IPv6 Maintenance Working Group
Internet-Draft
Intended status: Informational
Expires: September 9, 2010

A. Matsumoto
T. Fujisaki
NTT
R. Hiromi
Intec Netcore
March 8, 2010

Solution approaches for address-selection problems
draft-ietf-6man-addr-select-sol-03.txt

Abstract

In response to address selection problem statement and requirement documents, this document describes solution approaches and evaluates proposed solution mechanisms in line with requirements. It also examines the applicability for each solution mechanism, and concludes that no single solution solves the problems and the combination of the solutions should be the way forward.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on September 9, 2010.

Table of Contents

1. Introduction

Today, the RFC 3484 mechanism is widely implemented in major OSs. However, many people, including us, have found that in many sites the default address selection rules do not work perfectly. RFC 5220 [RFC5220] lists problematic cases that resulted from incorrect address selection. RFC 5221 [RFC5221] enumerates requirements for address-selection mechanisms.

In the IETF mailing lists and in the internet-draft archives, some mechanisms for solving address-selection problems were proposed. This document describes possible design approaches for solving address selection problems. After summarizing each existing proposal mechanism and analyzing how well the method corresponds with the requirements, applicability domain for each mechanism are evaluated.

2. Solution Approaches

There are two types of approaches that can control the behavior of hosts in terms of the selection of destination address and source address. The first type is proactive, where the host is given the necessary information to decide the destination and source addresses before the beginning of data transmission. The other type is reactive, where the host decides appropriate destination address and source addresses through trial and error.

2.1. Proactive approach

In this approach, hosts should have all or a part of the information for selecting appropriate address pair before the actual data transmission.

2.2. Reactive approach

In this approach, the host does not have initial information for address selection. It will try using different pairs of destination and source addresses until the connection is established. It can take advantage of error messages, and can make use of it to prune unworkable pair of addresses.

3. Solution Mechanisms

This section describes the evaluation of the four mechanisms. The evaluation value has a 3-point scale for each of 8 requirements in the requirement document. The meanings of the points are as follows.

1 : bad
2 : fair
3 : good

About "Effectiveness", the score is 1 if the approach solves no problematic cases described in the problem statement document, 2 if it can handle at least one, and 3 if it solves every case.

3.1. Policy Distribution (Proactive Approach)

In this approach, a host obtains everything, usually policies, needed to select addresses prior to communication. The address selection design team is working on this approach. [I-D.ietf-6man-addr-select-considerations] DHCPv6 and RA are possible transport protocols. There is a proposal document [I-D.fujisaki-dhc-addr-select-opt] in which DHCPv6 is used for this purpose.

This approach can take advantage of the RFC 3484 Policy Table, which is already widely deployed. By distributing policies for the Policy Table, you can auto-configure a host's address selection behavior.

3.1.1. Requirements correspondence analysis

1. Effectiveness: 3
   It can support all cases by using the policy table.

2. Timing: 3
   All information for communication is in a host in advance. Communication starts at once when it is necessary and the communication process refers to local policy information, which does not cause any delay or incorrect address selection.

3. Dynamic update: 3
   Though it depends on what protocol is used to distribute the policies, some mechanisms support information updates from the server, such as RA and DHCPv6 RECONFIGURE.

4. Node-specific behavior: 3
   For distribution to individual hosts in the same segment, DHCPv6 and unicast based RA can be used.

5. Application-specific behavior: 2
   The policy table itself doesn't support application-specific address selection. It can be done using the address selection API. [RFC5014]

6. Multiple interfaces: 2

If all interfaces belong to the same administration domain, it is possible for the address-selection information to be controlled by administrators of that domain. However, if not, routing information and address selection policies are not always equivalent between domains. For such cases, the algorithm for merging policies have to be considered, which is documented in this document. [I-D.arifumi-6man-addr-select-conflict]

7. Central control: 3
It can support central control. A site administrator or a service provider can determine users' policy tables.

