TOC

none yet	A. Suhonen
Internet-Draft	Tampere University of Technology,
Updates: 3484 (if approved)	Finland
Intended status: Experimental	July 29, 2009
Expires: January 30, 2010	

Address Selection Using Source Address Specific Routing Tables
draft-axu-addr-sel-00

Status of this Memo

Copyright Notice

Abstract

RFC 3484 defines two algorithms for default source and destination address selection, but it has several shortcomings as specified in RFC 5220. RFC 5221 lists some requirements for any attempts to update the original RFC. This document specifies an alternate address selection algorithm to fulfill those requirements.

Table of Contents

1. Introduction

TOC

[RFC3484] (Draves, R., “Default Address Selection for Internet Protocol version 6 (IPv6),” February 2003.) defines default address selection rules for IPv6 and IPv4. Several shortcomings in the original address selection rules have been identified in [RFC5220] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules,” July 2008.) and its sister document [RFC5221] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Requirements for Address Selection Mechanisms,” July 2008.) specifies some requirements for any attempts to update the original address selection algorithm.

A further concern comes from multipath protocols. When SCTP (Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L., and V. Paxson, “Stream Control Transmission Protocol,” October 2000.) [RFC2960], for example, finds that its active source destination address pair is no longer functional, it will need to start searching for a new one.
The communicating hosts may both have a dozen addresses so it might take unacceptably long to iterate through all combinations before finding a functional pair. On the other hand, many of the invalid combinations could be filtered out using this algorithm, making the process noticeably faster.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.) [RFC2119].

2. Filter Algorithm

When a host has several addresses, they SHOULD each be associated with their own routing tables. When selecting source and destination addresses, the first stage is to filter out combinations where the routing table attached with the source (local) address does not have a valid route for the destination (remote) address. In other words, if a destination address can't be found from the routing table for a given source address the system MUST discard that destination address for that source address.
If none of the possible destination addresses can be found in the routing table for a source address, then that source address MUST be discarded for those destination addresses.
One side effect of this filter algorithm is that it doesn't need to know anything about scopes. The routing tables associated with source address candidates will determine what destination addresses they are usable with. This effect is demonstrated below and later in this document.

2.1. Link Local Scope

The routing table associated with a link local address (e.g. 169.254.123.45%le0) SHOULD only have one external unicast route, the link local network for that link (e.g. 169.254.0.0%le0 /16). In addition, if the host supports multicast on this link, a route for the local scope multicast space SHOULD also appear in this table.
This means that the link local address is usable only with other link local addresses on the same link.
The localhost addresses and prefixes (127.0.0.1/8 and ::1/128) SHOULD be treated like link local scope in this algorithm.

2.2. Autoconfiguration for Global Scope

When addresses are assigned to interfaces dynamically through stateless or stateful autoconfiguration the process usually also yields a default route. That default route SHOULD be placed only into the routing table associated with that address. In addition, if the host and network support multicast, a route for the global scope multicast space SHOULD also appear in this table.
This usually means that the next hop of that default route will only be useable with the source address learned from that default router.
Some autoconfiguration methods (see [RFC3442] (Lemon, T., Cheshire, S., and B. Volz, “The Classless Static Route Option for Dynamic Host Configuration Protocol (DHCP) version 4,” December 2002.) and [RFC4191] (Draves, R. and D. Thaler, “Default Router Preferences and More-Specific Routes,” November 2005.)) can be used to communicate other routes in addition to the default route. Those routes SHOULD likewise be added only into the routing table associated with the address configured using that same interchange.
Examples of autoconfiguration methods include RARP, DHCPv4, ICMPv6 RA, DHCPv6, Teredo, 6to4, ISATAP, PPP, mDNS.

2.3. Site Local Scope

The routing tables for site local addresses SHOULD have routes for site local address space. They SHOULD NOT have the default route, so that they would be automatically eliminated when selecting address pairs for site external communication.
However, if the site edge automatically translates site local addresses to global addresses, the routing tables associated with site local scope addresses MAY have the default route.

