3.</a>	<a href="#">Operating . . . . .</a>	<a href="#">8</a>
<a href="#">3.3.1.</a>	<a href="#">General Summary . . . . .</a>	<a href="#">8</a>
<a href="#">3.3.2.</a>	<a href="#">Customer Calls . . . . .</a>	<a href="#">8</a>
<a href="#">3.3.3.</a>	<a href="#">Abuse . . . . .</a>	<a href="#">9</a>
<a href="#">3.4.</a>	<a href="#">Other Minor Problems . . . . .</a>	<a href="#">9</a>
<a href="#">3.4.1.</a>	<a href="#">Changing Platforms . . . . .</a>	<a href="#">9</a>
<a href="#">3.4.2.</a>	<a href="#">Preference in Documentation . . . . .</a>	<a href="#">9</a>
<a href="#">3.4.3.</a>	<a href="#">Legibility . . . . .</a>	<a href="#">9</a>
<a href="#">4.</a>	<a href="#">A Recommendation for IPv6 Text Representation . . . . .</a>	<a href="#">9</a>
<a href="#">4.1.</a>	<a href="#">Handling Leading Zeros in a 16 Bit Field . . . . .</a>	<a href="#">10</a>
<a href="#">4.2.</a>	<a href="#">"::" Usage . . . . .</a>	<a href="#">10</a>
<a href="#">4.2.1.</a>	<a href="#">Shorten As Much As Possible . . . . .</a>	<a href="#">10</a>
<a href="#">4.2.2.</a>	<a href="#">Handling One 16 Bit 0 Field . . . . .</a>	<a href="#">10</a>
<a href="#">4.2.3.</a>	<a href="#">Choice in Placement of "::" . . . . .</a>	<a href="#">10</a>
<a href="#">4.3.</a>	<a href="#">Lower Case . . . . .</a>	<a href="#">10</a>

<a href="#">5.</a>	Text Representation of Special Addresses . . . . .	<a href="#">10</a>
<a href="#">6.</a>	Notes on Combining IPv6 Addresses with Port Numbers . . . . .	<a href="#">11</a>
<a href="#">7.</a>	Prefix Representation . . . . .	<a href="#">12</a>
<a href="#">8.</a>	Security Considerations . . . . .	<a href="#">12</a>
<a href="#">9.</a>	IANA Considerations . . . . .	<a href="#">12</a>
<a href="#">10.</a>	Acknowledgements . . . . .	<a href="#">12</a>
<a href="#">11.</a>	References . . . . .	<a href="#">12</a>
<a href="#">11.1.</a>	Normative References . . . . .	<a href="#">12</a>
<a href="#">11.2.</a>	Informative References . . . . .	<a href="#">13</a>
<a href="#">Appendix A.</a>	For Developers . . . . .	<a href="#">13</a>
	Authors' Addresses . . . . .	<a href="#">13</a>

## [1.](#) Introduction

A single IPv6 address can be text represented in many ways. Examples are shown below.

```
2001:db8:0:0:1:0:0:1
```

```
2001:0db8:0:0:1:0:0:1
```

```
2001:db8::1:0:0:1
```

```
2001:db8::0:1:0:0:1
```

```
2001:0db8::1:0:0:1
```

```
2001:db8:0:0:1::1
```

```
2001:db8:0000:0:1::1
```

```
2001:DB8:0:0:1::1
```

All the above represent the same IPv6 address. This flexibility has caused many problems for operators, systems engineers, and customers. The problems will be noted in [Section 3](#). Also, a canonical representation format to avoid problems will be introduced in [Section 4](#).

### [1.1.](#) Requirements Language

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.](#) Text Representation Flexibility of [RFC4291](#)

Examples of flexibility in [Section 2.2 of \[RFC4291\]](#) are described below.

### [2.1.](#) Leading Zeros in a 16 Bit Field

'It is not necessary to write the leading zeros in an individual field.'

In other words, it is also not necessary to omit leading zeros. This means that, it is possible to select from such as the following example. The final 16 bit field is different, but all these addresses mean the same.

2001:db8:aaaa:bbbb:cccc:dddd:eeee:0001

2001:db8:aaaa:bbbb:cccc:dddd:eeee:001

2001:db8:aaaa:bbbb:cccc:dddd:eeee:01

2001:db8:aaaa:bbbb:cccc:dddd:eeee:1

### [2.2.](#) Zero Compression

'A special syntax is available to compress the zeros. The use of "::" indicates one or more groups of 16 bits of zeros.'

It is possible to select whether or not to omit just one 16 bits of zeros.

2001:db8:aaaa:bbbb:cccc:dddd::1

2001:db8:aaaa:bbbb:cccc:dddd:0:1

In case where there is more than one zero fields, there is a choice of how many fields can be shortened. Examples follow.

