6man	A. Suhonen	
Internet-Draft	Tampere University of Technology,	
Updates: 3484 (if approved)	Finland	
Intended status: Experimental	July 12, 2010	
Expires: January 13, 2011		

TOC

Address Selection Using Source Address Specific Routing Tables
draft-axu-addr-sel-01

Abstract

RFC 3484 defines two algorithms for default source and destination address selection, but it has several shortcomings. This document specifies an alternate address selection algorithm that uses routing tables as policy input.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.
Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.
Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as “work in progress.”
This Internet-Draft will expire on January 13, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.
This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

§ Author's Address

---

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 for IPv4 in relation to IPv6. 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., “Stream Control Transmission Protocol,” September 2007.) [RFC4960], for example, finds that its active source destination address pair is no longer functional, it will need to start searching for a new one. If the multipath protocol doesn't respect address selection policy, it may cause similar security incidents as the old address selection algorithm. A multipath protocol should also consult the algorithm during the session and not only on connection setup. SHIM6 (Nordmark, E. and M. Bagnulo, “Shim6: Level 3 Multihoming Shim Protocol for IPv6,” June 2009.) [RFC5533] has similar concerns.

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.

This algorithm always performs address selection on source-destination address pairs with additional information such as next hop attached. Preferences are also associated with address pairs instead of single addresses.

1.1. Terminology

TOC

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 [RFC2119] (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.).

An autoconfiguration method is any process or protocol that acquires or creates an IP address for a link. Examples of autoconfiguration methods include RARP (Finlayson, R., Mann, T., Mogul, J., and M. Theimer, "Reverse Address Resolution Protocol," June 1984.) [RFC0903], DHCPv4 (Droms, R., "Dynamic Host Configuration Protocol," March 1997.) [RFC2131], ICMPv6 RA (Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," September 2007.) [RFC4862], DHCPv6 (Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," July 2003.) [RFC3315], Teredo (Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)," February 2006.) [RFC4380], 6to4 (Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds," February 2001.) [RFC3056], ISATAP (Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)," March 2008.) [RFC5214], IPv4 Link Local (Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses," May 2005.) [RFC3927], PPP (Simpson, W., "The Point-to-Point Protocol (PPP)," July 1994.) [RFC1661] and mDNS.
An autoconfiguration agent is a process or daemon that executes an autoconfiguration method on the host.
Autoconfiguration methods may also produce other information such as default routes and DNS resolvers. The complete collection of information produced by an autoconfiguration method is called an autoconfiguration blob in this document.

---

2. Filter Algorithm

TOC

When a host has several addresses, they SHOULD each be associated with their own routing tables. The first stage in selecting source and destination addresses SHOULD be 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. If more than one route matches, the most specific route SHOULD be checked for validity.
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.

2.1. Scope Recognition

TOC

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. Whenever scopes are mentioned in this draft, they are always mentioned in the context of generating policy input for the algorithm.

---

3. Precedences and Labels

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 after the filtering stage according to preference. Higher precedence values have higher preference. In effect, the precedence is for the address pair, not for a single address.
When two or more address pairs have the same precedence, their destination prefix lengths MUST be compared and the longer prefixes MUST be considered more preferable.
When two or more address pairs are still equal, their destination metrics in the routing table MUST be compared and address pairs with better metrics MUST be considered more preferable.
If there are still ties, they MAY be broken by some Equal Cost Multi-Path load sharing techniques. Otherwise the algorithm SHOULD output the equal address pairs in the same order as they appeared in its input.
The label abstraction used by the original RFC (Draves, R., “Default Address Selection for Internet Protocol version 6 (IPv6),” February 2003.) [RFC3484] loosely corresponds to the routing table abstraction in this algorithm. If a database table join is performed on the source address policy table and the destination address policy table as defined by [RFC3484] (Draves, R., “Default Address Selection for Internet Protocol version 6 (IPv6),” February 2003.) on the label field and the precedences are added together, the result resembles the policy tables used by this algorithm.
This 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.

TOC

4. Routing Table and Address Properties

FIXME: come up with a better section title
This section details how source address specific routing tables should be populated to be usable as input data for this algorithm.

