Independent Submission
Request for Comments: 8135
Category: Experimental
ISSN: 2070-1721

M. Danielson
Net Insight AB
M. Nilsson
Besserwisser Networks

1 April 2017

Complex Addressing in IPv6

Abstract

The 128-bit length of IPv6 addresses (RFC 4291) allows for new and innovative address schemes that can adapt to the challenges of today's complex network world. It also allows for new and improved security measures and supports advanced cloud computing challenges.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

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

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8135.

Table of Contents

1. Introduction

This document introduces the fundamental concepts of complex addressing in IPv6, allowing for a wide range of complex addressing schemes to be supported and further developed.

Traditional network addressing schemes such as those used in IPv4 [RFC791] and IPv6 [RFC4291] have been confined to unsigned or integer numbers, representing fixed-point numbers. This has provided natural numbers for early implementations but is not well adapted to the challenges of future networks. Further, these fixed addresses have been proven unsuitable for mobility and virtualization in today's world, where cloud computing defies the traditional fixed addressing model. The increased size of addresses as allowed in IPv6, the significant drop in price of floating-point hardware, and the availability of a well-established floating-point format in IEEE 754 [IEEE754] allow for taking not only the step to floating-point addressing but also the step to complex addressing.

2. 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].

3. Natural Addresses

3.1. Integer Addresses

Traditional addresses are integer addresses and can be expressed in a three-dot format, for example, 113.129.213.11 for the integer 1904334091, a rare IPv4 double-palindromic address. These fixed-point addresses were well adapted to early network usage where each computer on the Internet had a fixed location and thus a fixed address. These addresses are also known as natural addresses. As computers have become more powerful and able to handle larger numbers and thus larger addresses, they have also become more transportable (e.g., laptops and mobile phones). The transportable aspect of computers makes fixed-point addresses moot, as machines can move around rather than be confined to a relatively fixed point.

3.2. Prime Addresses

The prime address (that is, the primary address of a recipient) is an important subclass of integer addresses. Such an address is not divisible by anything but the recipient itself, which means it must be regarded as a unique address. While many prime addresses have been experimentally identified, it has proven to be quite hard to

identify a prime address amongst other addresses without resorting to time-consuming computations. Current use includes security and intelligence, where post boxes are obscured amongst others using large prime addresses.

3.3. Composite Addresses

Composite addresses are formed by two or more prime addresses and thus constitute a shared address, allowing the address to be home for multiple prime addresses. Large composite addresses can be difficult to distinguish from prime addresses, which can be a factor to consider. Composite addresses have also become quite important in addressing new light structures and are used in airplanes to make them lightweight and durable. This is important in connecting to the cloud.

4. Complex Addresses

4.1. Floating Addresses

Floating-point addresses allow for a more flexible addressing scheme better adapted for today’s mobile computers, thus allowing for mobile IP [RFC5944]. Support for floating-point numbers is well established in the form of floating numbers as described in IEEE 754 [IEEE754], which allows both 32-bit and 64-bit floating-point numbers to be represented; this is well matched to the requirements of fitting into a 128-bit IPv6 address.

The use of floating addresses does not, however, imply that devices will be watertight. Please download the watertight app from your app store or distribution server. Also, keep your device well patched, as long-term durability of duct tape is limited, particularly if exposure to salt water is expected. Apply suitable environmentally sound lubricants for best sliding performance.

Duct tape can be used to affix a floating address to a fixed address, such as a physical address. For long-term outdoor adhesion, please use UV-stable, nuclear-grade duct tape in layers: Layer 1 [OSI], the physical layer, for affixing the floating address to the physical address and then final layer, called Layer 7, for the application of UV protection. Intermediate layers can be applied depending on the complexity needed.

4.2. Real Addresses

An important aspect of floating-point addresses is that one needs to establish the real address of a device that has a floating address, such that IP packets can be routed to it through the network. Letting part of the address act as the real floating-point value allows means to express real addresses within this address scheme, thus solving a complex addressing problem.

Real addresses are typically assigned to real estate. Multi-homing is supported when the real estate connects to two or more road networks over individual road interfaces. Each road interface can often handle multiple real addresses. Mobile homes are assigned their current real address.

4.3. Imaginary Addresses

Another important aspect of floating-point addresses is that they can be in several possible locations; thus, one must be able to imagine the address as being somewhere other than where the real address would make you believe. The imaginary address provides this orthogonal property. When the imaginary address is found to be 0, then the imaginary address and the real address are considered equal, and the real address has been found.

