Abstract

This document discusses problems that can occur when dual stack nodes that have IPv6 enabled by default are deployed in IPv4 or mixed IPv4 and IPv6 environments. The problems include application connection delays, poor connectivity, and network insecurity. The purpose of this memo is to raise awareness of these problems so that they can be fixed or worked around, not to try to specify whether IPv6 should be enabled by default or not.
Table of Contents

1. Introduction ...................................................... 3
2. No IPv6 Router .................................................... 3
   2.1 Problems with Default Address Selection for IPv6 ............ 3
   2.2 Neighbor Discovery’s On-Link Assumption Considered Harmful .................................................... 5
   2.3 Transport Protocol Robustness ................................ 6
3. Other Problematic Scenarios ....................................... 6
   3.1 IPv6 Network of Smaller Scope ................................ 6
      3.1.1 Alleviating the Scope Problem ............................ 7
   3.2 Poor IPv6 Network Performance ................................. 7
   3.3 Security .......................................................... 8
      3.3.1 Mitigating Security Risks .................................. 8
4. Application Robustness ............................................. 9
5. Security Considerations ........................................... 9
6. References .......................................................... 9
   6.1 Normative References .......................................... 9
   6.2 Informative References ......................................... 10
      Authors’ Addresses .............................................. 10
A. Acknowledgments .................................................. 11
B. Changes from draft-ietf-v6ops-v6onbydefault-02 .................. 11
C. Changes from draft-ietf-v6ops-v6onbydefault-01 .................. 11
D. Changes from draft-ietf-v6ops-v6onbydefault-00 .................. 11
   Intellectual Property and Copyright Statements ................. 13
1. Introduction

This document specifically addresses operating system implementations that implement the dual stack IPv6 model, and would ship with IPv6 enabled by default. It addresses the case where such systems are installed and placed in IPv4-only or mixed IPv4 and IPv6 environments, and documents potential problems that users on such systems could experience if the IPv6 connectivity is non-existent or sub-optimal. The purpose of this document is not to try to specify whether IPv6 should be enabled by default or not, but to raise awareness of the potential issues involved.

This memo begins in Section 2 by examining problems within IPv6 implementations that defeat the destination address selection mechanism defined in [RFC3484] and contribute to poor IPv6 connectivity. Starting with Section 3 it then examines other issues that network software engineers and network and systems administrators should be aware of when deploying dual stack systems with IPv6 enabled.

2. No IPv6 Router

Consider a scenario in which a dual stack system has IPv6 enabled and is placed on a link with no IPv6 routers. The system is using IPv6 Stateless Address Autoconfiguration [RFC2462], so it only has a link-local IPv6 address configured. It also has a single IPv4 address that happens to be a private address as defined in [RFC1918].

An application on this system is trying to communicate with a destination whose name resolves to public and global IPv4 and IPv6 addresses. The application uses an address resolution API that implements the destination address selection mechanism described in Default Address Selection for IPv6 [RFC3484]. The application will attempt to connect to each address, in the order they were returned, until one succeeds. Since the system has no off-link IPv6 routes, the optimal scenario would be if the IPv4 addresses returned were ordered before the IPv6 addresses. The following sections describe what things can go wrong with this scenario.

2.1 Problems with Default Address Selection for IPv6

The Default Address Selection for IPv6 [RFC3484] destination address selection mechanism could save the application a few useless connection attempts by placing the IPv4 addresses in front of the IPv6 addresses. This would be desired since all IPv6 destinations in this scenario are unreachable (there’s no route to them), and the system’s only IPv6 source address is inadequate to communicate with off-link destinations even if it did have an off-link route.
Let’s examine how the destination address selection mechanism behaves in the face of this scenario when given one IPv4 destination and one IPv6 destination.

The first rule, "Avoid unusable destinations", would prefer the IPv4 destination over the IPv6 destination, but only if the IPv6 destination were determined to be unreachable. The unreachability determination for a destination as it pertains to this rule is an implementation detail. One implementable method is to do a simple forwarding table lookup on the destination, and to deem the destination as reachable if the lookup succeeds. The Neighbor Discovery on-link assumption mentioned in Section 2.2 makes this method somewhat irrelevant, however, as an implementation of the assumption could simply be to insert an IPv6 default on-link route into the system’s forwarding table when the default router list is empty. The side-effect is that the rule would always determine that all IPv6 destinations are reachable. Therefore, this rule will not necessarily prefer one destination over the other.