4.1. Local Scope

TOC

The routing tables associated with localhost addresses (127.0.0.1 and ::1) SHOULD only have routes to localhost address space. (127.0.0.0/8 and ::1/128) In addition, if the host supports node local multicast, a route for the node local scope multicast space MAY also appear in this table. (e.g. ff01::/16, ff11::/16)
The routing tables associated with any other addresses assigned to the host SHOULD have a host route for the address itself pointing to the loopback interface.
The default precedence for all local scope route entries SHOULD be 500.

4.2. Link Local Scope

TOC

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 link local scope multicast space MAY also appear in this table. (e.g. ff02::/16, ff12::/16)
This means that the link local source address is usable only with other link local destination addresses on the same link.
The default precedence for all link local scope route entries SHOULD be 500.

4.3. Autoconfiguration for Global Scope

TOC

When global scope addresses are assigned to interfaces dynamically through stateless or stateful autoconfiguration the process MUST yield 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. (e.g. ff0e::/16, ff1e::/16)
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.
The default precedence for all global scope route entries SHOULD be 400.
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 150.
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.
Another example of automatically generated extra routes is fc00::/7 for ULA addresses as defined by [RFC4193] (Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses," October 2005.). However, the routing table for a ULA should only have a route for the address space it's usable in, and that route should be more specific than the default routes for globally usable source addresses, it should win according to the longest matching prefix rule.

4.4. Limited Scope

TOC

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 limited scope addresses to global addresses, the routing tables associated with limited scope addresses MAY have the default route.
This algorithm effectively treats all addresses that aren't associated with a default route as limited scope. The system MAY automatically recognize addresses within the ULA prefix fc00::/7 [RFC4193] (Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses," October 2005.) and treat them as limited scope.
The default precedence for limited scope addresses SHOULD be the same as global scope addresses (400), but it SHOULD be simple for the system or network administration to change this setting.

In addition, if the host and network support multicast, a route for the site local scope multicast space MAY also appear in this table. (e.g. ff05::/16, ff15::/16)

4.5. Transition Technique Routing Tables

TOC

The default precedence for all route entries for source addresses generated through transition techniques SHOULD be 300.
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 350. This is to make using the same transition technique source address more preferable than some different transition technique.
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.

4.6. IPv4 Compatible Routing Tables

TOC

The precedence for all IPv4 compatible global scope route entries SHOULD be easily configurable. The default precedence should be 400. If administration wishes to promote the use of IPv6, then the IPv4 entries should have a precedence of 200.

4.7. Mobility

TOC

The precedence of route entries in the tables for home addresses and care-of addresses SHOULD be easily configurable. The default precedence for home addresses should be 425 and for care-of addresses it should be 400. If an address is simultaneously a home address and a care-of address, then the precedence should be 450. When the host is ''at one home'', that address will be used, and when the host is visiting ''at the other home'', the home address of that other home will be preferred.

4.8. Reachability Information

TOC

If the next-hop information associated with a route in any table has been found unreachable or the link is down the precedence of the

affected routes MAY be temporarily dropped to zero until they work again.

4.9. 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 autoconfiguration agent, the host SHOULD also always use the default route received from the same autoconfiguration agent.
This algorithm supports those efforts by making it possible to restrict a process to one routing table for both address resolution and selection.

4.10. 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.

4.11. 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 protocols can be used to convey address selection configuration information for this algorithm.
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.

---

5. 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.

5.1. Source Address Selection Rules

TOC

5.1.1. Prefer Same Address

TOC

The interface address is added with a higher metric to its address specific routing table than any other routes. This ensures that this algorithm conforms to this rule.

5.1.2. Prefer Appropriate Scope

TOC

The routes of different scopes are assigned different precedences. They correspond to the scope values of the original algorithm.

TOC

5.1.3. Avoid Deprecated Addresses

If precedences for deprecated addresses are zeroed, they should automatically be depreferred against any other addresses.

5.1.4. Prefer Home Address

TOC

The higher precedences assigned to home addresses make them always preferable when compared to almost everything else, apart from link local addresses. If a host has two home addresses in different networks, the rules presented will make it prefer the correct address depending on the current location of the mobile node.