8. Route selection: 2
Current solutions, such as DHCPv6 and RA, do not have a mechanism for cooperation with routing protocols. This could be done with other techniques such as "source address based routing" or "Default Router Preferences and More-Specific Routes" RFC 4191. [RFC4191]

9. Compatibility with RFC 3493: 3
This approach is able to coexist with any kind of applications (socket API). In detail, any types of function such as getaddrinfo(), getsockname(), connect() or other typical system calls will work without alterations if this mechanism is applied to a host.

10. Compatibility and Interoperability with RFC 3484: 3
The basic idea of this approach has a compatibility with RFC3484. This approach make RFC3484 policy table configurable to put some hints related with it's individual network case.

11. Security: 2
By injecting address selection policy, a session hijacking can be possible in this mechanism. A combination of Layer 2 securing techniques and this mechanism will be able to be effective against security concerns. DHCP and RA protocol have own security measures and they also protect from them.

3.1.2. Other issues

None.

3.2. Routing system assistance (Proactive Approach)

Fred Baker proposed this approach. A host asks the DMZ routers or the local router which is the best pair of source and destination addresses when the host has a set of addresses A and the destination host has a set of addresses B. Then, the host uses the policy

provided by the server/routing system as a guide in applying the response.

3.2.1. Requirements correspondence analysis

1. Effectiveness: 3
   A routing system knows about information about paths toward the destination and information about which of their prefixes should be used. Therefore, it is possible to select an appropriate pair of source and destination addresses.

2. Timing: 3
   A routing system always has up-to-date routing information, so it will be possible to provide suitable information whenever requests come.

3. Dynamic update: 3
   A routing system always has up-to-date routing information, and it will be possible to provide suitable information whenever requests come.

4. Node-specific behavior: 3
   Node-specific information can be provided if a server recognizes individual nodes.

5. Application-specific behavior: 2
   A routing system does not care about applications. Using address selection API allows nodes to behave in an application-specific way.

6. Multiple Interfaces: 2
   The same as the policy based mechanism.

7. Central Control: 3
   It is possible to provide address selection information from one source.

8. Route Selection: 3
   It is possible to give next-hop selection advice to a host. As routers have routing information, it would seem to be easier for routers to implement this function.

9. Compatibility with RFC 3493: 1
   In the existing TCP/IP protocol stack implementation, destination address selection is mainly the role of the application and not that of the kernel unlike source address selection. Therefore, implementing this model without affecting applications is not feasible.

10. Compatibility and Interoperability with RFC 3484: 2<
Currently it just proposed and there is no implementation.
Therefore, it depends on how to implement with this requirement,
but it can be coexist with RFC3484.

11. Security: 2

This approach has a weakness on hijacking just like the previous mechanism.

3.2.2. Other issues

- It has a possible scalability problem. A host must consult the routing system every time it starts a connection if the host does not have address selection information for the destination host or if the information lifetime has expired.

- The existing host/router OS implementation must be changed a lot. This can be problematic in some cases.

3.3. Trial-and-error based (Reactive Approach)

M. Bagnulo presented a new address selection idea in his draft. Hirotaka Matsuoka extended and elaborated this approach in his draft. [I-D.matsuoka-multihoming-try-and-error] When the host notices that a network failure has occurred or packets have been dropped somewhere in the network by, for example, an ingress filter, the host stores a negative cache of address selection.

Hosts may use some kinds of error messages, e.g, ICMP error messages, from a network to detect that sent packets did not reach the destination quickly.

The host stores a cache of address selection information so that the host can select an appropriate source address for new connections.

3.3.1. Requirement correspondence analysis

1. Effectiveness: 2
This solution is not effective for the destination address selection cases, such as a problem about IPv4 or IPv6 prioritization described in the problem statement document.

2. Timing: 1

Hosts should try to use all the available source addresses to the maximum to find an appropriate source address. If the host tries the next source address after the previous trial using another source address has failed, it may take a long time because this trial-and-error process lasts until the connection succeeds. If the host does not use an error message from a network to detect a connection error, it takes longer to wait for a time-out. This issue largely depends on the cacue functionality, but there is no established algorithm for this.