2.4. Additional Filter Constraints

TOC

The address selection algorithm MAY also be given additional filter constraints, such as "use only link#3" or "do not use next-hop 10.0.0.1". [RFC5014] (Nordmark, E., Chakrabarti, S., and J. Laganier, “IPv6 Socket API for Source Address Selection,” September 2007.) specifies an interface that does something very similar.
Work is going on in the MIF-wg (Blanchet, M. and P. Seite, “Multiple Interfaces Problem Statement,” June 2009.) [I-D.blanchet-mif-problem-statement] to tie address selection and next-hop selection with DNS resolver selection and other similar resources. That is, when using the DNS resolvers received from one DHCP server, the terminal should also always use the default route received from that DHCP server.
This algorithm supports those efforts by making it possible to restrict a process to one routing table for both address resolution and selection.

2.5. Forwarding

TOC

If a host is configured to forward packets between networks, it SHOULD combine the routing tables for the networks in question into one. Link local scope tables MUST NOT be combined.
If the host has multiple addresses from different global scope prefixes then system administration MAY specify which addresses are combined to form routing tables. The resulting functionality resembles the VRF functionality found in some modern routers.
One purpose behind this algorithm is to move source routing burden from the network to the host. So if a router wants to advertise two (or more) prefixes on the subnet, but to keep their routing separate, it should use different link local and link layer addresses when advertising them. It can then choose the correct VRF to forward a packet depending on which link layer address it received it on.

2.6. Dynamic Routing Protocols

TOC

Hosts don't usually run dynamic routing protocols, but since they sometimes do, this subsection is included for completeness.
Dynamic routing protocol instances are usually bound to links or interfaces. With this algorithm network administrators MAY bind routing protocol instances to specific addresses or prefixes on a link and the routing tables associated with them. The routing protocol instance MUST update only the routing table it is associated with.

A reasonable default setting is that all addresses that are not link local are associated with the routing protocol instance. Thus, they will share a routing table.
If the network administration wants to separate traffic belonging to different upstream operator prefixes, it may wish to run separate routing protocol instances throughout the network for different upstream prefixes.

---

3. Precedences and Labels

TOC

TBD
My original thought was to follow the metrics systems of the original RFC here, since candidate filtering and proper next hop selection were my primary concerns. However, it might be a good idea to just rethink the issue one more time.
Perhaps it might be a good idea to associate preferences with individual routes and/or whole routing tables. In that case, the routing table lookup performed in the filtering phase would also yield the precedence of the address in addition to next-hop information.
The label abstraction used by the original RFC loosely corresponds to the routing table abstraction in this algorithm. That is, different scopes had different labels in [RFC3484] (Draves, R., “Default Address Selection for Internet Protocol version 6 (IPv6),” February 2003.) but in this algorithm different scopes SHOULD have their own routing tables.
The rest of this section outlines one approach to sorting addresses by preference.

3.1. Route and Table Preferences

TOC

Each routing table has a default precedence, meaning all routes added to that table will have that precedence in the absence of a specific precedence.
This precedence MUST be used to sort the source and destination address pairs according to preference. In effect, the precedence is for the address pair, not for a single address.
When two routes have the same precedence, their prefix lengths MUST be compared and the longer prefix MUST be considered more preferable.
The algorithm normally performs both source and destination address selection simultaneously and efficiently.
In order to perform source address selection, only one destination address SHOULD be presented to the algorithm, which will then look for the address in all tables and sort the source addresses where it was found according to the precedences.

In order to perform destination address selection, only one source address SHOULD be presented to the algorithm along with the set of destination addresses. The algorithm will then look for all the given destination addresses in the table associated with the source address and sort the results according to the precedences.