Imaginary addresses are important in handling home locations above the normal real estate, that is, for cloud computing. The cloud can be identified using the imaginary address, whose floating address is adapted to a real address as the cloud gently floats by. During windy conditions, this may be difficult to achieve; during network storms, the real address of a cloud can become very unstable. Such storms can occasionally become so strong that they impact real estate and rearrange homes, making the real address quite surreal.

4.4. Flying Addresses

An extension to the imaginary address is the flying address format, which is adapted to the mobility of avian carriers. Avian carriers and their datagrams, as described in [RFC6214], are best addressed with flying addresses, which typically take up ICAO Class G [ICAO-A11] airspace, below the cloud, as can be expected from a lower-layer technology.

4.5. Complex Addresses

With the introduction of the real address and imaginary address parts, the full width of complex addresses can be realized. Both the real and imaginary parts are represented in 64-bit floating-point numbers as described in [IEEE754], thus allowing for the floating-point aspect of addresses. The real address part provides for the real address of a device, whereas the imaginary part allows for the orthogonal addressing of the floating-point address. This allows for complex addressing schemes where both the real and imaginary addresses can be found.

Complex addresses allow for address arithmetic in the usual way but can now go beyond the fixed-point limitations. Adding imaginary parts to the address has not been possible before due to the high cost of early floating-point hardware, which hampered imagination.

5. Supported Addressing Schemes

5.1. Absolute Addresses

It has become increasingly important to establish the absolute address of a device for many purposes, including but not limited to, use by law enforcement. This was manageable with fixed-point addresses but has become increasingly difficult with increased address mobility and floating-point addresses. The complex address scheme provides a method for getting the absolute address by performing the absolute function on the complex address.

5.2. Address Argument

It has become increasingly obvious that there is debate about the address of certain services or functions, leading to address arguments. This is another difficulty with fixed-point addresses, as their one-dimensional form does not allow for an argument to be resolved. The complex addressing scheme provides an elegant solution to these address arguments, as the result of the address argument can trivially be found by taking the argument (i.e., arctan or atan) function of the complex address. Using the appropriate function, full argument resolution can be found without signs of ambiguity.

5.3. Safe Addresses

A safe address is the address of a safe house. This is used in various security scenarios -- the safety lies in that those in need can reach the safe house at the safe address but there is no indication that the address has this role. By use of the imagination, this address can be made less real, simply by making the

imaginary part large enough not to be taken as a real address. Since it is a floating address, the real address can be made 0, thus making it completely imaginary, and the address argument will be orthogonal to any real address, providing it is hard to establish its real address. It is naturally still possible to establish the absolute address when needed.

5.4. Virtual Addresses

Virtual addresses, where the same network interface can have multiple addresses, have traditionally been an important concept. With the complex addressing scheme, the imaginary part allows for a much wider range of virtualization than just normal multiple real addresses for a particular interface. This goes beyond normal cloud computing, where virtualization just allows you to operate somebody else's computer. The new imaginative address capabilities and higher altitude addresses due to the increased range allow you to operate a cloud within a cloud, so that you just run on top of somebody else's cloud. This high altitude allows for supersonic cruise speed for high-performance computing.

5.5. Rational Addresses

Engineers tend to always look at problems rationally, including the problem of addressing. The traditional fixed-point address has, however, only supported a subset of rational addresses, but with the new complex addressing scheme, a larger subset of rational addresses can be reached or approximated, allowing for a larger rationale to be found.

The rationale for this is that with the use of floating addresses, the power of 2 now can perfectly divide. Further, approximations for other dividends can often be sufficiently precise. The full scope of rational numbers has not been reached, however, as the committee was quite imprecise on the use of floating addresses but agreed that this initial support of rational addresses could be acknowledged and helpful while its usage is TBD.

5.6. Irrational Addresses

Support for irrational addresses has been very poor in the traditional addressing scheme, since fixed-point addresses did not support any irrational behavior by design, even if proofs for irrational addresses have been known to be jotted down. The new scheme allows for approximations of irrational addresses to be supported; even though no rationale for why this would be needed could be found, it is a neat feature to handle the irrationality of the world today.