The second rule, "Prefer matching scope", could prefer the IPv4 destination over the IPv6 destination, but only if the IPv4 destination’s scope matches the scope of the system’s IPv4 source address. Since [RFC3484] considers private addresses (as defined in [RFC1918]) of site-local scope, then this rule will not prefer either destination over the other. The link-local IPv6 source doesn’t match the global IPv6 destination, and the "site-local" IPv4 source doesn’t match the global IPv4 destination. The tie-breaking rule in this case is rule 6, "Prefer higher precedence". Since IPv6 destinations are of higher precedence than IPv4 destinations in the default policy table, the IPv6 destination will be preferred.

The solution in this case could be to add a new rule after rule 2 (rule 2.5) that avoids non-link-local IPv6 destinations whose selected source addresses are link-local. Of course, if the host is manually assigned a global IPv6 source address, then rule 2 will automatically prefer the IPv6 destination, and there is no fix other than to make sure rule 1 considers IPv6 destinations unreachable in this scenario.

Fixing the destination address selection mechanism by adding such a rule is only a mitigating factor if applications use standard name resolution API’s that implement this mechanism, and these applications try addresses in the order returned. This may not be an acceptable assumption in some cases, as there are applications that use hard coded addresses and address search orders and/or literal addresses passed in from the user.

For example, one such application is the DNS resolver. In this case,
a configuration file usually contains a list of literal addresses to be used as DNS name servers. The resolver client tries these servers in the order that they appear in the file, bypassing address selection rules.

Such applications will obviously be subject to whatever connection delays are associated with attempting a connection to an unreachable destination. This is discussed in more detail in the next few sections.

2.2 Neighbor Discovery’s On-Link Assumption Considered Harmful

Let’s assume that the application described in Section 2 is attempting a connection to an IPv6 address first, either because the destination address selection mechanism described in Section 2.1 returned the addresses in that order, or because the application isn’t trying the addresses in the order returned. Regardless, the user expects that the application will quickly connect to the destination. It is therefore important that the system quickly determine that the IPv6 destination is unreachable so that the application can try the IPv4 destination.

Neighbor Discovery’s [RFC2461] conceptual sending algorithm states that when sending a packet to a destination, if a host’s default router list is empty, then the host assumes that the destination is on-link. This issue is described in detail in [I-D.ietf-v6ops-onlinkassumption]. In summary, this assumption makes the unreachability detection of off-link nodes in the absence of a default router a lengthy operation. This is due to the cost of attempting Neighbor Discovery link-layer address resolution for each destination, and potential transport layer costs associated with connection timeouts. The transport layer issues are discussed later in Section 2.3.

On a network that has no IPv6 routing and no IPv6 neighbors, making the assumption that every IPv6 destination is on-link will be costly and incorrect. If an application has a list of addresses associated with a destination and the first 15 are IPv6 addresses, then the application won’t be able to successfully send a packet to the destination until the attempts to resolve each IPv6 address have failed. This could take 45 seconds (MAX_MULTICAST_SOLICIT * RETRANS_TIMER * 15). This could be compounded by any transport timeouts associated with each connection attempt, bringing the timeouts to even dozens of minutes.

If IPv6 hosts don’t assume that destinations are on-link as described above, then communication with destinations that are not on-link and unreachable should immediately fail. The IPv6 implementation should
be able to immediately notify applications or the transport layer that it has no route to such IPv6 destinations, so that applications won’t waste time waiting for address resolution to fail.

If hosts need to communicate with on-link destinations in the absence of default routers, then they need to be explicitly configured to have on-link routes for those destinations.

2.3 Transport Protocol Robustness

Making the same set of assumptions as Section 2.2, regardless of how long the network layer takes to determine that the IPv6 destination is unreachable, the delay associated with a connection attempt to an unreachable destination can be compounded by the transport layer. When the unreachability of a destination is obviated by the reception of an ICMPv6 destination unreachable message, the transport layer should make it possible for the application or API to deal with this. It could fail the connection attempt, pass ICMPv6 errors up to the application, or pass them up to an API that is handling this for the application, etc.

