Journal of Cloud Computing
a SpringerOpen Journal

RESEARCH Open Access

# Dot-base62x: building a compact and user-friendly text representation scheme of ipv6 addresses for cloud computing

Zhenxing Liu*, Lu Liu*, James Hardy, Ashiq Anjum, Richard Hill and Nick Antonopoulos

* Correspondence: z.liu3@unimail.derby.ac.uk; l.liu@derby.ac.uk
Distributed and Intelligent Systems Research Group, School of Computing and Mathematics, University of Derby, Derbyshire, United Kingdom

**Abstract**

Cloud computing has dramatically reshaped the whole IT industry in recent years. With the transition from IPv4 to IPv6, services running in Cloud computing will face problems associated with IPv6 addressing: the notation is too long (39 bytes), there are too many variants of a single IPv6 address and a potential conflict may exist with conventional http_URL notation caused by the use of the colon (:). This paper proposes a new scheme to represent an IPv6 address with a shorter, more compact notation (27 bytes), without variants or conflicts with http_URL. The proposal is known as dot-base62x as it is an IPv6 address with Base62x and uses the well-known period (or dot) as a group delimiter instead of the colon. The relative merits and demerits of other works that predate this paper have been reviewed and critically evaluated. Cloud computing, as a continuously emerging mainstream of network-based applications, is likely to be a forerunner in the use of IPv6 as the base protocol. As a result, Cloud computing will benefit most from the new, compact and user-friendly textual representation of IPv6 address proposed by this paper.

**Keywords:** IPv6 address, Cloud computing, Base62x, Colon hexadecimal, Text Encoding/Decoding

## Introduction

Cloud computing paradigm has emerged as an energy- efficient, fault-tolerant and on-demand approach which enables ubiquitous network accesses to a shared pool of flexibly reconfigurable computing resources. Networks, servers, storage, applications and services can be rapidly deployed with minimal management input or service provider interaction [1]. It is also marketed as a fast, low cost method for small and medium-size business to setup an online presence.

Cloud computing relies on the infrastructure of Internet; as a consequence, it will be significantly affected by the transition from current IPv4 to next generation IPv6. It is notable that all cloud computing service modes, e.g. SaaS, PaaS and IaaS, are made possible with the underlying support of TCP/IP. Without a reliable, efficient networking infrastructure it is unlikely that cloud computing would be able to develop. It is anticipated that there will be a protracted period of change and that dual-stack IP networking will be utilised for a considerable time.

The reasons why IPv6 is necessary and how the new IP scheme copes with the increasing demand from IT industry can be read from Davis' book [2] and other resources [3,4]. One of the most distinct motivators for change is the depletion of IPv4 addresses, i.e. current IPv4 *Class A* address ranges have been fully allocated, restricting the availability of IP addresses for new Internet users and services. The recent exponential growth of the Internet and the impending exhaustion of the IPv4 address space is the biggest one of major problems. Although the 32-bit address space of IPv4 permits over 4.2 billion addresses, previous and current allocation practices have limited the number of publicly useable IPv4 addresses to a few hundred million. This practice, combined with the rapid expansion of numbers of internetworked devices, has resulted in public IPv4 addresses becoming relatively scarce, forcing many users and organizations to use a NAT to map a single public IPv4 address to multiple private IPv4 addresses. Figure 1 shows a prediction of the rate of IPv4 address pool depletion based on current usage.

The demand for new IP addresses is continuously increasing and it is speculated that after the depletion of IPv4, there will be a very high number of address and/or ports translated networks, which will be highly inefficient and very likely to be inconsistently applied.

A secondary motivator for transition has previously been that IPv6 may provide significant technical advantages over IPv4. Cloud computing may be able to satisfy the increasing demands for real-time interaction, peer to peer services, secure communication and complex network management but it seems unlikely that IPv4 networks will be able to fully meet the necessary transport criteria. The advantages offered by IPv6 will become a necessity for cloud computing to develop as fully and as rapidly as possible.

In order to address the limitations of IPv4, the Internet Engineering Task Force (IETF) developed the IP version 6 (IPv6) suite of protocols and standards. When compared with IPv4, the advantages of IPv6 fall into two categories:

✓ *Changes that address fundamental inadequacies of IPv4.*

- Larger address space. Probably the most commonly known advantage of IPv6 over IPv4 is its enlarged address space. While IPv4 address is 32-bit long,

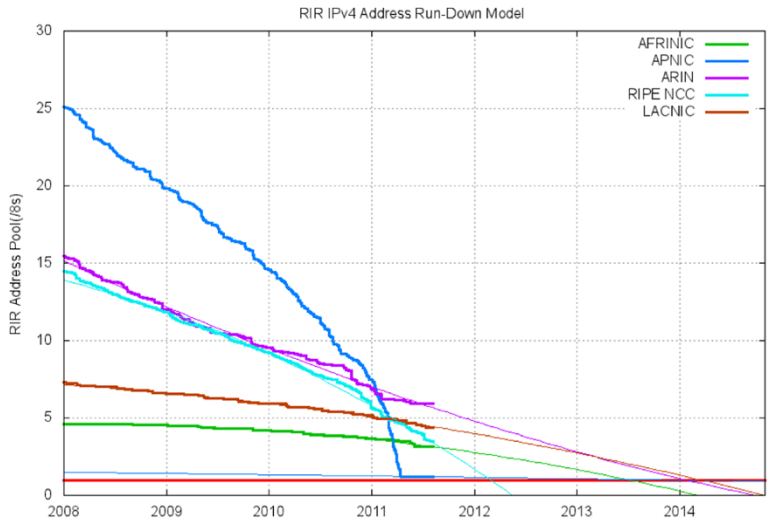


**Figure 1** Regional IPv4 address consumption [3].

IPv6 uses a 128-bit address, the four fold increase in length results in approximately $8\times10^{28}$ more addresses being available and resolving the public address depletion issues.

• Depreciation of NAT. The address space provided by IPv6 removes the need to connect multiple machines to the Internet using a single address and network address translation (NAT). NAT is not included in the IPv6 suite. Without NAT, direct peer-to-peer communication is possible, allowing machines to connect to each other without intermediate "broker" services, like mail exchangers/relays, web proxies, DNS forwarders or SIP gatekeepers, which are run by a service provider.

• Stateless and Stateful Address Configuration. The large address space allows for a simplified address configuration mechanism, providing a service similar to the dynamic host configuration protocol (DHCP) but avoiding the need to maintain state information about address leases.

Other modifications in this group also include removal of broadcasting, enhancements to multicast and streamlined routing tables which together improve performance, manageability and flexibility.