3.1.1. Local and Link Local Scope Routing Tables

The default precedence for all local and link local scope route entries SHOULD be 50.

3.1.2. Global Scope Routing Tables

The default precedence for all global scope route entries SHOULD be 40. System or network administrators or operating systems MAY alter this default precedence to account for things like link speeds. Such environmental precedence modifiers SHOULD NOT alter the precedence by more than +-4.
The system MAY automatically add depreference routes to global scope routing tables. These routes will cover address space reserved for transition techniques, such as 2002::/16 (FIXME: add xrefs) and 2001::/32. They SHOULD have the same next-hop information as the default route in the same table, but their precedence SHOULD be 15.
The system MAY automatically add blackhole routes to global scope routing tables for illegal address combinations. An example of such an illegal combination is IPv6 prefix 2002:a00::/24, which corresponds to 6to4 addresses generated from IPv4 addresses inside 10.0.0.0/8 which can't be used on the Internet.

3.1.3. Transition Technique Routing Tables

The default precedence for all route entries for source addresses generated through transition techniques SHOULD be 30.
The transition table SHOULD NOT of course have a depreference route for its own address space. Instead, the precedence of the route for its own address space SHOULD be 35.
Individual transition techniques or the system administrator MAY specify different default precedences to establish relative preferences between transition techniques or the proxies/servers associated with them.

3.1.4. IPv4 Compatible Routing Tables

TOC

The default precedence for all IPv4 compatible global scope route entries SHOULD be 20.

3.1.5. Reachability Information

TOC

If the next-hop information associated with a route in any table has been found unreachable or the interface link is down the precedence of that route MAY be temporarily dropped to zero until it works again.

4. RFC3484 Rule Comparison

TOC

The algorithm defined by [RFC3484] (Draves, R., “Default Address Selection for Internet Protocol version 6 (IPv6),” February 2003.) uses a set of rules to perform its function. Those rules are compared to this algorithm in this section.
FIXME: write this section

5. RFC5220 Concerns

TOC

[RFC5220] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules,” July 2008.) presents several problems and issues with the original default address selection algorithm. The following subsections address these issues.

5.1. Multiple Routers on a Single Interface

TOC

This problem was one of the starting points for the development of this algorithm. This algorithm solves the problem by having separate routing tables for addresses learned from different routers.

TOC

5.2. Ingress Filtering Problem

This problem was one of the starting points for the development of this algorithm. This algorithm solves the problem by having separate routing tables for different addresses.

5.3. Half-Closed Network Problem

TOC

This problem was one of the starting points for the development of this algorithm. This algorithm solves the problem by having separate routing tables for different addresses.
System or network administration MUST specify allowed or disallowed connections by modifying the routing tables.

5.4. Combined Use of Global and ULA

TOC

This algorithm solves the problem by having separate routing tables for different addresses. Scope of address usage is controlled by the routing tables.
Implementations MAY recognize ULA addresses and other site local addresses as scopes of their own, and treat them properly when autogenerating the routing tables.
System or network administration MUST specify allowed or disallowed address pair selection by modifying the routing tables.

5.5. Site Renumbering

TOC

When the autoconfiguration client discovers that a prefix or address has been deprecated, it SHOULD drop the route precedences for all the routes associated with the deprecated resource to zero.
When such deprecated routing information finally times out and is no longer in use, the routing table associated with it MAY be removed entirely.

5.6. Multicast Source Address Selection

TOC

TBD

5.7. Temporary Address Selection

TOC

Conceivably temporary addresses could be associated with routing tables of their own, instead of sharing routing tables with the addresses used to generate the temporary addresses.
The precedences for the table for a temporary address would be lower than that of a similar but more permanent address. Clients wishing to make use of the temporary address would add appropriate constraints to their address selection.
Alternatively, if the system or network administration wishes that the host use a temporary address with some certain destination network, a route to that network could be added to the routing table for the temporary address with a higher than normal precedence.