2001:db8:0:0:0::1

2001:db8:0:0::1

2001:db8:0::1

2001:db8::1

In addition, [[RFC4291](#)] in [section 2.2](#) notes,

'The ":" can only appear once in an address.'

This gives a choice on where, in a single address to compress the zero. Examples are shown below.

2001:db8::aaaa:0:0:1

2001:db8:0:0:aaaa::1

### [2.3.](#) Uppercase or Lowercase

[RFC4291] does not mention about preference of uppercase or lowercase. Various flavors are shown below.

2001:db8:aaaa:bbbb:cccc:dddd:eeee:aaaa

2001:db8:aaaa:bbbb:cccc:dddd:eeee:AAAA

2001:db8:aaaa:bbbb:cccc:dddd:eeee:AaAa

## [3.](#) Problems Encountered with the Flexible Model

### [3.1.](#) Searching

#### [3.1.1.](#) General Summary

A search of an IPv6 address if conducted through a UNIX system is usually case sensitive and extended options to allow for regular

expression use will come in handy. However, there are many applications in the Internet today that do not provide this capability. When searching for an IPv6 address in such systems, the system engineer will have to try each and every possibility to search for an address. This has critical impacts especially when trying to deploy IPv6 over an enterprise network.

#### [3.1.2.](#) Searching Spreadsheets and Text Files

Spreadsheet applications and text editors on GUI systems, rarely have the ability to search for a text using regular expression. Moreover, there are many non-engineers (who are not aware of case sensitivity and regular expression use) that use these application to manage IP addresses. This has worked quite well with IPv4 since text representation in IPv4 has very little flexibility. There is no incentive to encourage these non-engineers to change their tool or learn regular expression when they decide to go dual-stack. If the entry in the spreadsheet reads, 2001:db8::1:0:0:1, but the search was conducted as 2001:db8:0:0:1::1, this will show a result of no match. One example where this will cause problem is, when the search is being conducted to assign a new address from a pool, and a check was being done to see if it was not in use. This may cause problems to the end-hosts or end-users. This type of address management is very often seen in enterprise networks and also in ISPs.

#### [3.1.3.](#) Searching with Whois

The "whois" utility is used by a wide range of people today. When a record is set to a database, one will likely check the output to see if the entry is correct. If an entity was recorded as 2001:db8::/48, but the whois output showed 2001:0db8:0000::/48, most non-engineers would think that their input was wrong, and will likely retry several times or make a frustrated call to the database hostmaster. If there

was a need to register the same address on different systems, and each system showed a different text representation, this would confuse people even more. Although this document focuses on addresses rather than prefixes, this is worth mentioning since problems encountered are mostly equal.

#### [3.1.4.](#) Searching for an Address in a Network Diagram

Network diagrams and blue-prints often show what IP addresses are assigned to a system devices. In times of trouble shooting, there may be a need to search through a diagram to find the point of failure (for example, if a traceroute stopped at 2001:db8::1, one would search the diagram for that address). This is a technique quite often in use in enterprise networks and managed services. Again, the different flavors of text representation will result in a time-consuming search, leading to longer MTTR in times of trouble.

## [3.2.](#) Parsing and Modifying

### [3.2.1.](#) General Summary

With all the possible text representation ways, each application must include a module, object, link, etc. to a function that will parse IPv6 addresses in a manner that no matter how it is represented, they will mean the same address. Many system engineers who integrate complex computer systems to corporate customers will have difficulties finding that their favorite tool will not have this function, or will encounter difficulties such as having to rewrite their macro's or scripts for their customers.

### [3.2.2.](#) Logging

If an application were to output a log summary that represented the address in full (such as 2001:0db8:0000:0000:1111:2222:3333:4444), the output would be highly unreadable compared to the IPv4 output. The address would have to be parsed and reformed to make it useful for human reading. Sometimes, logging for critical systems is done by mirroring the same traffic to two different systems. Care must be taken that no matter what the log output is, the logs should be parsed so they will mean the same.

### [3.2.3.](#) Auditing: Case 1

When a router or any other network appliance machine configuration is audited, there are many methods to compare the configuration information of a node. Sometimes, auditing will be done by just comparing the changes made each day. In this case, if configuration was done such that 2001:db8::1 was changed to 2001:0db8:0000:0000:



it was better, a simple diff will show that a different address was configured. If this was done on a wide scale network, people will be focusing on 'why the extra zeros were put in' instead of doing any real auditing. Lots of tools are just plain 'diff's that do not take into account address representation rules.

#### [3.2.4.](#) Auditing: Case 2

Node configurations will be matched against an information system that manages IP addresses. If output notation is different, there will need to be a script that is implemented to cover for this. The result of an SNMP GET operation, converted to text and compared to a textual address written by a human is highly unlikely to match on first try.