✓ *Advanced features introduced with IPv6.* e.g.

• Network built-in security. The current standards bring a full implementation of IPv6 to include network layer encryption and authentication using IPsec as a mandatory function. Among other advantages of fully integrated network traffic encryption this provides the means to encrypt traffic even within a local network, thus providing protection from insiders trying to sniff network traffic.

• Mobile IPv6. The IPv6 standards include a feature called "Mobile IPv6". This allows "roaming" while maintaining a "home" network address at all times, keeping all existing network connections open even while the underlying network connectivity changes.

• Improved Quality-of-Service (QoS). A new capability is added in IPv6 to enable the labelling of packets belonging to particular traffic flows for which the sender requests special handling. This is done through the "flow label" component in the IPv6 header [5]. This feature is particularly pertinent in real-time services, such as in video-conferencing. Flow-label allows all packets in an IPv6 flow to packet to be routed in a consistent manner to reduce jitter and prevent packets arriving out of sequence.

Whilst we believe that IPv6 will begin a new and improved communications era for the whole IT industry, we also accept that IPv6 itself is not perfect.

First, it is obvious that with such a large address space ($3.4\times10^{38}$ or 340 undecillion addresses) a significant number of characters will be required to represent any single address. A full IPv6 address consists of 32 bytes or a string of 39 characters (including 7 delimiters) in human readable form which is both challenging to remember and prone to mistakes when read, written or deployed. A longer notation means more buffer space is needed when saving, there is an increased cost in bandwidth and

latency time during transit, and more computing power is used when reading/writing, searching/parsing, etc.

Second, the current IPv6 notation of "colon hexadecimal" [5] has another issue that there are too many variants of text representation for a single IPv6 address [6]. With such a degree of flexibility in representing an address, it might become prone to misinterpretation in both human and computer environments (searching, parsing and modifying, logging and operating). As an example, this IPv6 address:

2001 : 0db8 : 0000 : 0000 : 0001 : 0000 : 0000 : 0001

can be presented in at least eight different forms by using all of the published and accepted compression and omission methods.

Third, the use of the colon (:) separator in place of the dot (.) presents both a potential ambiguity with current http_URL/Windows UNC and the annoyance of being a "two-key" entry on most. It is unpredictable that how many systems and applications will be affected by this incompatibility.

Bearing these issues in mind and considering the increasing demands of cloud computing, this paper introduces a novel scheme to present an IPv6 address in Base62x with period (or "dot") delimiters as used in IPv4. A particular relevance to cloud computing is to provide a consistent and readily manageable approach as early as possible to prevent ambiguity and repeated effort at a future date, the alternative being to continue on with a less useable format. This scheme will overcome the highlighted issues and offer other benefits after its implementation. This is the key finding of the study.

The remaining sections of this paper are organized as follows. Section 2 is a literature review of other works that relate to the issues identified above. Section 3 gives a brief introduction to Base62x notation including its definition, algorithm and usage. Then in section 4 the new scheme, dot-base62x is presented and explained in detail with experiments and analysis. Section 5 re-iterates some of the benefits with dot-base62x notation. A conclusion of the proposal is given in section 6 and there are suggestions for future consideration in section 7.

## Literature reviews

It is accepted that this paper is not the first to make critical comment on current IPv6 text representation and raise issues as described in section 1. It is also a near certainty that this paper will not the last until those issues have been solved in a better and more acceptable way. In other works technicians and engineers have expressed their opinions about the current IPv6 notation with words like "pretty long" [7] "a bit unwieldy" [4], "ugly", "untidy", "awkward" and "difficult to comprehend and work with".

Since the current IPv6 address scheme was introduced by IETF RFC 1884 in 1995 [8], some work has been done to address current IPv6 text representation issues related to excessive length, too many appearance/forms of representation and the potential ambiguity with existing http_URL/Windows UNC. Here are some representative examples to be discussed in detail as below.

### Papers aiming to shorten IPv6 address notation

#### *Elz's informational RFC 1924*

RFC 1924 [9] invented a method to present an IPv6 address in base85. The base85 system consists of the following characters list in an ascending order:

$$\prime 0\prime..\prime 9\prime,\prime A\prime..\prime Z\prime,\prime a\prime..\prime z\prime,\prime !\prime,\prime \prime,\prime \prime,\prime \prime,\prime \%\prime,\prime \&\prime,\prime (\prime,\prime )\prime,\prime *\prime,\prime +\prime,\prime -\prime,\prime ;\prime,\prime <\prime,\prime =\prime,\prime >\prime,\prime ?\prime,\prime @\prime,\prime \prime,\prime \_\prime,\prime `\prime,\prime \{\prime,\prime |\prime,\prime \}\prime, \text{and} \prime \sim \prime.$$

Base85 uses this character set to express any numerical value, including IPv6 addresses. As an example of use, a standard RFC 1884/4291 format IPv6 address of

$$1080:0:0:0:8:800:200C:417A,$$

is translated to a base85 representation of

$$4)+k\,\&C\#VzJ4br > 0wv\%Yp.$$

The encoded string is clearly much shorter than the original one, but this is the only clearly apparent benefit. Primarily, it is an order of magnitude harder to read, use and understand. It also necessitates the user to learn a whole new alphabet. Finally, 85 is not a "bit-boundary" number, base85 therefore does not fully utilize all of the required 7 bits and as a result will produce over-length and discontiguous binary strings.

Due partly to the unusual format of "base 85" to express IPv6 address, the suitability of RFC1924 has been debated many times. It does however show that a new direction for achieving a shorter notation for IPv6 address was recognized very early into its development and it explains the logic process of its author who was trying to cope with the issues born with RFC 1884.

To a lesser degree, the author had found it was necessary to make some changes to RFC 1884 before RFC1924 was published in 1996 and had carried on his thought to a complete scheme.

#### *Translucent implementation in traditional base 64*

Parwez [10] proposed "another brave idea to present a translucent representation of currently implemented IPv6 address with a more compact and end-user friendly format for IT professionals especially for naïve users in networking environments."

Simply it can be said that presenting IPv6 address in base 64, the transformation goes under rules:

> *"The character set to encode the base64 IPv6 address is: 0–9, a to z, A to Z, . (dot) and—(hyphen); Case sensitive IP scheming; each character represents 6-bits; Last character has to be among 0–3; Maximum number of characters are 22 or more precisely 21.33 characters; . . ."*

Some examples of IPv6 address taken directly from this paper are listed below:

NUML.EDU.PK.ISB − 10.10.20.30
encyclopedia.com.US − 02
IT − − − − − − − − − − − − − − − − − − − − −.1
IT + + + − − −.1
IEEE − AaBbCcDdEeFfGgHh3
:: 1

Although these "addresses" are in a format that is unfamiliar, the reduced character set makes them appear far less daunting than the Base85 example given earlier.