5.8. IPv4 or IPv6 Prioritization

TOC

This is a configuration issue with the routing tables.
Connection pooling, as specified in Section 7.3 (Connection Pooling), could mitigate this problem.

5.9. ULA and IPv4 Dual-Stack Environment

TOC

This special case is easily handled by omitting the default route for the routing table for ULA addresses.

5.10. ULA or Global Prioritization

TOC

Already covered in Section 5.4 (Combined Use of Global and ULA).

---

6. RFC5221 Requirements

TOC

[RFC5221] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Requirements for Address Selection Mechanisms,” July 2008.) defines a set of requirements for the address selection algorithm. The subsection headings used in that document have been copied here and an explanation of how this algorithm deals with each issue is given.

6.1. Effectiveness

TOC

The effectiveness of the proposed solution to solve problems presented in [RFC5220] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules,” July 2008.) is covered by Section 5 (RFC5220 Concerns).

6.2. Timing

TOC

This algorithm relies on other methods and protocols to submit address selection configuration and information and to place it in the routing table.
Once the routing table is updated, the address selection algorithm will start making decisions based on the new information.

6.3. Dynamic Behavior Update

TOC

From the point of view of this algorithm, this problem is a feature of autoconfiguration methods. If the autoconfiguration methods rewrite routing tables, the address selection algorithm will always use the updated information when it's invoked.

6.4. Node-Specific Behavior

TOC

From the point of view of this algorithm, this problem is a feature of autoconfiguration methods. This algorithm will happily make address selection decisions according to any input it is given.

6.5. Application-Specific Behavior

TOC

Additional filter constraints from Section 2.4 (Additional Filter Constraints) can be used to influence address selection per application.

TOC

6.6. Multiple Interface

This algorithm doesn't differenciate between cases where a host has multiple interfaces and where it has multiple prefixes on a single interface. If it solves a problem satisfactorily for one case, it solves it identically for the other case as well.

6.7. Central Control

TOC

This algorithm doesn't specify new methods for central control. It does, however, work well with other protocols that provide methods of central control, such as routing protocols.

6.8. Next-Hop Selection

TOC

The next-hop and interface used is a side product of the source address specific routing table lookup, which is performed in the filtering stage.
A very pleasing feature of this algorithm is that there can be multiple routers advertising different prefixes on the same subnet, and this algorithm will still select proper address pairs and next-hops to satisfy any SAVI requirements.

6.9. Compatibility with RFC 3493

TOC

TBD
On first impression, this algorithm shouldn't have any impact on the Socket API. Then again, routing table index could be referenced as part of some process.
Solaris, for example, creates new alias-interfaces for each new address assigned to a physical interface. So if_index could also be used to uniquely identify a source address specific routing table on that platform. Other operating systems do not work the same way.

6.10. Compatibility and Interoperability with RFC 3484

TOC

When a host implementing this address selection algorithm and a host implementing the [RFC3484] (Draves, R., “Default Address Selection for Internet Protocol version 6 (IPv6),” February 2003.) algorithm interact, this algorithm will become constrained by the choices made by the peer.

6.11. Security

TOC

Security issues raised in [RFC5221] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Requirements for Address Selection Mechanisms,” July 2008.) are covered by Section 10.2 (RFC5221 Requirements).

7. Implementation Issues and Other Concerns

TOC

Some popular operating systems already implement all the features required to implement this algorithm. In such cases all that is required is to integrate the features together.
The trickiest feature required by this algorithm is probably support for multiple routing tables. This may also create backward compatibility issues in some implementations. More discussion may be required here.

7.1. Low Memory and Power Concerns

TOC

The biggest worry is that creating lots of routing tables will waste memory and power. However, when compared to the old way (see Appendix A (Routing Table Example)), memory consumption doesn't explode. Every route that was present in the monolithic routing table will usually be present in only one source address specific routing table.
CGAs (ADD XREF) MAY reuse the same routing table.