3. Dynamic update: 3
   If hosts detect a connection failure using some reliable mechanism, such like TCP or ICMP error messages, a connection failure caused by some changes in the network will be detected immediately by the hosts.

4. Node-specific behavior: 2
   This solution does not have a function for node-specific behavior. However, it is not impossible to implement by deploying a packet filter configured for each node at the gateways through which the packets from nodes pass.

5. Application-specific behavior: 2
   This solution does not have a function for application-specific behavior. However, the mechanism of this approach does not exclude address selection by each application.

6. Multiple interfaces: 3
   If the protocol-stack or an application supports interface selection and it tries to establish a connection by changing addresses and also interfaces, it can find a working combination of addresses and interface.

7. Central control: 2
   The only way that a central administrator has to control the node behavior is switching a filter on/off on the network. Therefore, advanced control such as traffic engineering and QoS is almost impossible.

8. Route Selection: 2
   This solution does not refer to next-hop selection for the transmission of a packet. So, it should be used with some routing function such as RFC 4191 on the nodes.

9. Compatibility with RFC 3493: 1

This approach has possibility to interfere with coexistence with applications(socket APIs). The returen value of functions would be changed for its meaning. A case is suspected that the return value of connect() system call will change its state from "non-blocking" to "blocking" and this will bring alteration to the application behaviours. Because of this suspicion, this approach scores 1.

10. Compatibility and Interoperability with RFC 3484: 2
It depends on how to implement with this requirement. But there will be possible conflict which a result will be overwrite the 3484 policy table without any permission of domain administrators.

11. Security: 1
This approach has a weakness on hijacking. Moreover, it may expose the privacy of the user host by showing all or a part of the addresses of the host to the destination host.

3.3.2. Other issues

- A host must learn address selection information for each destination host. Therefore, the number of cache entries could be very large.

- The existing host/router OS implementation must be changed a lot. In particular, changing the source address of the existing connection is not so easy and has a big impact on the existing TCP/IP protocol stack implementation.

3.4. All at once (Reactive Approach)

This is another proposal by Fred Baker at 6man ML. By trying all the pairs of source and destination addresses at once, or in a very shord period of time, a connection can be established as far as there is at least one working pair of addresses.

This mechiansm can easily combined with the error message retrieval and cache function discussed in Section 3.3 to prune the unnecessary connection trials.

3.4.1. Requirement correspondence analysis

1. Effectiveness: 2
This solution is not effective for the destination address selection cases, such as a problem about IPv4 or IPv6 prioritization described in the problem statement document.

2. Timing: 3

The user does not have to wait for extra period of time.

3. Dynamic update: 3
It can reflect dynamically changing network, as far as it always tries all possible addresses and next-hops.

4. Node-specific behavior: 2
This solution does not have a function for node-specific behavior. However, it is not impossible to implement by setting a packet filter for each node on the gateways through which the packets from nodes pass.

5. Application-specific behavior: 2
This solution does not have a function for application-specific behavior. However, the mechanism of this approach does not exclude address selection by each application.

6. Multiple interfaces: 3
If the protocol-stack supports interface selection and it tries to establish a connection by changing addresses and also interfaces, it can find a working combination of addresses and interface. But, in such a environment, a host has to try all the combinations of source address, destination address, and next-hop addresses, which can be huge amount of connection establishment trial.

7. Central control: 2
The only way that a central administrator has to control the node behavior is switching a filter on/off on the network. Therefore, advanced control such as traffic engineering and QoS is almost impossible.

8. Route Selection: 2
This solution does not refer to next-hop selection for the transmission of a packet. Therefore, it should be used with some routing function such as RFC 4191 on the nodes.

9. Compatibility with RFC 3493: 1
For non-connected transport protocols, there is no universal nor established way of telling a connection is successful or failure from the viewpoint of anything other than the application. In this sense, it does not work for non-connected transport protocols.