The base64 scheme attempts to solve the address length issue by introducing more symbols in a similar manner to Base85 scheme. Both schemes shorten the address representation but in doing so they sacrifice readability and manageability.

It is difficult to suggest that this scheme would be readily accepted by academia or industry as it could introduce more complication than the original RFC 1884/4291.

### A recommendation for unifying all different variants

As noted in the first section of this paper, with at least eight formats there are too many legitimate ways to represent the same single IPv6 address in current colon hexadecimal notation. Since IPv6 addresses are not just used in IP header as binary mode numbers, the high number of variants is likely to cause unpredictable problems when handling literal IPv6 addresses as necessary in different computer systems and applications.

To avoid this confusion, Kawamura proposed IETF RFC 5952 to achieve the goal that any single IPv6 address should have only one textual representation. The suggestions for unifying all variants include methods for handling leading zeros in a 16-Bit field, use of "::" and lowercase. This is a positive attempt to avoid confusion by removing the misleading and mismatching among many similar forms of a single IPv6 address.

Since many possible methods had been provided, confusion could arise due to individual systems, applications and manufacturers adopting the method which they favoured most for their own purposes. The RFC proposes to avoid this situation by permitting only one shortened IPv6 address format. Addresses represented in any other way are not legitimate and therefore not permitted.

### Proposals to resolve colon-related conflicts

#### *Extra square brackets in domain part of http_URL*

IETF RFC 2732 [11], "Format for Literal IPv6 Addresses in URL's" was created to address the colon-related issue. RFC 2732 narrates the situation where why this is necessary and how to handle it with one more pair of square brackets.

> *"The textual representation defined for literal IPv6 addresses in [ARCH] is not directly compatible with URL's. Both use ":" and "." characters as delimiters. This document defines the format for literal IPv6 Addresses in URL's for implementation in World Wide Web browsers. The goal is to have a format that allows easy "cut" and "paste" operations with a minimum of editing of the literal address."*

This proposal introduces further characters into IPv6 URLs in the form of square bracket to distinguish between the different meanings of the colon character. Examples of URLs which employ this format are

http://[2001:db8:0000:0:1::1]:8080/file/to/path?query.
http://[FEDC:BA98:7654:3210:FEDC:BA98:7654:3210]:80/index.html
http://[1080:0:0:0:8:800:200C:417A]/index.html
http://[3ffe:2a00:100:7031::1]

#### *Microsoft changes to overcome unc violation*

In Microsoft Windows operating systems, IPv4 addresses are valid location identifiers in Uniform Naming Convention (UNC) path names, e.g.

\\127.0.0.1\C$.

In a UNC path name the colon is an illegal character which means that the use of colon separated IPv6 addresses are also illegal in UNC names. For this reason, Microsoft implemented a transcription algorithm to represent an IPv6 address in form of a domain name that can be used in UNC paths. To achieve this, Microsoft registered and reserved the second-level domain "ipv6-literal.net" on the Internet. IPv6 addresses are transcribed as a hostname or subdomain name within this name space, in the following fashion

2001 : db8 : 85a3 : 8d3 : 1319 : 8a2e : 370 : 7348

is written as

2001 − db8 − 85a3 − 8d3 − 1319 − 8a2e − 370 − 7348.ipv6 − literal.net.

This notation is automatically resolved by Microsoft software without any queries to DNS name servers. If the IPv6 address contains a zone index, it is appended to the address portion following an 's' character as

fe80 − −1s4.ipv6 − literal.net.

As proposed, this method involves more cost and greater complexity, which means it is not the best choice.

It is anticipated that colon-related issues in IPv6 address notation will not be limited to only the http_URL and Windows UNC examples that are presented here.

There is a very clear requirement to resolve all of these problems preferably in a single method and without introducing further compound confusion.

### On-going studies on these issues

The well-known and well documented problem of depleted IPv4 addresses means that globally IPv6 addressing is continually attracting more attention than ever before. Most of the papers, publicity, guidance and training, though not directly attempting to resolve addressing issues, will encourage technicians and engineers to challenge, change or accept IPv6. IPv6 is no longer a *future* problem and decisions currently being made are likely to impact networks, communications and the Internet for many years to come. Figure 2 shows the trends of "IPv6 address" in Google's search engine.

Cloud computing will generate a requirement for even more address space, much of which will be IPv6 by choice to take advantage of the technical improvements or out of simple necessity because IPv4 are not available. Presently only approximately 0.2% of Internet addresses are IPv6-based [13]. Although this is a significant number in real terms, it is still considered feasible to modify the IPv6 address notation prior to mass migration. VeriSign observed a fourfold increase in IPv6 traffic over its infrastructure in 2010, this level of change suggests the urgency for Internet community to accept an alternative to the benefit of the whole IT industry.

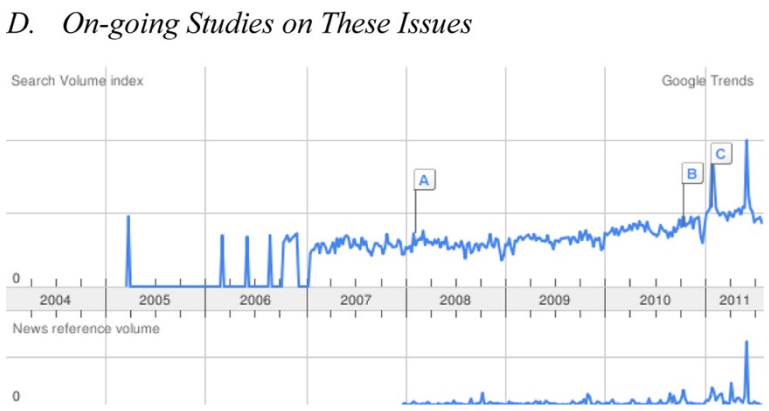


**Figure 2 "IPv6 address" in Google Trends [12].**

## Base62x

In IETF RFC 4648 [14], its authors summarized that base encoding "*is used in many situations to store or transfer data in environments that, perhaps for legacy reasons are restricted to US-ASCII (Cerf, 1969) data. Base encoding can also be used in new applications that do not have legacy restrictions, simply because it makes it possible to manipulate objects with text editors*".

Many base encoding schemes have been invented over time for a multitude of different purposes. Within these schemes discrepancies are occasionally noted as the scheme is applied to an increased level of use and scrutiny. As a result, RFC 4648 states "*The purpose of this specification is to establish common alphabet and encoding considerations. This will hopefully reduce ambiguity in other documents, leading to better interoperability*".