#### [3.2.5.](#) Verification

Some protocols require certain data fields to be verified. One example of this is X.509 certificates. If an IPv6 address field in a certificate was incorrectly verified by converting it to text and making a simple textual comparison to some other address, the certificate may be mistakenly shown as being invalid due to a difference in text representation methods.

#### [3.2.6.](#) Unexpected Modifying

Sometimes, a system will take an address and modify it as a convenience. For example, a system may take an input of 2001:0db8:0::1 and make the output 2001:db8::1. If the zeros were input for a reason, the outcome may be somewhat unexpected.

### [3.3.](#) Operating

#### [3.3.1.](#) General Summary

When an operator sets an IPv6 address of a system as 2001:db8:0:0:1:0:0:1, the system may take the address and show the configuration result as 2001:DB8::1:0:0:1. Someone familiar with IPv6 address representation will know that the right address is set, but not everyone may understand this.

#### [3.3.2.](#) Customer Calls

When a customer calls to inquire about a suspected outage, IPv6 address representation should be handled with care. Not all customers are engineers nor have the same skill in IPv6 technology. The network operations center will have to take extra steps to

humanly parse the address to avoid having to explain to the customers that 2001:db8:0:1::1 is the same as 2001:db8::1:0:0:0:1. This is one thing that will never happen in IPv4 because IPv4 address cannot be abbreviated.

### [3.3.3.](#) Abuse

Network abuse is reported along with the abusing IP address. This 'reporting' could take any shape or form of the flexible model. A team that handles network abuse must be able to tell the difference between a 2001:db8::1:0:1 and 2001:db8:1::0:1. Mistakes in the placement of the "::" will result in a critical situation. A system that handles these incidents should be able to handle any type of input and parse it in a correct manner. Also, incidents are reported over the phone. It is unnecessary to report if the letter is an uppercase or lowercase. However, when a letter is spelled uppercase, people tend to clarify that it is uppercase, which is unnecessary information.

## [3.4.](#) Other Minor Problems

### [3.4.1.](#) Changing Platforms

When an engineer decides to change the platform of a running service, the same code may not work as expected due to the difference in IPv6 address text representation. Usually, a change in a platform (e.g. Unix to Windows, Cisco to Juniper) will result in a major change of code anyway, but flexibility in address representation will increase the work load.

### [3.4.2.](#) Preference in Documentation

A document that is edited by more than one author, may become harder to read.

### [3.4.3.](#) Legibility

Capital case D and 0 can be quite often misread. Capital B and 8 can also be misread.

## [4.](#) A Recommendation for IPv6 Text Representation

A recommendation for a canonical text representation format of IPv6 addresses is presented in this section. The recommendation in this document is one that, complies fully with [[RFC4291](#)], is implemented

by various operating systems, and is human friendly. The recommendation in this section SHOULD be followed by systems when

generating an address to represent as text, but all implementations MUST accept and be able to handle any legitimate [[RFC4291](#)] format. It is advised that humans also follow these recommendations when spelling an address.

#### [4.1.](#) Handling Leading Zeros in a 16 Bit Field

Leading zeros MUST be suppressed. For example 2001:0db8::0001 is not acceptable and must be represented as 2001:db8::1. A single 16 bit 0000 field MUST be represented as 0.

#### [4.2.](#) "::" Usage

##### [4.2.1.](#) Shorten As Much As Possible

The use of symbol "::" MUST be used to its maximum capability. For example, 2001:db8::0:1 is not acceptable, because the symbol "::" could have been used to produce a shorter representation 2001:db8::1.

##### [4.2.2.](#) Handling One 16 Bit 0 Field

The symbol "::" MUST NOT be used to shorten just one 16 bit 0 field. For example, the representation 2001:db8:0:1:1:1:1:1 is correct, but 2001:db8::1:1:1:1:1 is not correct.

##### [4.2.3.](#) Choice in Placement of "::"

When there is an alternative choice in the placement of a "::", the longest run of consecutive 16 bit 0 fields MUST be shortened (i.e. the sequence with three consecutive zero fields is shortened in 2001:0:0:1:0:0:0:1). When the length of the consecutive 16 bit 0 fields are equal (i.e. 2001:db8:0:0:1:0:0:1), the first sequence of zero bits MUST be shortened. For example 2001:db8::1:0:0:1 is correct representation.

#### [4.3.](#) Lower Case

The characters "a", "b", "c", "d", "e", "f" in an IPv6 address MUST be represented in lower case.