10. Compatibility and Interoperability with RFC 3484: 1

This mechanism uses every possible addresses for its rather initial connection setup. In that sense, it does not follow the address selection rules of RFC 3484.

11. Security: 1
This approach has a weakness on hijacking. Moreover, it may expose the privacy of the user host by showing all or a part of the addresses of the host to the destination host.

3.4.2. Other issues

- It brings unnecessary traffic to network, host, routers, and destinations.

- It breaks DNS round robin based load balancing.

- A host must learn address selection information for each destination host. Therefore, the number of cache entries can be very large.

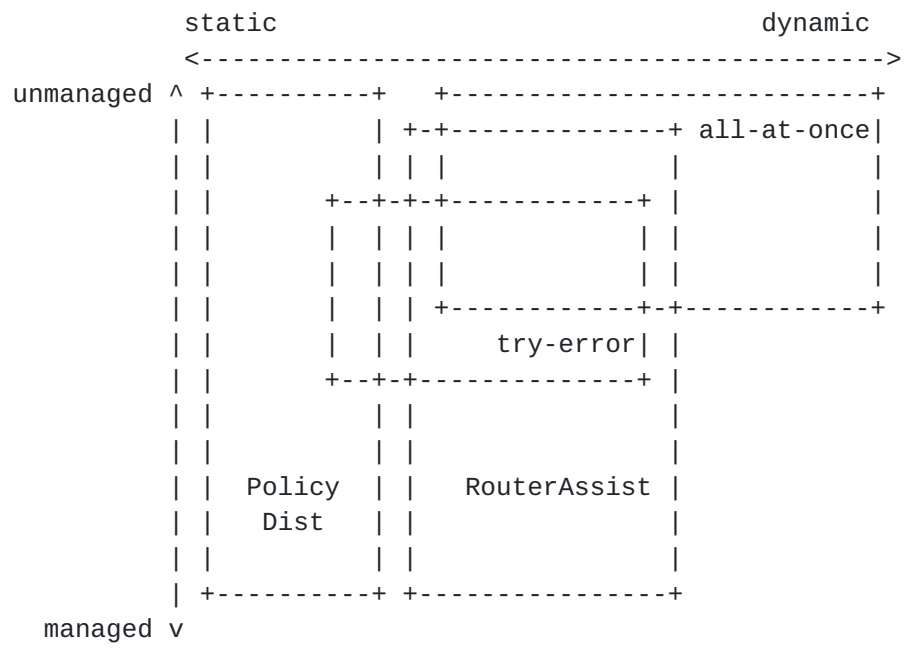
- The existing host OS implementation must be changed significantly.

4. Applicability Comparison

If you actually want to choose one mechanism to solve your address selection problem, it is important to figure out which approach is best suited to your situation. This section tries to evaluate the applicability of each approach from several aspects.

4.1. Dynamic-static and managed-unmanaged

We use two axes to evaluate the applicability of the four approaches. One axis shows whether or not the network structure changes dynamically and the other axis shows whether the site is managed or unmanaged. In a managed network, by our definition, a network administrator manages his or her network, routers, and hosts. For example, an enterprise network is managed, whereas a home network and a SOHO network are unmanaged.



PolicyDist:

- In a dynamic site, the policy table must be updated accordingly and traffic for policy table distribution increases.

try-and-error:

- This is a slightly manageable than all-at-once in that it does not change the paths of established connections dynamically.

- In a very dynamic site, the use of an address selection information cache does not have a good effect. This results in connection failure and may degrade usability badly.

- Even in a very static site, a host may try inappropriate addresses or next-hops and experience connection failures.

RouterAssist:

- A host must send at least as many queries as the number destination hosts. Therefore, in a static site, this method is not optimal.

- In a very dynamic site, address selection information cache is no help. If the cache function is not used, then connection failures do not occur.

all-at-once:

- In a static site, all-at-once is not desirable because of its connection sequence overhead and timeout-wait for path exploration.