Base62x is a piece of work in the field of base encoding which strives to overcome some issues with conventional base 64 system.

As discussed in RFC 4648, traditional base 64 needs three more symbols ("+", "/", "=") to organize its algorithm and representation. As noted by the RFC, this could be problematic in some scenarios:

- Encoded into structures that mandate other requirements. For base 16 and base 32, this determines the use of upper- or lowercase alphabets. For base 64, the non-alphanumeric characters (in particular, "/") may be problematic in file names and URLs.
- Used as identifiers. Certain characters, notably "+" and "/" in the base 64 alphabet, are treated as word-breaks by legacy text search/index tools.

There are nine other groups of variants [15] substituted into traditional base 64 as an attempt to resolve the problems introduced by these otherwise controversial symbols.

Base62x was first described in a paper [16] where it is proposed as an alternative approach to Base64 for non-alphanumeric characters and is intended to be an improved implementation of Base64. It does not use any symbols in its representation, only case sensitive alphabetical (a-z, A-Z) and numeric characters (0–9).

Some of the differences between the original Base64 and Base62x can be seen in Table 1. In the new scheme, the symbols "+", "/" and "=" are not used. Instead, the character "x" (or any other one amongst the group of 0–9, a-z and A-Z) is a special tag and subsequently x1 represents number 61, x2 for 62 and x3 for 63. As a result, the new alphabet series is 0–9, A-Z, a-z (excluding x) and x1, x2, x3.

Table 2 is taken directly from the original paper and contains some examples of string encoded in Base62x .

Since there is no symbol used in Base62x index table, it shortens the length of IPv6 address without adversely affecting readability, one of the important requirements of the proposed IPv6 address notation.

## Dot-base62x notation of ipv6 address

### Dot-base62x definitions

Theory of number base has been explained in a great detailed means in Knuth's book [17]. Number bases are also called positional numeric systems (Figure 3).

Knuth's book [17] shows that as $b$ is incremented the number notation will be shorter and compact, and at the same time, the set of $a$ will get larger.

It is also obvious that a large base number should be used if a shorter is desired. This answers the question why the papers discussed in previous sections employed base 85 or base 64 to shorten the representation of IPv6 addresses. For an IP address, base 16 is the closest option next to base 10, afterwards there are base 32, base 64 and base 85 in Elz's informal RFC paper.

This proposed new scheme of IPv6 address notation presented by this paper is called *dot-base62x*. The binary and colon-hexadecimal representations of an IPv6 address are shown in Figure 4.

This long address is commonly depicted as eight pairs of bytes, but it can also be considered in three sections as shown in Figure 5.

The first half of the address is a 64-bit subnet prefix comprising of a six byte (48 bits) Global Routing Prefix and a two byte (16 bits) Subnet ID. The second part of the address is another 64 bits known as the Interface ID and is used mainly in a unicast addressing. For the purpose of this paper, IP6 addressing could be described using the format:

$$\text{yyy.yyy.yy.yyy.yyy.yy}\ (3.3.2.3.3.2),$$

**Table 1 Codes index comparisons**

| *Base62x* | | | | *Base64* | | | |
|---|---|---|---|---|---|---|---|
| *Val* | *Enc* | *Val* | *Enc* | *Val* | *Enc* | *Val* | *Enc* |
| 0 | 0 | . | | 0 | A | . | |
| 1 | 1 | . | | 1 | B | . | |
| 2 | 2 | . | | 2 | C | . | |
| 3 | 3 | 60 | z | 3 | D | 60 | 8 |
| 4 | 4 | 61 | x1 | 4 | E | 61 | 9 |
| . | | 62 | *x2* | . | | 62 | + |
| . | | 63 | x3 | . | | 63 | / |
| . | | (tag) | x | . | | (pad) | = |

**Table 2 Examples of Base62x**

| No. | Original text | Encoded text in Base62x |
|---|---|---|
| 1 | aBC | OK93 |
| 2 | A__B* | GLx1VGYe |
| 3 | COMPSAC 2011 | Gqx1DK5D1Go0oC34n |
| 4 | 中文简体 | vBYjvfQ7vww0vBsJ |
| 5 | メインページ | uuEXuuAauuEpuuEQuuEzuuAu |

where each "y" stands for one byte (8 bits or two characters). After encoding into Base62x, "yyy" (3 bytes, 24bits or six hex characters) will be replaced by "xxxx" (four base62x 6-bit characters) and "yy" (2 bytes, 16 bits or four hex characters) will be replaced by "xxx" (three base62x 6-bit characters) in Base62x.

Therefore, using the proposed new notation scheme, an IPv6 address in Basex62x will be in the general format

$$\text{xxxx.xxxx.xxx.xxxx.xxxx.xxx} \quad (4.4.3.4.4.3),$$

where each "x" represents any one six-bit character (using the code scheme 0–9, a-z, A-Z, x1-3). As before, the first half of the address indicates the subnet prefix and the second half indicates the interface ID. The first 3-digit group indicates the subnet ID.

The proposed scheme is known as ***dot-base62x*** notation of IPv6 address and has the following features:

- Encoded in Base62x
- Dot-separated six segments
- Prototype length: 22 codes + 5 dots = 27 characters
- Character range: 0–9, A-Z, a-z
- Case-sensitive

#### Conversions to/from dot-base62x

The process of converting an IPv6 address into dot-base62x can be summarized as these steps:

- S1. Divide the given 16-byte address into 6 segments as 3:3:2:3:3:2
- S2. Convert each segment into Base62x
- S3. Separate the Base62x encoded string into 4:4:3:4:4:3 as xxxx.xxxx.xxx.xxxx.xxxx.xxx

Figure 6 is a graphical representation of the following example. In the middle of the illustration there is a string

00100000000000010000110110111000000000000000000000010111100111011000000101010100000000011111111111111110001010001001110001011010

$$(\cdots a_3 a_2 a_1 a_0 . a_{-1} a_{-2} \cdots)_b =$$
$$\cdots + a_3 b^3 + a_2 b^2 + a_1 b^1 + a_0 + a_{-1} b^{-1} + a_{-2} b^{-2} + \cdots$$

**Figure 3 Positional number systems.**

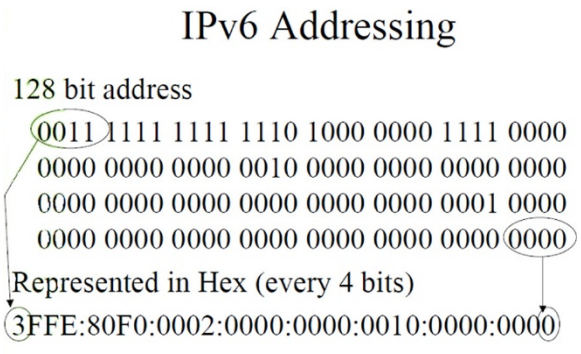


**Figure 4** IPv6 address in base 16.

which is the binary representation of the IPv6 address

2001 : 0DB8 : 0000 : 2F3B : 02AA : 00FF : FE28 : 9C5A.