7.2. Differing Larger Scopes

TOC

The default route for global scope addresses is 0::0/0, but this route will also cover addresses of potentially incompatible scopes. For example, the basic algorithm would accept a link local destination address with a global scope source address.
One way to prevent this would be to add blackhole routes into the routing tables of global scope addresses for address space belonging to incompatible scopes. The filter algorithm SHOULD treat a blackhole

route as an indication that no valid route was found for addresses matching the blackhole in that table.

7.3. Connection Pooling

TOC

When trying to establish a new connection, the stack MAY send open packets to all source/destination/nexthop combinations that pass the filter stage at a pace of three per second until it receives a response.
When the connection is established the addresses are fixed (for non-multipathing protocols, such as TCP).
If the peer also responds to the other connection attempts after the first connection is established, those connections MAY either be reset immediately, or the stack MAY pool them for a short while in an incomplete handshake state, in case some application tries to open an identical socket.
This would benefit applications such as web browsers, mail transfer agents and database clients, which routinely create more than one connection between the same two hosts and the same destination port.
It would also benefit dual stacked or multi-homed hosts where some of the addresses or networks are misconfigured and don't work.

7.4. Using Just One Table with Tags

TOC

It is possible to implement this algorithm with just one routing table, if tags or bitfields are used to identify which routing table each route really belongs to.
However, since a less specific route in one table can have higher precedence than a more specific route in another table, care must be taken in the implementation.
It is also possible to implement this algorithm without interfering with the actual routing table at all, by just mirroring all the routing table information and changes in a policy table used by this algorithm only.

8. Acknowledgements

TOC

This document was written using the template derived from an initial version written by Pekka Savola and contributed by him to the xml2rfc project.

Thanks to the following people for giving feedback during the writing of this document: Jari Arkko, Jan Melen, Arifumi Matsumoto, James Morse, .

9. IANA Considerations

TOC

This document has no IANA Actions.

10. Security Considerations

TOC

10.1. RFC5220 Considerations

TOC

Section 4 of [RFC5220] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules,” July 2008.) raises a concern that a malicious attacker can gather information about addresses connected to the target host by triggering the address selection algorithm on the target host by various methods and listening to what candidates it produces.
This algorithm doesn't completely remove that possibility, but due to the filtering stage, the attacker can only gain information on addresses routable to the address used by the attacker.

10.2. RFC5221 Requirements

TOC

Section 3 of [RFC5221] (Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, “Requirements for Address Selection Mechanisms,” July 2008.) lists two security concerns which are dealt with in subsections below.

10.2.1. List of threats introduced by new address-selection mechanism

TOC

This specification relies on existing autoconfiguration methods and routing protocols to distribute address selection hints. Each of those

SHOULD have their own methods to combat leakage, hijacking and denial of service.

10.2.2. List of recommendations in which security mechanism should be applied

TOC

This specification relies on existing autoconfiguration methods and routing protocols to distribute address selection hints. Each of those SHOULD have their own methods to combat leakage, hijacking and denial of service.

11. References

TOC

11.1. Normative References

TOC

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC2960]	Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L., and V. Paxson, "Stream Control Transmission Protocol," RFC 2960, October 2000 (TXT).
[RFC3442]	Lemon, T., Cheshire, S., and B. Volz, "The Classless Static Route Option for Dynamic Host Configuration Protocol (DHCP) version 4," RFC 3442, December 2002 (TXT).
[RFC3484]	Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)," RFC 3484, February 2003 (TXT).
[RFC4191]	Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes," RFC 4191, November 2005 (TXT).
[RFC5220]	Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, "Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules," RFC 5220, July 2008 (TXT).
[RFC5221]	Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, "Requirements for Address Selection Mechanisms," RFC 5221, July 2008 (TXT).