3. Other Problematic Scenarios

This section describes problems that could arise for a dual stack system with IPv6 enabled when placed on a network with IPv6 connectivity.

3.1 IPv6 Network of Smaller Scope

A network that has a smaller scope of connectivity for IPv6 as it does for IPv4 could be a problem in some cases. If applications have access to name to address mapping information that is of greater scope than the connectivity to those addresses, there is obvious potential for suboptimal network performance. Hosts will attempt to communicate with IPv6 destinations that are outside the scope of the IPv6 routing, and depending on how the scope boundaries are enforced, applications may not be notified that packets are being dropped at the scope boundary.

If applications aren’t immediately notified of the lack of reachability to IPv6 destinations, then they aren’t able to efficiently fall back to IPv4. They then have to rely on transport layer timeouts which can be minutes in the case of TCP.
An example of such a network is an enterprise network that has both IPv4 and IPv6 routing within the enterprise and has a firewall configured to allow some IPv4 communication, but no IPv6 communication.

3.1.1 Alleviating the Scope Problem

To allow applications to correctly fall back to IPv4 when IPv6 packets are destined beyond their allowed scope, the devices enforcing the scope boundary must send ICMPv6 Destination Unreachable messages back to senders of such packets. The sender’s transport layer should act on these errors as described in Section 2.3.

3.2 Poor IPv6 Network Performance

Most applications on dual stack nodes will try IPv6 destinations first by default due to the Default Address Selection mechanism described in [RFC3484]. If the IPv6 connectivity to those destinations is poor while the IPv4 connectivity is better (i.e., the IPv6 traffic experiences higher latency, lower throughput, or more lost packets than IPv4 traffic), applications will still communicate over IPv6 at the expense of network performance. There is no information available to applications in this case to advise them to try another destination address.

An example of such a situation is a node which obtains IPv4 connectivity natively through an ISP, but whose IPv6 connectivity is obtained through a configured tunnel whose other endpoint is topologically such that most IPv6 communication is done through triangular IPv4 paths. Operational experience on the 6bone shows that IPv6 RTT’s are poor in such situations.
3.3 Security

Enabling IPv6 on a host implies that the services on the host may be open to IPv6 communication. If the service itself is insecure and depends on a security policy enforced somewhere else on the network (such as in a firewall), then there is potential for new attacks against the service.

A firewall may not be enforcing the same policy for IPv4 as for IPv6 traffic, which could be due to misconfiguration of the firewall. One possibility is that the firewall could have more relaxed policy for IPv6, perhaps by letting all IPv6 packets pass through, or by letting all IPv4 protocol 41 packets pass through. In this scenario, the dual stack hosts within the protected network could be subject to different attacks than for IPv4.

Even if a firewall has a stricter policy or identical policy for IPv6 traffic than for IPv4 (the extreme case being that it drops all IPv6 traffic), IPv6 packets could go through the network untouched if tunneled over a transport layer. This could open the host to direct IPv6 attacks. It should be noted that IPv4 packets can also be tunneled, so this is not a new security concern for IPv6. Firewalls must be deliberately and properly configured.

A similar problem could exist for virtual private network (VPN) software. A VPN could protect all IPv4 packets but transmit all others onto the local subnet unprotected. At least one widely used VPN behaves this way. This is problematic on a dual stack host that has IPv6 enabled on its local network. It establishes its VPN link and attempts to communicate with destinations that resolve to both IPv4 and IPv6 addresses. The destination address selection mechanism prefers the IPv6 destination so the application sends packets to an IPv6 address. The VPN doesn’t know about IPv6, so instead of protecting the packets and sending them to the remote end of the VPN, it passes such packets in the clear to the local network.

This is problematic for a number of reasons. The first is that if the node has a default IPv6 route, the packets will be forwarded off-link to an unknown destination. Another is if no legitimate router is on-link and the node makes the on-link assumption discussed in Section 2.2, the packets will simply be sent onto the local link to be potentially viewed by a node spoofing the destination. A third is if a rogue IPv6 router exists on-link. In that case the malicious node will simply be sent all IPv6 packets in the clear.

3.3.1 Mitigating Security Risks

The security policy implemented in firewalls, VPN software, or other
devices, must take a stance whether it applies equally to both IPv4 and IPv6 traffic. It is probably desirable for the policy to apply equally to both IPv4 and IPv6, but the most important thing is to be aware of the potential problem, and to make the policy clear to the administrator and user.