5.1.5. Prefer Outgoing Interface

TOC

The metric for the route on the table for the address on the outgoing interface should be better than on other interfaces, so all else being equal, it should break the tie to ensure that this rule is met.

5.1.6. Prefer Matching Label

TOC

This algorithm doesn't have labels at all. However, automatically added depreference routes will take care of this rule.

5.1.7. Prefer Public Addresses

TOC

The section on temporary address selection (Temporary Address Selection) already deals with this rule.

5.1.8. Use Longest Matching Prefix

TOC

This is a debated rule so reproducing it is also questionable.

TOC

5.2. Destination Address Selection Rules

5.2.1. Avoid Unusable Destinations

TOC

This algorithm conforms to this rule by design.

5.2.2. Prefer Matching Scope

TOC

The routes of different scopes are assigned different precedences. The routes of matching scopes are assigned higher precedences than routes of differing scopes.

5.2.3. Avoid Deprecated Addresses

TOC

If precedences for deprecated addresses are zeroed, they should automatically be depreferred against any other addresses.

5.2.4. Prefer Home Address

TOC

The higher precedences assigned to home addresses make them always preferable when compared to almost everything else, apart from link local addresses. If a host has two home addresses in different networks, the rules presented will make it prefer the correct address depending on the current

---

5.2.5. Prefer Matching Label

TOC

This algorithm doesn't have labels at all. However, automatically added depreference routes will take care of this rule.

TOC

5.2.6. Prefer Higher Precedence

This algorithm primarily compares precedence values only. All the rules above are encoded as precedence values into the routing tables.

5.2.7. Prefer Native Transport

TOC

This rule essentially is a special case of the matching scope and matching label rules. The special routing table generation rules for transition mechanisms make sure that this algorithm behaves the same way as the original algorithm.

5.2.8. Prefer Smaller Scope

TOC

The routes of different scopes are assigned different precedences. They correspond to the scope values of the original algorithm.

5.2.9. Use Longest Matching Prefix

TOC

This is a debated rule so reproducing it is also questionable.

5.2.10. Leave Order Unchanged

TOC

This new algorithm conforms to this rule.

---

6. 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.

TOC

6.1. Multiple Routers on a Single Interface

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.

6.2. Ingress Filtering Problem

TOC

This algorithm will always choose the correct link and next-hop address for each source address. However, if several source addresses share the same next-hop on the same link, then there's nothing the algorithm can do. The problem has to be fixed inside the router announcing the prefixes.

6.3. 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.
The default assumption is that the auto-configuration method supplies a default route for all globally usable addresses. The routing tables of source addresses usable only within a closed network SHOULD NOT have a default route. They SHOULD only have routes to the networks they are usable within.
System or network administration MUST specify allowed or disallowed connections by modifying the auto-configuration input or the routing tables.

6.4. 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 limited scope 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 auto-configuration input or the routing tables.

6.5. 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.

6.6. Multicast Source Address Selection

TOC

Multicast address selection works the same way as unicast address selection. The source address candidate routing tables SHOULD have only the appropriate multicast scope routes.

6.7. Temporary Address Selection

TOC

A temporary addresses MAY be associated with routing tables of its own, instead of sharing a routing table with the address used to generate the temporary address.
The precedences for the table for a temporary address would be lower than that of a similar but 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.

6.8. IPv4 or IPv6 Prioritization

TOC

This is a configuration issue with the routing tables. The algorithm itself doesn't dictate policy for sites.

6.9. ULA and IPv4 Dual-Stack Environment

TOC

This special case is easily handled by omitting the default route from the routing table for ULA addresses. This would result in site-external

destination IPv6 addresses not having any usable source addresses and thus they would never be considered by this algorithm.

6.10. ULA or Global Prioritization

TOC

Already covered in Section 6.4 (Combined Use of Global and ULA).

---

7. 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.

7.1. 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 6 (RFC5220 Concerns).

7.2. 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. These other methods include auto-configuration and routing protocols.
Once the routing table is updated, the address selection algorithm will start making decisions based on the new information.

7.3. Dynamic Behavior Update

TOC

From the point of view of this algorithm, this problem is a feature of auto-configuration methods. If the autoconfiguration methods rewrite

routing tables, the address selection algorithm will always use the updated information when it's invoked.