Firstly, divide the binary string into 6 segments by the proportions of 3:3:2:3:3:2,

001000 000000 000100 001101 (3 bytes, 24 bits)
101110 000000 000000 000000 (3 bytes, 24 bits)
0010 111100 111011 (2 bytes, 16 bits)
000000 101010 101000 000000 (3 bytes, 24 bits)
111111 111111 111000 101000 (3 bytes, 24 bits)
1001 110001 011010 (2 bytes, 16 bits)

Secondly, encode each segment using 6-bit Base62x,

804D k000 2zy 0ge0 x3x3ue 9nQ

Thirdly, add the period (or dot) as a delimiter,

804D.k000.2zy.0ge0.x3x3ue.9nQ

Two more IPv6 addresses conversions have been demonstrated using dot-base62x as below.

Given the IPv6 address in IPv4 style looks as:

128.91.45.157.220.40.0.0.0.252.87.212.200.31.255

Its binary format is:

10000000010110110010110110011101110111000010100000000000000000000000000000000000000000111111000101011111010100110010000001111111111111

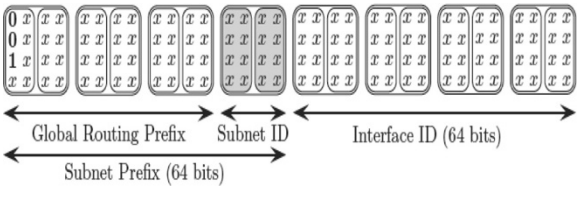


**Figure 5** Globally-routed unicast address format address [4].

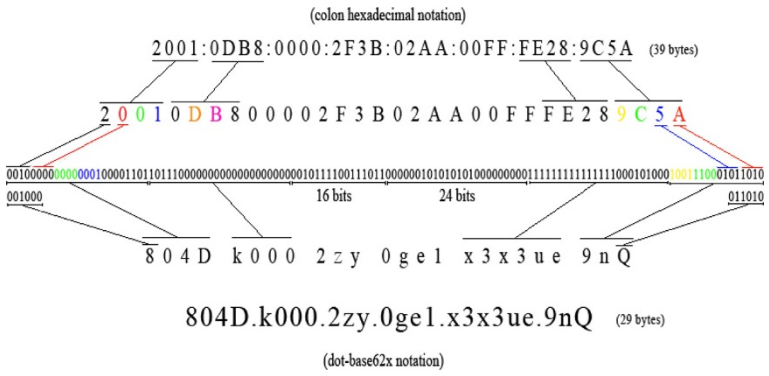


**Figure 6** Logic of dot-base62x notation, compared with colon hexadecimal.

When each segment is converted into dot-base62x format, the following string is the result

W5ij.dTme.000.003z.Lx1J8.1x3x3.

As a final example, an IPv6 address is given as

fe80 : 0000 : 0000 : 0000 : 020c : f1ff : fefd : d2be,

After encoding to dot-base62x, it becomes

x3e00.0000.000.0Wpn.x3x3yx1.DAx2.

Using the steps listed, a conversion program was written to automate the process of converting an IPv6 address from base 16 to Base62x. A screenshot (Figure 7) of the conversion program is shown.

A batch conversion experiment was undertaken to observe the format of the dot-base62x notation in a real world simulation. The minimum and maximum length of each group was recorded and compared after converting different groups of randomly generated IPv6 addresses. The size of the test groups were as $10^2$, $10^3$, $10^4$ and $10^5$ IPv6 addresses.

"Generated randomly" when used here means that real network environments are simulated in so much as any byte of the 16-byte address can be randomly assigned any value ranging from decimal 0–255 or 0x0 to 0xFF. All possible IP addresses in any real world networks have an equally probability of being examined in this simulation. Therefore the addresses randomly generated will have any value the decimal range from zero to $3.4\times10^{38}$ or, as more commonly shown in hexadecimal form, from

0000 : 0000 : 0000 : 0000 : 0000 : 0000 : 0000 : 0000

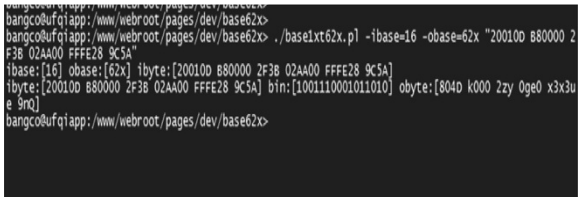


**Figure 7** Dot-base62x conversion from base16.

to the maximum value of

FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF.

In each set of the sample IPv6 addresses, the program converts each address into its corresponding dot-base62x format and records the length and the time taken to convert. On completion of a set, the maximum and minimum values are displayed and the average value calculated for the set. The experiment was repeated with three groups of random addresses for each set size to avoid anomalous or ambiguous results.

The partial output of the test program for one set of $10^2$ address data is shown below.

*No./IPv6 address in base* 16/*Encoded in Base*62*x*/*Length of Base*62*x string*
0/4*A*8*A*2*A*3647*FFC*9*D*04*D*4*BD*3*A*9*A*9713*ADC*/*Ieeg DaVx*3 *CdG JKlJ gQbn* 3*hS*/23
1/3*DD*1762*B*3*A*2*F*8*B*0*BF*3*C*6941*FD*180*F*45*F*/*FT*5*s Apel* 8*iB zzQK* 7*x*160 *FHV*/23
2/*AC*20*B*54*AFB*7*CEA*5*B*3*E*566*ADDE*976*F*3*F*6/*h*22*r Iljz EfR FbPg tUbs FFs*/22
3/6*BA*802*A*9832*A*9*F*58617*CBD*66*E*17*B*0*F*55/*QwW*2 *gOCg* 9*x*1*O ONox*1 *Pk*5*y* 0*x*1*L*/25
4/8*E*87*F*0*CE*81*AE*619*B*3*B*6*ACB*56842*E*574*D*/*ZeVm pe*6*k* 66*R EshB LeGk* 5*TD*/22
96/243007*A*355*A*3*A*4*C*866*BB*5*ECBF*3804271/9307 *erMZ AJ*8 *PhjU ox*3*E*0 49*n*/23
97/402*C*256*FCC*6*F*96*B*5133*D*230575*DB*2851/*G*2*mb Rznl* 9*Qr* 4*pqZ* 1*NNR* 2*XH*/22
98/*C*529*A*3*D*51783*A*7*B*9822*D*1317*E*34867*C*9/*nIcZ rHU*3 *AUv WYqJ* 5*x*2*D*8 6*V*9/23
99/181*D*4*B*2*EEE*21654*FFA*3*EACB*69*C*0*EB*917/61*rB BkuX* 6*LF x*2*Zwi jfmE BaN*/23
*addr.count* : 100 *min.length* : 22 *max.length* : 25 *avg.length* : 22.91 *timecost* : 17.1488

Figure 8 shows the observed maximum, minimum and average dot-base62x address length data as a line graph.

