IPv6 Operations
Internet-Draft
Intended status: Best Current Practice
Expires: July 6, 2016

L. Colitti
V. Cerf
Google
S. Cheshire
D. Schinazi
Apple Inc.
January 3, 2016

Host address availability recommendations
draft-ietf-v6ops-host-addr-availability-04

Abstract

This document recommends that networks provide general-purpose end hosts with multiple global IPv6 addresses when they attach, and describes the benefits of and the options for doing so.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In most aspects, the IPv6 protocol is very similar to IPv4. This similarity can create a tendency to think of IPv6 as 128-bit IPv4, and thus lead network designers and operators to apply identical configurations and operational practices to both. This is generally a good thing because it eases the transition to IPv6 and the operation of dual-stack networks. However, in some areas it can lead to carrying over IPv4 practices that are not appropriate in IPv6 due to significant differences between the protocols.

One such area is IP addressing, particularly IP addressing of hosts. This is substantially different because unlike IPv4 addresses, IPv6 addresses are not a scarce resource. In IPv6, each link has a virtually unlimited amount of address space [RFC7421]. Thus, unlike IPv4, IPv6 networks are not forced by address availability considerations to provide only one address per host. On the other hand, providing multiple addresses has many benefits including application functionality and simplicity, privacy, future applications, and the ability to deploy the Internet without the use of NAT. Providing only one IPv6 address per host negates these benefits.

This document describes the benefits of providing multiple addresses per host and the problems with not doing so. It recommends that networks provide general-purpose end hosts with multiple global addresses when they attach, and lists current options for doing so. It does not specify any changes to protocols or host behavior.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

2. Common IPv6 deployment model

IPv6 is designed to support multiple addresses, including multiple global addresses, per interface ([RFC4291] section 2.1, [RFC6434] section 5.9.4). Today, many general-purpose IPv6 hosts are configured with three or more addresses per interface: a link-local address, a stable address (e.g., using EUI-64 or Opaque Interface Identifiers [RFC7217]), one or more privacy addresses [RFC4941], and possibly one or more temporary or non-temporary addresses obtained using DHCPv6 [RFC3315].

In most general-purpose IPv6 networks, including all 3GPP networks ([RFC6459] section 5.2) and Ethernet and Wi-Fi networks using SLAAC [RFC4862], IPv6 hosts have the ability to configure additional IPv6 addresses from the link prefix(es) without explicit requests to the network.

3. Benefits of providing multiple addresses

Today, there are many host functions that require more than one IP address to be available to the host:

- Privacy addressing to prevent tracking by off-network hosts [RFC4941].

- Multiple processors inside the same device. For example, in many mobile devices both the application processor and baseband processor need to communicate with the network, particularly for recent technologies like ePDG.

- Extending the network (e.g., "tethering").

- Running virtual machines on hosts.

o Translation-based transition technologies such as 464XLAT [RFC6877] that provide IPv4 over IPv6. Some of these require the availability of a dedicated IPv6 address in order to determine whether inbound packets are translated or native ([RFC6877] section 6.3).

o ILA ("Identifier-locator addressing") [I-D.herbert-nvo3-ila].

o Future applications (e.g., per-application IPv6 addresses [TARP]).

Examples of how the availability of multiple addresses per host has already allowed substantial deployment of new applications without explicit requests to the network are:

o 464XLAT. 464XLAT is usually deployed within a particular network, and in this model the operator can ensure that the network is appropriately configured to provide the CLAT with the additional IPv6 address it needs to implement 464XLAT. However, there are deployments where the PLAT (i.e., NAT64) is provided as a service by a different network, without the knowledge or cooperation of the residential ISP (e.g., the IPv6v4 Exchange Service <http://www.jpix.ad.jp/en/service/ipv6v4.html>). This type of deployment is only possible because those residential ISPs provide multiple IP addresses to their users, and thus those users can freely obtain the extra IPv6 address required to run 464XLAT.

o /64 sharing [RFC7278]. When the topology supports it, this is a way to provide IPv6 tethering without needing to wait for network operators to deploy DHCPv6 PD, which is only available in 3GPP release 10 ([RFC6459] section 5.3).

4. Problems with restricting the number of addresses per host

Providing a restricted number of addresses per host implies that functions that require multiple addresses will either be unavailable (e.g., if the network provides only one IPv6 address per host, or if the host has reached the limit of the number of addresses available), or that the functions will only be available after an explicit request to the network is granted. The necessity of explicit requests has the following drawbacks:

o Increased latency, because a provisioning operation, and possibly human intervention with an update to the service level agreement, must complete before the functionality is available.

o Uncertainty, because it is not known in advance if a particular operation function will be available.

o Complexity, because implementations need to deal with failures and somehow present them to the user. Failures may manifest as timeouts, which may be slow and frustrating to users.

o Increased load on the network's provisioning servers.