11.2. Informative References

TOC

[I-D.blanchet-mif-problem-statement]	Blanchet, M. and P. Seite, “Multiple Interfaces Problem Statement,” draft-blanchet-mif-problem-statement-01 (work in progress), June 2009 (TXT).
[RFC5014]	Nordmark, E., Chakrabarti, S., and J. Laganier, “IPv6 Socket API for Source Address Selection,” RFC 5014, September 2007 (TXT).

Appendix A. Routing Table Example

TOC

This section demonstrates how this algorithm affects the routing table of a multi-homed host. Appendix A.1 (Before) shows the routing table using only methods without this algorithm. Appendix A.2 (After Conversion) shows the routing tables produced on the same host if this algorithm is applied.

A.1. Before

TOC

This routing table was initially copied from a system running Linux 2.6.25. The addresses were then greatly simplified to make the table fit better on the page.

Network	Next-Hop	Link	Metric
2001::/32	::	teredo	256
2001:db8:1::/64	::	eth0	256
2001:db8:2::/64	::	eth1	256
fe80::/64	::	teredo	256
fe80::/64	::	eth0	256
fe80::/64	::	eth1	256
::/0	::	teredo	1029
::/0	fe80::13	eth0	1024
::/0	fe80::ce	eth1	1024
::/0	::	lo	-1 !U
::1/128	::	lo	0
2001:db8:1:0:a00:ff:fedc:a/128	::	lo	0

2001:db8:2:0:200:ff:fec4:b/128	::	lo	0
2001:0:c200:201::3/128	::	lo	0
fe80::a00:ff:fedc:a/128	::	lo	0
fe80::200:ff:fec4:b/128	::	lo	0
fe80::ffff:ffff:ffff/128	::	lo	0

Table 1: Routing Table w/o Modifications

"!U" after metric denotes unreachable or blackhole routes.

A.2. After Conversion

TOC

These tables contain and implement just the basic idea. Thus the combined size of these tables is equal to Table 1 (Routing Table w/o Modifications). Optional improvements are presented in the next subsection.

Network	Next-Hop	Link	Metric
::/0	::	lo	-1 !U
::1/128	::	lo	50

Table 2: Routing Table for ::1

Network	Next-Hop	Link	Metric
2001::/32	::	teredo	35
::/0	::	teredo	30
2001:0:c200:201::3/128	::	lo	50

Table 3: Routing Table for 2001:0:c200:201::3%teredo

Network	Next-Hop	Link	Metric
fe80::/64	::	teredo	50
fe80::ffff:ffff:ffff/128	::	lo	50

Table 4: Routing Table for fe80::ffff:ffff:ffff%teredo

Network	Next-Hop	Link	Metric
2001:db8:1::/64	::	eth0	40
::/0	fe80::13	eth0	40
2001:db8:1:0:a00:ff:fedc:a/128	::	lo	50

Table 5: Routing Table for 2001:db8:1:0:a00:ff:fedc:a%eth0

Network	Next-Hop	Link	Metric
fe80::/64	::	eth0	50
fe80::a00:ff:fedc:a/128	::	lo	50

Table 6: Routing Table for fe80::a00:ff:fedc:a%eth0

Network	Next-Hop	Link	Metric
2001:db8:2::/64	::	eth1	40
2001:db8:2:0:200:ff:fec4:b/128	::	lo	50
::/0	fe80::ce	eth1	40

Table 7: Routing Table for 2001:db8:2:0:200:ff:fec4:b%eth1

Network	Next-Hop	Link	Metric
fe80::/64	::	eth1	50
fe80::200:ff:fec4:b/128	::	lo	50

Table 8: Routing Table for fe80::200:ff:fec4:b%eth1

Author's Address

TOC

	Aleksi Suhonen
	Tampere University of Technology, Finland
	Korkeakoulunkatu 1
	Tampere 33720
	FI
Phone:	+358 45 670 2048
Email:	i-d-2009@ssd.axu.tm