5.7. Transcendent Addresses

As a natural extension to irrational addresses, one can include approximation to the transcendent addresses, which transcend beyond the physical address or even the real address. While only approximated due to limited precision, they can still be used to locate the floating address for the life of Pi [PI], as Pi's life floats by.

6. Geometric Addresses

6.1. Round Addresses

In order to cope with the complexity of the real world, real addresses (both rational or irrational) have always needed to be rounded up for them to be represented. This rounding provides what is known as round addresses and is achieved using a rounding function. This practice is maintained in the complex addressing scheme and is a necessity for support of rational and irrational addresses.

Round addresses are needed to efficiently forward packets around ring-type networks like Token Ring [IEEE-802.5] or Resilient Packet Ring (RPR) [IEEE-802.17].

Common round words include "ring", "circle", and "sphere"; other round words are discouraged, especially when using the network.

6.2. Square Addresses

As is well established, some addresses regularly in use cannot be directly used on the Internet. Addresses in text form are often referred to as square addresses, because the characters traditionally take up a square on the screen and because they act as a square peg in the round hole of Internet addresses. In order to convert these square addresses into round floating-point numbers, the Domain Name Service (DNS) was introduced to replace the host tables.

Host tables are the old-school way of looking up a square number and converting it to round form. Such tables were published for all known square numbers, but they where inherently out of date as new square numbers kept occurring -- new round numbers had to be calculated from these square numbers and then had to be tabulated and published.

Square addresses often use square pi (see Appendix A).

6.3. Polar Addresses

A misconception on square addresses is that they would represent the world as being a flat earth. While the complex addressing scheme supports Cartesian coordinates, alternative polar addresses can be formed. Since a flat earth would not have poles through which the rotation axis would fit, this proves that the earth is not flat in terms of square addresses but only has a square address representation. Polar addresses are trivially achieved using the absolute address and address argument methods. Recovering the complex address is trivially achieved using the exponential function on the complex polar address.

The polar address also has a use for addressing Santa Claus, who is well known for living at the North Pole. This address can only be reached by use of the imaginary address, as it takes a certain amount of imagination in order to address Santa Claus. Traditional integer and fixed addressing schemes do not allow for such imaginative addresses, but the complex addressing scheme trivially handles it. The North American Aerospace Defense Command (NORAD) Santa Tracker would not have been possible without imaginative use of polar addresses when their secret phone address was revealed.

6.4. Root Server

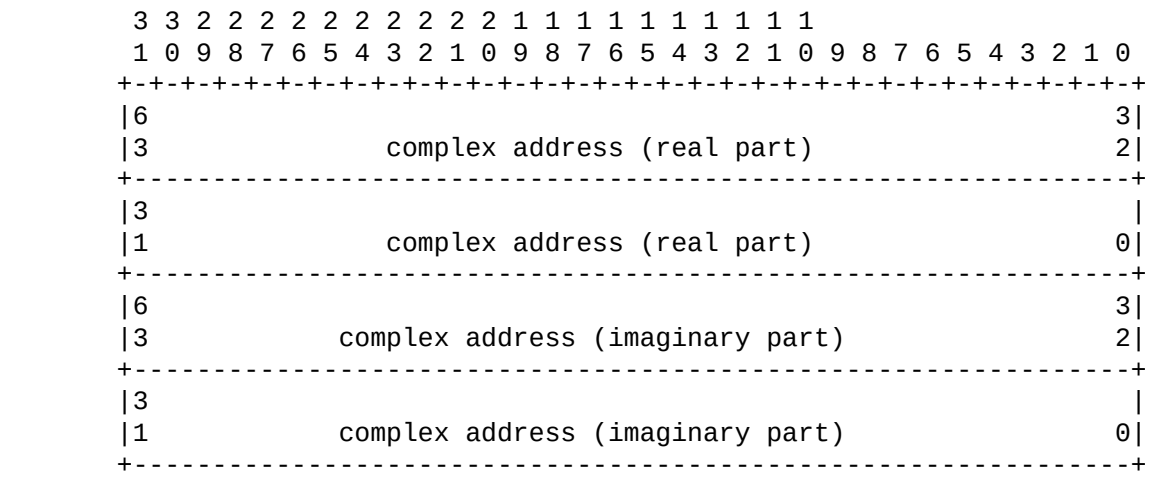
The DNS system uses a small set of known root servers, which provides the root service in order to attain the address of a node. The complex address provides a solution such that each client can in itself act as a root server as they now can use built-in floating-point hardware or software to get the root address from the squared address. This offloads the root servers for common benefits, but the traditional root servers can operate in parallel, easing the transition to the complex address system.