Some operators may desire to configure their networks to limit the number of IPv6 addresses per host. Reasons might include hardware limitations (e.g., TCAM or neighbor cache table size constraints), business models (e.g., a desire to charge the network's users on a per-device basis), or operational consistency with IPv4 (e.g., an IP address management system that only supports one address per host). However, hardware limitations are expected to ease over time, and an attempt to generate additional revenue by charging per device may prove counterproductive if customers respond (as they did with IPv4) by using NAT, which results in no additional revenue, but leads to more operational problems and higher support costs.

5. Overcoming limits using Network Address Translation

These limits can mostly be overcome by end hosts by using NAT, and indeed in IPv4 most of these functions are provided by using NAT on the host. Thus, the limits could be overcome in IPv6 as well by implementing NAT66 on the host.

Unfortunately NAT has well-known drawbacks. For example, it causes application complexity due to the need to implement NAT traversal. It hinders development of new applications. On mobile devices, it reduces battery life due to the necessity of frequent keepalives, particularly for UDP. Applications using UDP that need to work on most of the Internet are forced to send keepalives at least every 30 seconds <http://www.ietf.org/proceedings/88/slides/slides-88-tsvarea-10.pdf>. For example, the QUIC protocol uses a 15-second keepalive [I-D.tsvwg-quic-protocol]. Other drawbacks of NAT are well known and documented [RFC2993]. While IPv4 NAT is inevitable due to the limited amount of IPv4 space available, that argument does not apply to IPv6. Guidance from the IAB is that deployment of IPv6 NAT is not desirable [RFC5902].

The desire to overcome the problems listed in Section 4 without disabling any features has resulted in developers implementing IPv6 NAT. There are fully-stateful address+port NAT66 implementations in client operating systems today: for example, Linux has supported NAT66 since late 2012 <http://kernelnewbies.org/Linux_3.7#head-103e14959eeb974bbd4e862df8afe7c118ba2beb>. A popular software hypervisor also recently implemented NAT66 to work around these issues <https://communities.vmware.com/docs/DOC-29954>. Wide

deployment of networks that provide a restricted number of addresses will cause proliferation of NAT66 implementations.

This is not a desirable outcome. It is not desirable for users because they may experience application brittleness. It is likely not desirable for network operators either, as they may suffer higher support costs, and even when the decision to provide only one IPv6 address per device is dictated by the network's business model, there may be little in the way of incremental revenue, because devices can share their IPv6 address with other devices. Finally, it is not desirable for operating system manufacturers and application developers, who will have to build more complexity, lengthening development time and/or reducing the time spent on other features.

Indeed, it could be argued that the main reason for deploying IPv6, instead of continuing to scale the Internet using only IPv4 and large-scale NAT44, is because doing so can provide all the hosts on the planet with end-to-end connectivity that is constrained not by accidental technical limitations, but only by intentional security policies.

6. Options for providing more than one address

Multiple IPv6 addresses can be provided in the following ways:

- Using Stateless Address Autoconfiguration [RFC4862]. SLAAC allows hosts to create global IPv6 addresses on demand by simply forming new addresses from the global prefix assigned to the link.

- Using stateful DHCPv6 address assignment [RFC3315]. Most DHCPv6 clients only ask for one non-temporary address, but the protocol allows requesting multiple temporary and even multiple non-temporary addresses, and the server could choose to provide multiple addresses. It is also technically possible for a client to request additional addresses using a different DUID, though the DHCPv6 specification implies that this is not expected behavior ([RFC3315] section 9). The DHCPv6 server will decide whether to grant or reject the request based on information about the client, including its DUID, MAC address, and so on.

- DHCPv6 prefix delegation [RFC3633]. DHCPv6 PD allows the client to request and be delegated a prefix, from which it can autonomously form other addresses. If the prefix is shorter than /64, it can be divided into multiple subnets which can be further delegated to downstream clients. If the prefix is a /64, it can be extended via L2 bridging, ND proxying [RFC4389] or /64 sharing [RFC7278], but it cannot be further subdivided, as a prefix longer than /64 is outside the current IPv6 specifications [RFC7421].

While [RFC3633] assumes that the DHCPv6 client is a router, DHCPv6 PD itself does not require that the client forward IPv6 packets not addressed to itself, and thus does not require that the client be an IPv6 router as defined in [RFC2460].

	SLAAC	DHCPv6 IA_NA / IA_TA	DHCPv6 PD	DHCPv4
Extend network	Yes	No	Yes	Yes (NAT44)
"Unlimited" endpoints	Yes*	Yes*	No	No
Stateful, request-based	No	Yes	Yes	Yes
Immune to layer 3 on-link resource exhaustion attacks	No	Yes	Yes	Yes

[*] Subject to network limitations, e.g., ND cache entry size limits.

