Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: October 23, 2009

S. Kawamura
NEC BIGLOBE, Ltd.
M. Kawashima
NEC AccessTechnica, Ltd.

April 21, 2009

A Recommendation for IPv6 Address Text Representation
draft-kawamura-ipv6-text-representation-01

Status of this Memo

Copyright Notice

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 [RFC4291] 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 customers. The following draft will describe the problems that a flexible text representation has been causing. This document also recommends a canonical representation format that best avoids confusion.

Table of Contents

1. Introduction 4
  1.1. Requirements Language 4
2. Text representation flexibility of RFC4291 4
  2.1. leading zeros 4
  2.2. zero compression 5
  2.3. Uppercase or Lowercase 6
3. Problems Encountered with the Flexible Model 6
  3.1. Searching 6
    3.1.1. General Summary 6
    3.1.2. Searching Spreadsheets and Text Files 6
    3.1.3. Searching with Whois 7
    3.1.4. Searching for an Address in a Network Diagram 7
  3.2. Parsing and Modifying 7
    3.2.1. General Summary 7
    3.2.2. Logging 7
    3.2.3. Auditing. Case 1 8
    3.2.4. Auditing. Case 2 8
    3.2.5. Unexpected Modifying 8
  3.3. Operating 8
    3.3.1. General Summary 8
    3.3.2. Customer Calls 9
    3.3.3. Abuse 9
  3.4. Other Minor Problems 9
    3.4.1. Changing Platforms 9
    3.4.2. Preference in Documentation 9
    3.4.3. Legibility 9
4. A Recommendation for IPv6 Text Representation 10
  4.1. Handling Leading Zeros 10
  4.2. "::" usage 10
    4.2.1. shorten as much as possible 10
    4.2.2. one 16bit 0 field 10
    4.2.3. when "::" can be used twice 10
  4.3. Lower Case 10
5. Conclusion 10

6. Security Considerations 11
7. IANA Considerations 11
8. Acknowledgements 11
9. References 11
9.1. Normative References 11
9.2. Informative References 11
Appendix A. IPv6 Addresses with Embedded IPv4 Addresses 12
Appendix B. For developers 12
Appendix C. Prefix Issues 12
Appendix D. Phonetic Alphabet and Figure Code 12
Authors' Addresses 13

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 point to the same IPv6 address. This flexiblity 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 RFC 2119 [RFC2119].

2. Text representation flexibility of RFC4291

Examples of flexibility in Section 2.2 of RFC4291 are described below.

2.1. leading zeros

> '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 16bit field is different, but all these addresses are 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 16bits of zeros.

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

In case where there are 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

... and more

In addition, RFC4291 in section 2.2 notes,

'The "::" can also be used to compress leading or trailing zeros in an address.'

It is possible to choose whether to compress a leading zero or a trailing zero in a single address. 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

... more combinations

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 reagular 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 the whois database, one will likely check the output to see if the entry is correct. If a entity was recorded as 2001: db8::/48, but the whois ouput 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 contain IP addresses of systems. 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. This is not too much a problem if the output is to be just 'read' or 'managed' by a network engineer. However, 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. It must be noted that each additional line of a program will result in increased development fees that will be charged to the customers.

3.2.2. Logging

If an application were to ouput 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. This will result in additional code on the

applications which will result in extra fees charged to the customers. 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 2001:0db8:0000:0000:0000:0000:0000:0001 just because the new engineer on the block felt it was better, a simple diff will tell you 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 diffs that do not take into account address representation rules.

3.2.4. Auditing. Case 2

Node configurations will be matched against a 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. An SNMP GET of an interface address and text representation in a humanly written text file is highly unlikely to match on first try.

3.2.5. 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 (which is seen in some RIR databases). If the zeros were inputed 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. A distinguished engineer will know that the right address is set, but an operator, or a customer that is communicating with the operator to solve a problem, is usually not as distinguished as we would like. Again, the extra load in checking that the IP address is the same as was intended, will result in fees

that will be charged to the customers.