From Figure 8 it is clear that no major difference is observed across the sample ranges. The maximum lengths vary across a small range whilst the average and minimum lengths show practically zero deviation.

It is therefore predictable and conclusive that the average length will not increase with address samples ranging from $10^2$, $10^5$, $10^8$, $10^{20}$ to the largest set of current IPv6 addresses, 2^128, i.e. $3.4 \times 10^{38}$.

### Comparisons between dot-base62x and colon hexadecimal notation of IPv6 address

The differences between current colon hexadecimal and the proposed dot-base62x notation of IPv6 addresses have been listed in Table 3 which summarizes a few aspects of these two forms.

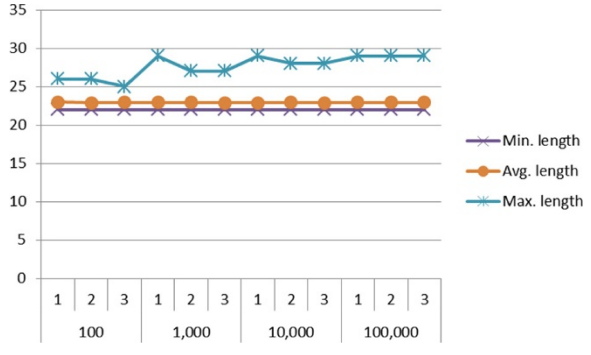


**Figure 8** Trends of the length of IPv6 in dot-base62x.

## Advantages and benefits of dot-base62x

The whole Internet community and especially Cloud computing, which will have a major reliance on IPv6, will benefit from the proposed scheme in the following aspects of IP-related systems and applications.

### Shorter notation

The original objective of this study was to find a shorter textual representation for IPv6 addressing. The length of an IPv6 address encoded in dot-base62x has a theoretical reduction in length of (39–27)/39 = 30.77% when compared to the same address in colon hexadecimal, i.e., from 39 to 27 in bytes. Figure 9 (Data from Table 3) shows the comparisons of lengths of IPv6 encoded in colon hexadecimal and dot-base62x.

It is clear from Figure 9 that dot-base62x has a theoretical length reduction of approximately 30% (orig. column) and approximately 24% in real use (avg. column). In practice, only the average values are significant; the minimum and maximum lengths are largely irrelevant due to the low probability of these extremes in real network environments.

IP addresses do not exist exclusively in the headers of IP packets but are also widely used in many network-based systems and applications. Any decrease in length can translate to saving space on disk, reduced bandwidth in transit and reduced cost of operation. As discussed in RFC 5952, there are many scenarios that utilise the IP address in a literal rather than binary mode. Some of these scenarios could include:

- Searching,
  - Searching Spread sheets and Text Files
  - Searching with WHOIS
  - Searching for an Address in a Network Diagram
- Parsing and Modifying,
  - Logging
  - Auditing

**Table 3 Comparisons of dot-base62x and colon hexadecimal**

| No. | Fields | Colon hexadecimal | Dot-base62x |
|---|---|---|---|
| 1 | Encoding base | Base 16 | Base62x |
| 2 | Separator | Colon (:) | Dot, full stop (.) |
| 3 | Number of separators | 7 | 5 |
| 4 | Segments/groups | 8 | 6 |
| 5 | Format address length | 39 | 27 |
| 6 | Minimum length | 2(::) | 11(0.0.0.0.0.0) |
| 7 | Maximum length | 45 | 47 |
| 8 | Average length | ~37 | ~28 |
| 9 | Bits operation | Each 4 bits | Each 6 bits |
| 10 | Format | xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx | xxxx.xxxx.yxx.xxxx.xxxx.yxx |
| 11 | Example | 2001:DB8:0:2F3B:2AA:FF:FE28:9C5A | W5ij.dTme.0.3z.Lx1J8.1x3x3 |
| 12 | Variants | Multiple forms | Only one |
| 13 | Status | IETF RFC | Newly-invented |

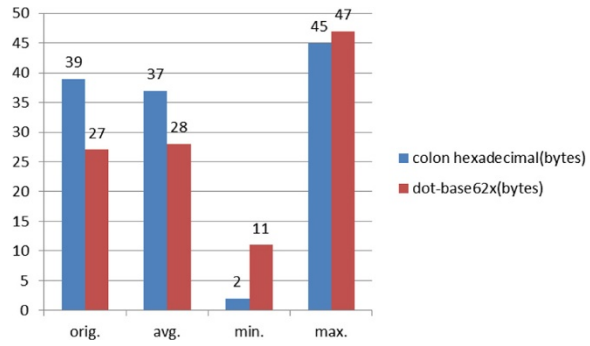


**Figure 9** IPv6 address lengths in dot-base62x and hexadecimal.

- Verification
- Unexpected Modifying

○ Operating,

○ Other Minor Problems.

All of these operations would benefit from reduced disk space and transit time as a consequence of the shortened IP address notation in dot-base62x format. RFC 5952 also discusses the further problem with colon hexadecimal notation in that different variants of presentation can lead to confusion.

#### Compact form, less segments, more human-friendly

The proposed scheme has a more compact form than current colon-hexadecimal notation.

Firstly, instead of the eight groups of characters in colon-hexadecimal, there are only six in dot-base62x. This simpler representation makes the scheme more human-friendly.

Secondly, within the six groups there are always two segments which have only three digits, an additional simplification and readily identified "eye position" marker. Therefore, the regular format of

xxxx.xxxx.yxx.xxxx.xxxx.yxx

is considered less daunting and more human readable than

xxxx : xxxx : xxxx : xxxx : xxxx : xxxx : xxxx : xxxx.

Furthermore, very useful information about the address can be simply read from the 27-byte string, e.g.

– The first half of the address is the globally unique network ID, the second half is interface ID,
– The first "yxx" stands for "subnet ID" for organization in unicast address.