There is still a risk that IPv6 packets could be tunneled over a transport layer such as UDP, implicitly bypassing the security policy. Some more complex mechanisms could be implemented to apply the correct policy to such packets. This could be easy to do if tunnel endpoints are co-located with a firewall, but more difficult if internal nodes do their own IPv6 tunneling.

4. Application Robustness

Enabling IPv6 on a dual stack node is only useful if applications that support IPv6 on that node properly cycle through addresses returned from name lookups and fall back to IPv4 when IPv6 communication fails. Simply cycling through the list of addresses returned from a name lookup when attempting connections works in most cases for most applications, but there are still cases where that’s not enough. Applications also need to be aware that the fact that a dual stack destination’s IPv6 address is published in the DNS does not necessarily imply that all services on that destination function over IPv6. This problem, along with a thorough discussion of IPv6 application transition guidelines, is discussed in [I-D.ietf-v6ops-application-transition].

5. Security Considerations

This document raises security concerns in Section 3.3. They are summarized below:

- Firewalls need to be configured properly to have deliberate security policies for IPv6 packets, including IPv6 packets encapsulated in other layers.

- Implementations of virtual private networks need to have a deliberate IPv6 security policy that doesn’t allow packets to accidentally appear in the clear when they were intended to be sent securely over the VPN.

6. References

6.1 Normative References

[I-D.ietf-v6ops-application-transition]

[I-D.ietf-v6ops-onlinkassumption]


6.2 Informative References

[I-D.ietf-tsvwg-sctpimpguide]


Authors’ Addresses

Sebastien Roy
Sun Microsystems, Inc.
1 Network Drive
UBU02-212
Burlington, MA  01801

EMail: sebastien.roy@sun.com
Appendix A. Acknowledgments

The authors gratefully acknowledge the contributions of Jim Bound, Fernando Gont, Tony Hain, Tim Hartrick, Mika Liljeberg, Erik Nordmark, Kacheong Poon, Pekka Savola, Randall Stewart, and Ronald van der Pol.

Appendix B. Changes from draft-ietf-v6ops-v6onbydefault-02

- Removed all text suggesting solutions to the problems described by this draft.
- Removed the all sub-sections of Section 2.3 that offered solutions to the problems being presented.
- Removed Section 3.2.1, which described a solution to dealing with poor IPv6 network performance.

Appendix C. Changes from draft-ietf-v6ops-v6onbydefault-01

- Added specificity to the DNS resolver problem in Section 2.1.
- Added a few paragraphs in Section 2.3.1.1 describing potential drawbacks to TCP aborting connections ins SYN-SENT or SYN-RECEIVED state.
- Added Section 2.3.1.3 describing how a higher level API could be used to manage connections.
- Expanded Section 2.3.3 to describe desired SCTP behavior when encountering soft errors.
- Added a summary of security concerns to Section 5.
- Miscellaneous editorial changes.

Appendix D. Changes from draft-ietf-v6ops-v6onbydefault-00

- Clarified in the abstract and introduction that the document is
meant to raise awareness, and not to specify whether IPv6 should be enabled by default or not.
o Shortened Section 2.2 and made reference to [I-D.ietf-v6ops-onlinkassumption].

o Added clarification in Section 2.3 about packets that are lost without ICMPv6 notification.

o Section 2.3 now has subsections for TCP, UDP, and SCTP.

o Removed text in Section 2.3.1.1 suggesting that hosts usually were only assigned one address when [RFC1122] was written.

o Added text in Section 2.3.1.1 suggesting a method for applications to advise TCP of their preference for ICMPv6 handling.

o Added Section 2.3.1.2.

o Added Section 2.3.2.

o Added Section 2.3.3.

o Strengthened wording in Section 3.1.1 to suggest that devices enforcing scope boundaries must send ICMPv6 Destination Unreachable messages.

o Clarified that the VPN problem described in Section 3.3 is due to a combination of the VPN software and either the on-link assumption and/or a "bad guy".

o Shortened Section 4 and made reference to [I-D.ietf-v6ops-application-transition].

o Miscellaneous editorial changes.
 Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF’s procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.