## 5. Text Representation of Special Addresses

Addresses such as IPv4-Mapped IPv6 addresses, ISATAP [[RFC5214](#)], and IPv4-translatable addresses [[I-D.ietf-behave-address-format](#)] have IPv4 addresses embedded in the low-order 32 bits of the address. These addresses have special representation that may mix hexadecimal and dot decimal notations. The decimal notation may be used only for

the last 32 bits of the address. For these addresses, mixed notation is RECOMMENDED if the following condition is met: The address can be distinguished as having IPv4 addresses embedded in the lower 32 bits solely from the address field through the use of a well known prefix. Such prefixes are defined in [[RFC4291](#)] and [[RFC5214](#)] at the time of writing. If it is known by some external method that a given prefix includes an IPv4 address, it MAY be represented as mixed notation. Tools that provide options to specify prefixes that is (or is not) to be represented as mixed notation may be useful.

There is a trade-off here where a recommendation to achieve exact match in a search (no dot decimals whatsoever) and recommendation to help the readability of an addresses (dot decimal whenever possible) does not result in the same solution. The above recommendation is aimed at fixing the representation as much as possible while leaving the opportunity for future well known prefixes to be represented in a human friendly manner as tools adjust to newly assigned prefixes.

The text representation method noted in [Section 4](#) should be applied for the leading hexadecimal part (i.e. `::ffff:192.0.2.1` instead of `0:0:0:0:ffff:192.0.2.1`).

## 6. Notes on Combining IPv6 Addresses with Port Numbers

When IPv6 addresses and port numbers are represented in text combined together, there are many different ways to do so. Examples are shown below.

- o `[2001:db8::1]:80`
- o `2001:db8::1:80`

- o 2001:db8::1.80
- o 2001:db8::1 port 80
- o 2001:db8::1p80
- o 2001:db8::1#80

The situation is not much different in IPv4, but the most ambiguous case with IPv6 is the second bullet. This is due to the "::"usage in IPv6 addresses. This style is NOT RECOMMENDED for its ambiguity. The [] style as expressed in [\[RFC3986\]](#) SHOULD be employed, and is the default unless otherwise specified. Other styles are acceptable when there is exactly one style for the given context and cross-platform portability does not become an issue. For URIs, [\[RFC3986\]](#) MUST be

followed.

## [7.](#) Prefix Representation

Problems with prefixes are just the same as problems encountered with addresses. Text representation method of IPv6 prefixes should be no different from that of IPv6 addresses.

## [8.](#) Security Considerations

This document notes on some examples where IPv6 addresses are compared in text format. The example on [Section 3.2.5](#) is one that may cause a security risk if used for access control. The common practice of comparing X.509 data is done in binary format.

## [9.](#) IANA Considerations

None.

## [10.](#) Acknowledgements

The authors would like to thank Jan Zorz, Randy Bush, Yuichi Minami, Toshimitsu Matsuura for their generous and helpful comments in kick starting this document. We also would like to thank Brian Carpenter, Akira Kato, Juergen Schoenwaelder, Antonio Querubin, Dave Thaler, Brian Haley, Suresh Krishnan, Jerry Huang, Roman Donchenko, Heikki Vatiainen, Dan Wing for their input. Also a very special thanks to Ron Bonica, Fred Baker, Brian Haberman, Robert Hinden, Jari Arkko, and Kurt Lindqvist for their support in bringing this document to the light of IETF working groups.

## 11. References

### 11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", [RFC 4291](#), February 2006.

### 11.2. Informative References

- [I-D.ietf-behave-address-format]  
Huitema, C., Bao, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", [draft-ietf-behave-address-format-04](#) (work in progress), January 2010.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), January 2005.
- [RFC4038] Shin, M-K., Hong, Y-G., Hagino, J., Savola, P., and E. Castro, "Application Aspects of IPv6 Transition", [RFC 4038](#), March 2005.
- [RFC5214] Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", [RFC 5214](#),

March 2008.

## [Appendix A](#). For Developers

We recommend that developers use display routines that conform to these rules. For example, the usage of `getnameinfo()` with flags argument `NI_NUMERICHOST` in FreeBSD 7.0 will give a conforming output, except for the special addresses notes in [Section 5](#). The function `inet_ntop()` of FreeBSD 7.0 is a good C code reference, but should not be called directly. See [[RFC4038](#)] for details.

### Authors' Addresses

Seiichi Kawamura  
NEC BIGLOBE, Ltd.  
14-22, Shibaura 4-chome  
Minatoku, Tokyo 108-8558  
JAPAN

Phone: +81 3 3798 6085  
Email: kawamucho@mesh.ad.jp

Masanobu Kawashima  
NEC AccessTechnica, Ltd.  
800, Shimomata  
Kakegawa-shi, Shizuoka 436-8501  
JAPAN

Phone: +81 537 23 9655  
Email: kawashimam@necat.nec.co.jp