Thirdly, an IPv4 address uses 4 character groups, using the dot-62x scheme an IPv6 address has 6 groups which is a more symmetric and aesthetically pleasing form.

Last but not least, the continuous two-key entry requirement of the colon symbol will very quickly lead to many years of tedium and non-standard keyboard mapping that is resolved by returning to the single keystroke period (or dot) separator.

#### Compatible with IPv4 dot-decimal

It is clear obvious that keeping the identical separator in both versions of IP will maintain consistency in the whole Internet community. People working within the field of networks are already familiar with dot-separated style IP address and will find in more acceptable for the transition from IPv4 to IPv6 if the proposed scheme became widely adopted.

Similar problems with colon-separated notation can be solved by dot-base62x format. A greater degree of compatibility is maintained. This in turn means more existing IPv4 systems and applications can be made to be seamlessly compatible with IPv6 addresses in dot-base62x.

#### Minimized the number of variants

RFC 5952 [6] recommends a unified representation to avoid confusion caused by multiple output forms of colon hexadecimal notation from a single IPv6 address.

The root cause of this problem is that colon hexadecimal simply provides too many methods to represent a single IP address, e.g. case-sensitive or case-insensitive, positions of double colon, whether or when or where compressing leading zeros.

Dot-base62x notation avoids this issue by introducing only one method to compress a given single IPv6 address, the identical method which has been used with IPv4. The method is the natural and intuitive human response of simply suppressing all leading zeros. Therefore, any single IPv6 in dot-basex62 has one and only one textual representation in the same way that an IPv4 is only written in one form.

Confusion in the scenarios listed in the previous section, e.g. searching, logging, parsing and operating, where IPv6 addresses may be used as a textual identifier will be avoided if those IP addresses are encoded in dot-base62x due to the unique format of any single IPv6 address.

#### Avoiding conflict with exist http_URL/Windows UNC

Clearly, with the exception of IPv4 itself, there was no intention for IPv6 address notation to conflict with other existing RFC standards. However, the colon symbol serves as a “port” identifier part in current http_URL scheme, which could lead to confusion between a colon hexadecimal address and http_URL.

The current remedy for this conflict introduces further complication by enclosing the IPv6 address in a pair of square brackets before using it as an IP address in http_URL, e.g., http://[2001:db8:0000:0:1::1]:8080/file/to/path?query. Dot-base62x has no such problem, by abandoning colon in its output form and instead using the “dot” as in IPv4, a greater degree of compatibility is maintained. This in turn means more existing IPv4 systems and applications can be made to be seamlessly compatible with IPv6 addresses in dot-base62x.

The same problem arises with Windows UNC due to the transition from the dot-separated format to the colon-separated format. Dot-base62x will avoid this conflict. These examples can be handled as easily as with IPv4 addresses.

2001:db8:28:3:f98a:5b31:67b7:67ef
➔ \\804D.k00e.3.x2OfR.CMUt.6Vl\
2001:4898:9:3:c069:aa97:fe76:2449
➔ \\8058.c009.3.m6cg.bx3vs.2H9\share

## Conclusion

This research has introduced a compact, user-friendly textual representation of IPv6 addresses.

The Internet has revolutionized human history in recent decades and it will continue to contribute to and reshape the world for many years to come. Cloud computing as the mainstream services of future IT applications will encounter many scenarios where IP addresses are used in plain text representation rather than binary mode. This study reviews the development of current Internet addressing with a primary focus on potential IP evolution.

Literature reviews show that current colon hexadecimal notation of IPv6 address has the following issues when being deployed in cloud computing.

- ✗ Too long. Usually it has 39 characters, sometimes up to 45 characters.
- ✗ Too many variants. A single IPv6 can have several variations in appearance which can cause confusion.
- ✗ Colon (:) conflicts with http_URL/Windows UNC.
- ✗ Incompatible with IPv4.

A new scheme, named as *dot-base62x*, has been proposed by this study as a means to encode IPv6 addresses in Base62x and separate the encoded string with five dots. The proposed dot-base62x has the following advantages and benefits compared with current colon hexadecimal notation.

- ✓ Shorter notation. An IPv6 address length reduction of 30% or so in theory and 24% or so in practice are achieved.
- ✓ Compact shape, less segments, more human-friendly. Instead of eight segments/groups in its output form, dot-base62x only has six segments. Among these six segments, there are two segments with only three digits, providing an even more compact format than the fixed 4-digit with colon hexadecimal.
- ✓ Compatible with IPv4 dot-decimal. More existing IPv4 systems and applications can be made to be seamlessly compatible with IPv6 addresses in dot-base62x.
- ✓ Minimized the number of variants. Dot-base62x notation uses one and only one method to compress a given single IPv6 address, the natural human method which has been used with IPv4.
- ✓ Avoiding conflict with exist http_URL. Dot delimiting is used in dot-base62x. The colon (:) is not used due to conflicts with http_URL which uses the colon to

identify a port. This also resolves the colon conflict problem with Microsoft Windows UNC.

From this paper we have reasonably concluded that current colon hexadecimal notation of IPv6 address is not the best or most suitable choice. We are confident that the proposed dot-base62x representation is a more suitable format and recommended its adoption for an IPv6 address in the coming IT era of Cloud computing.

## Recommendations and future work

With vigorously-developed cloud computing in recent years, more and more services and devices will become cloud based. Every service and device will need at least one IP address. As a consequence, IPv6 is unlikely to be the final addressing scheme used on the Internet. For this proposed scheme itself, there are a few options, recommendations and further works to be done.

### Fixed-width or various length of IPv6 in Base62x

Due to three double-digital characters being added in its index table, the length of strings encoded in dot-base62x may vary in a small range without any other compressing involved. This may be a major concern with dot-base62x when compared with colon hexadecimal.

Taking compressing and better compatibility into consideration, we recommend that keeping its varying length is a better choice from a long term and developmental point of view. As varying lengths are unavoidable in all schemes discussed, it is not considered significant that the length may extend 27 bytes in common use to 47 bytes in very rare extremes.

Lengths of IPv6 address are

$$\begin{array}{ll} 2 - 45 \text{ bytes} & (\text{colon hexadecimal}), \\ 11 - 47 \text{ bytes} & (\text{dot} - \text{base62x}). \end{array}$$

### Ambiguous characters

Dot-base62x uses all the possible alphanumeric characters in its output form, so it is likely that sometimes one of its output forms consists of these potentially ambiguous characters:

"0"(zero) and "o/O"(letter O)
"1"(one) and "l"(letter L, slightly higher than number one)
"2"(two) and "z/Z"(letter Z)

For example,

G2mb.Rznl.9Qr.4pqZ.1NNR.2XH.