6.5. Implementation Considerations

Implementation of floating-point addresses and complex addresses, as needed for complex addressing schemes, is trivial in today's context. IEEE 754 [IEEE754] allows for a common and agreed-upon format for representing floating-point numbers. The 64-bit floating-point representation is well established and supported throughout a wide range of devices. Support also exists in a wide range of computer languages, including C and FORTRAN. The C standard library (or libc) essentially makes all modern languages support it in a consistent manner. An independent implementation exists for Intercal. With ISO C99 [C99], the <complex.h> include provides even more direct support for complex numbers, enabling efficient handling of all aspects of complex addressing with minimal implementation effort.

7. IPv6 Address Mapping

In order to convey complex addresses in the IPv6 address format, the following mapping is provided:



The 128-bit IPv6 address is divided into two 64-bit parts, where the upper half holds the real part of the address while the lower half holds the imaginary part of the complex address. These are represented as 64-bit floating-point numbers as defined in [IEEE754]; therefore, the real and imaginary address MUST be in the format described in IEEE 754.

Since the real address is held in the real part of the complex address and the imaginary address is held in the imaginary part of the complex address, the proposed representation allows for compiler optimization such that these operations can be performed without performance hits, as could otherwise be expected with any real or complex addressing scheme.

8. IANA Considerations

This document does not require any IANA actions, though IANA may find it mildly amusing.

9. Security Considerations

Complex addressing is considered unsafe, as division by 0 still provides Not a Number (NaN) values. Users will have to be careful to identify the NaN as they can indicate infinity addresses, which are unrealistic as one needs to confine the address length to the address space. Many other traditional unsafe operations for fixed-point addresses have, however, been resolved. For example, the error

condition of having the square address of -1 is readily resolved as the root address becomes the complex address i. Thus, it has the real part of 0, which is reasonable for an address that is not real, and an imaginary part of 1, which is in itself reasonable since one can imagine this error to occur.

Division by 0 and other floating-point address calculations can cause a floating-point interrupt, which causes the execution address to deviate; it is typically pushed on a stack and replaced by the interrupt handler address. Recovery from such interrupts may require further recursive calls; hence, the overall computation time is unpredictable. It can cause a complete core dump, and dumping the core can have significant effects on the propulsion system and the time to reach anywhere in the address space. Care must be taken to avoid such measures, or engineering will be quite upset. Dumping the core also widely breaks security protocols, as leaks can have widespread consequences. NaN is also known as "No Agency Number", to mark the importance of keeping things secure.

10. References

10.1. Normative References

[C99] ISO, "Information technology -- Programming Languages -- C", ISO/IEC 9899, 1999.

[IEEE754] IEEE, "IEEE Standard for Floating-Point Arithmetic", IEEE 754, DOI 10.1109/IEEESTD.2008.4610935.

[OSI] ISO, "Information technology -- Open Systems Interconnection -- Basic Reference Model: The Basic Model", ISO/IEC 7498-1, 1994.

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <http://www.rfc-editor.org/info/rfc4291>.

[RFC6214] Carpenter, B. and R. Hinden, "Adaptation of RFC 1149 for IPv6", RFC 6214, DOI 10.17487/RFC6214, April 2011, <http://www.rfc-editor.org/info/rfc6214>.

10.2. Informative References

[ICAO-A11] ICAO, "Air Traffic Services, Annex 11 to the Convention on International Civil Aviation", July 2001, <http://www.icao.int/secretariat/PostalHistory/annex_11_air_traffic_services.htm>.

[IEEE-802.17] IEEE, "IEEE Standard for Information Technology - Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks - Specific Requirements Part 17: Resilient Packet Ring (RPR) Access Method and Physical Layer Specifications", IEEE 802.17, DOI 10.1109/IEEESTD.2011.6026209.

[IEEE-802.5] IEEE, "IEEE Standard for Information Technology - Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks - Part 5: Token Ring Access Method and Physical Layer Specifications", IEEE 802.5, DOI 10.1109/IEEESTD.1992.7438701.

[PI] "Life of Pi", 20th Century Fox, 2012.