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 NOC 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, but flexibility in address representation will increase the work load which will again, result in fees that will be charged to the customers, and also longer down time of systems.

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 RFC 4291, is implemented by various operating systems, and is human friendly.

4.1. Handling Leading Zeros

Leading zeros should be chopped for human legibility and easier searching. Also, a single 16 bit 0000 field should be represented as just 0. Place holder zeros are often cause of mis-reading.

4.2. "::" usage

4.2.1. shorten as much as possible

The use of "::" should be used to its maximum capability (i.e. 2001:db8::0:1 is not very clean).

4.2.2. one 16bit 0 field

"::" should not be used to shorten just one 16bit 0 field for it would tend to mislead that there are more than one 16 bit field that is shortened.

4.2.3. when "::" can be used twice

When cases where it is possible to use "::" in two or more different sections of an address, implementation to shorten the side with more 16bit 0 fields are more common (i.e. latter is shortened in 2001:0:0:1:0:0:0:1). When the length of 16bit 0 fields are equal (i.e. 2001:db8:0:0:1:0:0:1), the former is usually shortened. One idea to avoid any confusion, is for the operator to not use 16bit field 0 in the first 64 bits. By nature IPv6 addresses are usually assigned or allocated to end-users as longer than 32 bits (typicaly 48bit or longer).

4.3. Lower Case

Recent implementations tend to represent IPv6 address as lower case. It is better to use lower case to avoid problems such as described in section 3.3.3 and 3.4.3.

5. Conclusion

The recommended format of text representing an IPv6 address is

summarized as follows.

(1) omit leading zeros

(2) "::" used to their maximum extent whenever possible

(3) "::" used where shortens address the most

(4) "::" used in the former part in case of a tie breaker

(5) do not shorten one 16bit 0 field

(6) use lower case

Hints for developers are written in the Appendix section.

6. Security Considerations

None.

7. IANA Considerations

None.

8. Acknowledgements

The authors would like to thank Jan Zorz, Randy Bush, Yuichi Minami, Toshimitsu Matsuura for their generous and helpful comments.

9. References

9.1. Normative References

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

9.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, February 2003.

[RFC4038] Shin, M-K., Hong, Y-G., Hagino, J., Savola, P., and E. Castro, "Application Aspects of IPv6 Transition", RFC 4038, March 2005.

[RFC5156] Blanchet, M., "Special-Use IPv6 Addresses", RFC 5156, April 2008.

Appendix A. IPv6 Addresses with Embedded IPv4 Addresses

IPv4-Compatible IPv6 address and IPv4-Mapped IPv6 address are defined that carry an IPv4 address in the low-order 32 bits of the address. These addresses have special representation that combine hexadecimal and decimal notations. IPv4-Compatible IPv6 address is deprecated. Although the use of IPv4-Mapped IPv6 address is not recommended due to security and portability problems, the text representation method noted in this document should be applied for the hexadecimal part.

Appendix B. 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. The function inet_ntop() of FreeBSD7.0 is a good C code reference,

but should not be called directly. See RFC4038 for details.

Appendix C. Prefix Issues

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.

Appendix D. Phonetic Alphabet and Figure Code

The use of Phonetics Alphabet is essential for complete accuracy in voice communications. For example, ITU Phonetic alphabet and Figure Code is as follows (extracted hexadecimal from it):

Character	Word	Pronunciation
0	Nadazero	NAH-DAH-ZAY-ROH
1	Unaone	OO-NAH-WUN
2	Bissotwo	BEES-SOH-TOO
3	Terrathree	TAY-RAH-TREE
4	Kartefour	KAR-TAY-FOWER
5	Pantafive	PAN-TAH-FIVE
6	Soxisix	SOK-SEE-SIX
7	Setteseven	SAY-TAY-SEVEN
8	Oktoeight	OK-TOH-AIT
9	Novenine	NO-VAY-NINER
A	Alfa	AL FAH
B	Bravo	BRAH VOH
C	Charlie	CHAR LEE or SHAR LEE
D	Delta	DELL TAH
E	Echo	ECK OH

F	Foxtrot	FOKS TROT

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