7.4. 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.

7.5. Application-Specific Behavior

TOC

Additional filter constraints from Section 4.9 (Additional Filter Constraints) can be used to influence address selection per application.

7.6. Multiple Interface

TOC

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.

7.7. 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.

7.8. 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.

7.9. Compatibility with RFC 3493

TOC

FIXME 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.

7.10. 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. One key difference between this algorithm and the [RFC3484] (Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)," February 2003.) algorithm is that this algorithm considers all valid source address candidates, where as the original algorithm chooses only one source address per every destination address. This difference can be easily overcome by an extra filter rule, that accepts only the highest precedence source-destination address pair for every given destination address.

7.11. 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 11.2 (RFC5221 Requirements).

8. 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.

8.1. 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.

8.2. 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.

8.3. 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.

8.4. 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.

---

9. 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, Tim Chown, Brian E Carpenter, .

---

10. IANA Considerations

TOC

This document has no IANA Actions.

---

11. Security Considerations

TOC

11.1. 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.

11.2. 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.

11.2.1. 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.

11.2.2. 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 ensure integrity, authentication and authorization.

---

12. References

TOC

12.1. 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).
[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).
[RFC4960]	Stewart, R., “Stream Control Transmission Protocol,” RFC 4960, September 2007 (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).

12.2. 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).
[RFC0903]	Finlayson, R., Mann, T., Mogul, J., and M. Theimer, “Reverse Address Resolution Protocol,” STD 38, RFC 903, June 1984 (TXT).
[RFC1661]	Simpson, W., “The Point-to-Point Protocol (PPP),” STD 51, RFC 1661, July 1994 (TXT).
[RFC2131]	Droms, R., “Dynamic Host Configuration Protocol,” RFC 2131, March 1997 (TXT, HTML, XML).
[RFC3056]	Carpenter, B. and K. Moore, “Connection of IPv6 Domains via IPv4 Clouds,” RFC 3056, February 2001 (TXT).
[RFC3315]	Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, “Dynamic Host Configuration Protocol for IPv6 (DHCPv6),” RFC 3315, July 2003 (TXT).
[RFC3927]	Cheshire, S., Aboba, B., and E. Guttman, “Dynamic Configuration of IPv4 Link-Local Addresses,” RFC 3927, May 2005 (TXT).
[RFC4193]	Hinden, R. and B. Haberman, “Unique Local IPv6 Unicast Addresses,” RFC 4193, October 2005 (TXT).

[RFC4380]	Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)," RFC 4380, February 2006 (TXT).
[RFC4862]	Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," RFC 4862, September 2007 (TXT).
[RFC5014]	Nordmark, E., Chakrabarti, S., and J. Laganier, "IPv6 Socket API for Source Address Selection," RFC 5014, September 2007 (TXT).
[RFC5214]	Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)," RFC 5214, March 2008 (TXT).
[RFC5533]	Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6," RFC 5533, June 2009 (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	Prec
::/0	::	lo	-1 !U
::1/128	::	lo	500

Table 2: Routing Table for ::1

Network	Next-Hop	Link	Prec
2001::/32	::	teredo	350
::/0	::	teredo	300
2001:0:c200:201::3/128	::	lo	500

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

Network	Next-Hop	Link	Prec
fe80::/64	::	teredo	500
fe80::ffff:ffff:ffff/128	::	lo	500

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

Network	Next-Hop	Link	Prec
2001:db8:1::/64	::	eth0	400
::/0	fe80::13	eth0	400
2001:db8:1:0:a00:ff:fedc:a/128	::	lo	500

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

Network	Next-Hop	Link	Prec
fe80::/64	::	eth0	500
fe80::a00:ff:fedc:a/128	::	lo	500

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

Network	Next-Hop	Link	Prec
2001:db8:2::/64	::	eth1	400

2001:db8:2:0:200:ff:fec4:b/128	::	lo	500
::/0	fe80::ce	eth1	400

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

Network	Next-Hop	Link	Prec
fe80::/64	::	eth1	500
fe80::200:ff:fec4:b/128	::	lo	500

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-2010@ssd.axu.tm