Table 1: Comparison of multiple address assignment options

7. Number of addresses required

If we itemize the use cases from section Section 3, we can estimate the number of addresses currently used in normal operations. In typical implementations, privacy addresses use up to 8 addresses - one per day ([RFC4941] section 3.5). Current mobile devices may typically support 8 clients, with each one requiring one or more addresses. A client might choose to run several virtual machines. Current implementations of 464XLAT require use of a separate address. Some devices require another address for their baseband chip. Even a host performing just a few of these functions simultaneously might need on the order of 20 addresses at the same time. Future applications designed to use an address per application or even per resource will require many more. These will not function on networks that enforce a hard limit on the number of addresses provided to hosts.

8. Recommendations

In order to avoid the problems described above, and preserve the Internet's ability to support new applications that use more than one IPv6 address, it is RECOMMENDED that IPv6 network deployments provide multiple IPv6 addresses from each prefix to general-purpose hosts when they connect to the network. To support future use cases, it is RECOMMENDED to not impose a hard limit on the size of the address

pool assigned to a host. If the network requires explicit requests for address space (e.g., if it requires DHCPv6 to connect), it is RECOMMENDED that the network assign a /64 prefix to every host (e.g., via DHCPv6 PD). Using DHCPv6 IA_NA or IA_TA to request a sufficient number of addresses (e.g. 32) would accommodate current clients but sets a limit on the number of addresses available to hosts when they attach and would limit the development of future applications. Assigning prefixes longer than a /64 will limit the flexibility of the host to further assign addresses to any internal functions, virtual machines, or downstream clients that require address space - for example, by not allowing the use of SLAAC.

9. Operational considerations

9.1. Stateful addressing and host tracking

Some network operators - often operators of networks that provide services to third parties such as university campus networks - are required to track which IP addresses are assigned to which hosts on their network. Maintaining persistent logs that map user IP addresses and timestamps to hardware identifiers such as MAC addresses may be used to avoid liability for copyright infringement or other illegal activity.

It is worth noting that this requirement can be met without using stateful addressing mechanisms such as DHCPv6. For example, it is possible to maintain these mappings by scraping IPv6 neighbor tables, as routers typically allow periodic dumps of the neighbor cache via SNMP or other means, and many can be configured to log every change to the neighbor cache.

It is also worth noting that without L2 edge port security, hosts are still able to choose their own addresses - DHCPv6 does not offer any enforcement of what addresses a host is allowed to use. Such guarantees can only be provided by link-layer security mechanisms that enforce that particular IPv6 addresses are used by particular link-layer addresses (for example, SAVI [RFC7039]). If those mechanisms are available, it is possible to use them to provide tracking. This form of tracking is much more secure and reliable than DHCP server logs because it operates independently of how addresses are allocated. Additionally, attempts to track this sort of information via DHCPv6 are likely to become decreasingly viable due to ongoing efforts to improve the privacy of DHCP [I-D.ietf-dhc-anonymity-profile].

Thus, host tracking does not necessarily require the use of stateful address assignment mechanisms such as DHCPv6. Indeed, many large enterprise networks, including the enterprise networks of the

authors' employers, are fully dual-stack but do not currently use or support DHCPv6. The authors are directly aware of several networks that operate in this way, including Universities of Loughborough, Minnesota, Reading, Southampton, Wisconsin and Imperial College London.

9.2. Address space management

In IPv4, all but the world's largest networks can be addressed using private space [RFC1918], with each host receiving one IPv4 address. Many networks can be numbered in 192.168.0.0/16 which has roughly 64k addresses. In IPv6, that is equivalent to a /48, with each of 64k hosts receiving a /64 prefix. Under current RIR policies, a /48 is easy to obtain for an enterprise network.

Networks that need a bigger block of private space use 10.0.0.0/8, which has roughly 16 million addresses. In IPv6, that is equivalent to a /40, with each host receiving /64 prefix. Enterprises of such size can easily obtain a /40 under current RIR policies.

Currently, residential users typically receive one IPv4 address and a /48, /56 or /60 IPv6 prefix. While such networks do not provide enough space to assign a /64 per host, such networks almost universally use SLAAC, and thus do not pose any particular limit to the number of addresses hosts can use.

Unlike IPv4 where addresses came at a premium, in all these networks, there is enough IPv6 address space to supply clients with multiple IPv6 addresses.

9.3. Addressing link layer scalability issues via IP routing

The number of IPv6 addresses on a link has direct impact for networking infrastructure nodes (routers, switches) and other nodes on the link. Setting aside exhaustion attacks via Layer 2 address spoofing, every (Layer 2, IP) address pair impacts networking hardware requirements in terms of memory, MLD snooping, solicited node multicast groups, etc. Many of these costs are incurred by neighboring hosts.