[pibill] Wikipedia, "Indiana Pi Bill", March 2017, <https://en.wikipedia.org/w/index.php?title=Indiana_Pi_Bill&oldid=770393894>.

[RFC5944] Perkins, C., Ed., "IP Mobility Support for IPv4, Revised", RFC 5944, DOI 10.17487/RFC5944, November 2010, <http://www.rfc-editor.org/info/rfc5944>.

Appendix A. Square Pi

When using square numbers, it is customary to use square pi, a number that has seen limited exposure in traditional texts but is widely used in computer science. It is thus appropriate to publish a few related notes on square pi in order to assist users of square addresses on its correct usage.

While traditional pi or round pi is an irrational number, it can be rounded off to 3.14 or 3.14159; it has an incomprehensible number of decimals, which is quite inappropriate for a round number, but as we keep rounding it to fit our needs, we keep rationalizing it from its irrational behavior.

The radius of an object is the closest to the center of the object you get. The circumference is the radius times 2 pi. The diameter is the shortest distance across the object, which is thus the radius times 2. The area is pi times the square of radius.

For a round circle, the radius is from the center to anywhere on the circumference. For a square circle, the radius only reaches the circumference on the four points located closest to the center. These are typically oriented such that the real and imaginary axis goes through them, which is helpful in calculations, and no rotation symmetries need to be considered.

The square pi fills the same purpose as the round pi, but rather than being adapted to round objects, it is adapted to square objects. For a square circle, the math is exactly the same as for round circles, provided that the square pi is used with square circles and that round pi is used with round circles.

The value of square pi is 4.

The value of square pi adapts really well to the way that computers calculate, which is also why computer results often are represented in square numbers, providing a bit of a square feeling. It should be noted that the square root of pi is often used, and the square root of square pi is naturally 2, which is very easy to handle in calculations and effectively reduces the risk of irrational numbers.

Please note that the square pi should not be confused with the Indiana Pi Bill [pibill], which does not discuss the square pi but a failed attempt to do square calculation of the area and circumference of a round circle using traditional tools like rulers and compasses.

Appendix B. Implementation Example

The following is a simple implementation example to illustrate how some core concepts can be implemented in <complex.h> (as defined in ISO C99 [C99]).

```
#include <complex.h>
#include <math.h>
#include <stdio.h>

// Define type for complex address
typedef complex ca;

// Create complex address
ca ca_create_complex_address(double real_address,
        double imaginary_address)
{
        return real_address + I * imaginary_address;
}

// Get real address
double  ca_get_real_address(ca ca_val)
{
        return creal(ca_val);
}

// Get imaginary address
double  ca_get_imaginary_address(ca ca_val)
{
        return cimag(ca_val);
}

// Get complex address
complex ca_get_complex_address(ca ca_val)
{
        return ca_val;
}

// Get floating address
double  ca_get_floating_address(ca ca_val)
{
        return creal(ca_val);
}

// Get physical address
double  ca_get_physical_address(ca ca_val)
{
        return cimag(ca_val);
```

```
}

// Get absolute address
double  ca_get_absolute_address(ca ca_val)
{
        return cabs(ca_val);
}

// Get address argument
double  ca_get_address_argument(ca ca_val)
{
        return carg(ca_val)*360/(2*M_PI);
}

int main()
{
        ca ca1, ca2;

        ca1 = ca_create_complex_address(1.0, 0.0);
        printf("The complex address (%f,%f)\n",
                creal(ca1), cimag(ca1));
        printf("has the real address %f and imaginary address %f\n",
                ca_get_real_address(ca1),
                ca_get_imaginary_address(ca1));
        printf("This represents the floating address %e and \
physical address %f\n", \
                ca_get_floating_address(ca1),
                ca_get_physical_address(ca1));
        ca2 = ca_create_complex_address(0.0, 1.0);
        printf("The complex address (%f,%f)\n",
                creal(ca2), cimag(ca2));
        printf("This represents the absolute address %f\n",
                ca_get_absolute_address(ca2));
        printf("The address argument resolution is %f\n",
                ca_get_address_argument(ca2));
        return 0;
}
```

Authors' Addresses

Magnus Danielson
Net Insight AB
Vastberga Alle 9
Hagersten 12630
Sweden

Email: magda@netinsight.net

Mans Nilsson
Besserwisser Networks

Email: mansaxel@besserwisser.org