In this dot-base62x IPv6 address, it is hard to tell whether it is "1" (one) or "l" (letter L) in the 2nd segment "Rzn?". The same confusion arises from the 5th segment "?NNR". In this font style (Times New Roman), letter "l" is slightly thin and higher than number "1", thus it is letter "l" in segment 2, and number "1" in segment 5.

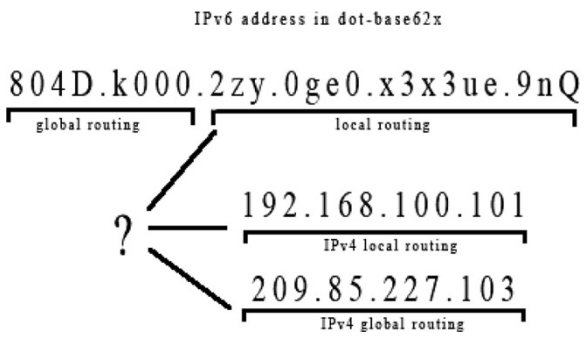


**Figure 10** IPv6 local routing and IPv4.

Fortunately, "z" and "2" has no such misreading problem in such scenario and it is distinguishable that it is letter "z" in segment 2, number "2" in segment 6. But "z" and "2" may be confused with high likelihood when handwritten.

Though such study goes beyond this report, it is necessary to advise a set of hints to write or display these illegible characters if some practical methods have been found in future work addressing this interpretation issue.

### Integration of IPv4 addressing

Dot-base62x uses six "dot separated" groups of characters to fully identify an address (see Figure 10). The last four groups are local subnet and specific device. It may be possible to interpret an IPv4 address as a locally sourced dot-base62x address, expediting the change to IPv6 by minimizing equipment changes, and consequently simplifying the program and greatly reducing the associated costs.

### Next generation: IPv8 or IPv10

It has been said that IPv6 will not be the single final version of IP. The world might need another successor to IPv6, e.g. IPv8 or IPv10. In that case, it may be seen that the length of IPv8 or IPv10 address is even longer than that of current IPv6, e.g. IPv8 may have 18 bytes or 24 bytes, so its forms may look as

FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF
(9 segments),
FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF : FFFF
(12 segments).

However, if dot-base62x is adopted, their notations are relatively compact and back-compatible, these addresses may look as

xxxx.xxxx.xxxx.xxxx.xxxx.xxxx (6 segments),
xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx (8 segments).

Along with dot-base62x this upgrading task becomes easier and more acceptable. We will actively anticipate the standards of IP address scheme after IPv6.

**Competing interests**
The authors declare that they have no competing interests.

**Authors' contributions**
The research presented in this paper is the result of teamwork. All authors read and approved the final manuscript.

**Acknowledgment**
The work reported in this paper has been supported by the RLTF HECloud Programme (Ref: RLTFD005), the Visiting Scholar Foundation of Key Laboratory of Dependable Service Computing in Cyber Physical Society of the Ministry of Education China (CPS-DSC 2012FWJJ01). Major Program of the National Natural Science Foundation of China, under contact No. 90818028 and China 973 Fundamental R&D Program (No. 2011CB302600).

Received: 16 December 2011 Accepted: 15 February 2012
Published: 12 April 2012

**References**

1. Liu Z, Lallie HS, Liu L (2011) "A Hash-based Secure Interface on Plain Connection", Proceedings of CHINACOM'11. ICST.OTG & IEEE Press, Harbin, China.
2. Davies J: Understanding IPv6, Second Edition, Microsoft Press, Redmond, Washington. ISBN-10: 0735624461, 978–0735624467, 2008, pp43-45, 50, 92.
3. Huston G: IPv4 Address Report, daily generated, Retrieved in August 2011, http://www.potaroo.net/tools/ipv4/index.html, 2011.
4. Stockebrand B (2007) IPv6 in Practice—A Unixer's Guide to the Next Generation Internet. Springer, Berlin Heidelberg, ISBN 3-540-24524-3, 978-3-540-24524-7.
5. Hinden R, Deering S: IP Version 6 Addressing Architecture, IETF RFC 4291, 2005, retrieved in August 2011, http://www.rfc.net/rfc4291.html.
6. Kawamura S, Kawashima MA: Recommendation for IPv6 Address Text Representation, IETF RFC, retrieved in August 2011, http://tools.ietf.org/html/rfc5952.
7. Kozierok CM: The TCP/IP guide: a comprehensive, illustrated Internet protocols reference, No Starch Press, Rotterdam Area. ISBN 159327047X, 9781593270476, pp7, 379.
8. Hinden R, Deering S: IP Version 6 Addressing Architecture, IETF RFC 1884, 1995, retrieved in August 2011, http://tools.ietf.org/html/rfc1884.
9. Elz R: "A Compact Representation of IPv6 Addresses", IETF RFC 1924, 1996, retrieved in August 2011, http://tools.ietf.org/html/rfc1924.
10. Parwez R, Haq E et al (2010) **Translucent implementation of ipv6 addressing scheme in communication networks**. International Journal for Infonomics (IJI) 3(3):338–344.
11. Hinden, et al: Format for Literal IPv6 Addresses in URL's. IETF RFC 2732, 1999, retrieved in September 2011, http://www.ietf.org/rfc/rfc2732.txt.
12. Google: Google Trends, retrieved in August 2011, http://www.google.com/trends?q=ipv6+address&ctab=0&geo=all&date=all.
13. Leavitt N: IPv6: Any Closer to Adoption?, Computer, IEEE Computer Society, 2011, Volume 44, Number 9, ISSN 0018–9162, pp15.
14. Josefsson S: The Base16, Base32, and Base64 Data Encodings, IETF RFC 4648, 2006, retrieved in August 2011, http://tools.ietf.org/html/rfc4648.
15. Wikipedia: Base64, 2011, retrieved in November 2011, http://en.wikipedia.org/wiki/Base64.
16. Liu Z, Liu L et al (2011) Base62x: An Alternative Approach to Base64 for non-Alphanumeric Characters, Proceedings of ICNC & FSKD. IEEE Computer Society Press, Shanghai, China.
17. Knuth DE (1997) The Art of Computer Programming Volume 2: Seminumerical Algorithms, Third Edition. Reading. Addison-Wesley, Massachusetts, p 195, ISBN 0-201-89684-2.

doi:10.1186/2192-113X-1-3
**Cite this article as:** Liu *et al.*: Dot-base62x: building a compact and user-friendly text representation scheme of ipv6 addresses for cloud computing. *Journal of Cloud Computing: Advances, Systems and Applications* 2012 **1**:3.