Hosts on such networks that create unreasonable numbers of addresses risk impairing network connectivity for themselves and other hosts on the network, and in extreme cases (e.g., hundreds or thousands of addresses) may even find their network access restricted by denial-of-service protection mechanisms. We expect these scaling limitations to change over time as hardware and applications evolve. However, switching to a DHCPv6 PD model providing a dedicated /64

prefix per host resolves these scaling limitations, with only one routing entry and one ND cache entry per host on the network.

Also, a DHCPv6 PD model with a dedicated /64 per host makes it possible for the host to assign IPv6 addresses from this prefix to an internal interface such as a loopback interface. This obviates the need to perform Neighbor Discovery and Duplicate Address Detection on the network interface for these addresses, reducing network traffic.

10. Acknowledgements

The authors thank Tore Anderson, Brian Carpenter, David Farmer, Wesley George, Erik Kline, Shucheng (Will) Liu, Dieter Siegmund, Mark Smith, Sander Steffann and James Woodyatt for their input and contributions.

11. IANA Considerations

This memo includes no request to IANA.

12. Security Considerations

None so far.

13. References

13.1. Normative References

[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>.

13.2. Informative References

[I-D.herbert-nvo3-ila]
Herbert, T., "Identifier-locator addressing for network virtualization", draft-herbert-nvo3-ila-01 (work in progress), October 2015.

[I-D.ietf-dhc-anonymity-profile]
Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity profile for DHCP clients", draft-ietf-dhc-anonymity-profile-04 (work in progress), October 2015.

[I-D.tsvwg-quic-protocol]
Iyengar, J. and I. Swett, "QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2", draft-tsvwg-quic-protocol-01 (work in progress), July 2015.

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <http://www.rfc-editor.org/info/rfc1918>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC2993] Hain, T., "Architectural Implications of NAT", RFC 2993, DOI 10.17487/RFC2993, November 2000, <http://www.rfc-editor.org/info/rfc2993>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, DOI 10.17487/RFC3633, December 2003, <http://www.rfc-editor.org/info/rfc3633>.

[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>.

[RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, DOI 10.17487/RFC4389, April 2006, <http://www.rfc-editor.org/info/rfc4389>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <http://www.rfc-editor.org/info/rfc4862>.

[RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <http://www.rfc-editor.org/info/rfc4941>.

[RFC5902] Thaler, D., Zhang, L., and G. Lebovitz, "IAB Thoughts on IPv6 Network Address Translation", RFC 5902, DOI 10.17487/RFC5902, July 2010, <http://www.rfc-editor.org/info/rfc5902>.

[RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, DOI 10.17487/RFC6434, December 2011, <http://www.rfc-editor.org/info/rfc6434>.

[RFC6459] Korhonen, J., Ed., Soininen, J., Patil, B., Savolainen, T., Bajko, G., and K. Iisakkila, "IPv6 in 3rd Generation Partnership Project (3GPP) Evolved Packet System (EPS)", RFC 6459, DOI 10.17487/RFC6459, January 2012, <http://www.rfc-editor.org/info/rfc6459>.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <http://www.rfc-editor.org/info/rfc6877>.

[RFC7039] Wu, J., Bi, J., Bagnulo, M., Baker, F., and C. Vogt, Ed., "Source Address Validation Improvement (SAVI) Framework", RFC 7039, DOI 10.17487/RFC7039, October 2013, <http://www.rfc-editor.org/info/rfc7039>.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <http://www.rfc-editor.org/info/rfc7217>.

[RFC7278] Byrne, C., Drown, D., and A. Vizdal, "Extending an IPv6 /64 Prefix from a Third Generation Partnership Project (3GPP) Mobile Interface to a LAN Link", RFC 7278, DOI 10.17487/RFC7278, June 2014, <http://www.rfc-editor.org/info/rfc7278>.

[RFC7421] Carpenter, B., Ed., Chown, T., Gont, F., Jiang, S., Petrescu, A., and A. Yourtchenko, "Analysis of the 64-bit Boundary in IPv6 Addressing", RFC 7421, DOI 10.17487/RFC7421, January 2015, <http://www.rfc-editor.org/info/rfc7421>.

[TARP] Gleitz, PM. and SM. Bellovin, "Transient Addressing for Related Processes: Improved Firewalling by Using IPv6 and Multiple Addresses per Host", August 2001.

Authors' Addresses

Lorenzo Colitti
Google
Roppongi 6-10-1
Minato, Tokyo 106-6126
JP

Email: lorenzo@google.com

Vint Cerf
Google
1875 Explorer St
10th Floor
Reston, VA 20190
US

Email: vint@google.com

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
US

Email: cheshire@apple.com

David Schinazi
Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
US

Email: dschinazi@apple.com