- In a managed site, it is not easy to manage in terms of node-specific address selection control and central control.

5. Security Considerations

Incorrect address selection can lead to serious security problems, such as session hijacking. However, we should note that address-selection is ultimately decided by nodes and their users. There are no means to enforce a specific address-selection behavior upon every end-host from outside the host. Therefore, a network administrator must take countermeasures against unexpected address selection.

6. IANA Considerations

This document has no actions for IANA.

7. Conclusions

In this document, we examined solutions to address selection problems in the IPv6 multi-prefix environment. From the requirement analysis and applicability evaluation, it can be concluded that there is no one perfect solution for this series of problems.

Hence, we have to combine two or more solutions together. The design team proposal and all-at-once proposal are extreme opposite, and combining these two seems to make the coverage biggest problem spaces.

8. References

8.1. Normative References

[RFC3484] Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)", RFC 3484, February 2003.

[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.

[RFC5221] Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, "Requirements for Address Selection Mechanisms", RFC 5221, July 2008.

8.2. Informative References

[I-D.arifumi-6man-addr-select-conflict]
Matsumoto, A., Fujisaki, T., and R. Hiromi, "Considerations of address selection policy conflicts", draft-arifumi-6man-addr-select-conflict-01 (work in progress), October 2009.

[I-D.axu-addr-sel]
Suhonen, A., "Address Selection Using Source Address Specific Routing Tables", draft-axu-addr-sel-00 (work in progress), July 2009.

[I-D.fujisaki-dhc-addr-select-opt]
Fujisaki, T., Matsumoto, A., and R. Hiromi, "Distributing Address Selection Policy using DHCPv6", draft-fujisaki-dhc-addr-select-opt-09 (work in progress), March 2010.

[I-D.ietf-6man-addr-select-considerations]
Chown, T., "Considerations for IPv6 Address Selection Policy Changes", draft-ietf-6man-addr-select-considerations-00 (work in progress), October 2009.

[I-D.ietf-shim6-multihome-shim-api]
Komu, M., Bagnulo, M., Slavov, K., and S. Sugimoto, "Socket Application Program Interface (API) for Multihoming Shim", draft-ietf-shim6-multihome-shim-api-13 (work in progress), February 2010.

[I-D.matsuoka-multihoming-try-and-error]
Matsuoka, H., "A Try and Error type approach for multihoming", draft-matsuoka-multihoming-try-and-error-00 (work in progress), April 2009.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, November 2005.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC5014] Nordmark, E., Chakrabarti, S., and J. Laganier, "IPv6 Socket API for Source Address Selection", RFC 5014,

September 2007.

[RFC5533] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, June 2009.

[RFC5534] Arkko, J. and I. van Beijnum, "Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming", RFC 5534, June 2009.

Appendix A. Other Solutions

Other than policy table distribution based approach, Aleksi Suhonen proposed his idea [I-D.axu-addr-sel], where a host has a separate routing table for each attached address. The document is still incompleted, but basically it should have characteristics in common with above mentioned policy table based mechanism except for the implementation characteristics.

shim6 [RFC5533] was designed for site-multihoming. This mechanism introduces a new sub-layer in IP layer, and new address selection method for session initiation and session survivability. it is documented in RFC 5534. [RFC5534] shim6 should fall into try-and-error based category, in that it establishes a connection by trying pairs of addresses.

Appendix B. Revision History

03:
- Combined shim6 and try-and-error mechanisms into one section 3.3.
- Removed deployment analysis in Section 4.
- Made the introduction short and brief.
- Added a reference to Design Team's work.
- Added a reference to address selection conflict draft.
- Created Appendix A. to mention about other solution proposals.
- Conclusion and abstract was modified to reflect the discussion at 6man ML.

02:
- Updated references for documents that were approved as RFCs.
- Added reference to Hirotaka Matsuoka's try-and-error mechanism.
- Added description about Aleksi Suhonen's routing table based mechanism.

01:

Corresponding to the increase of RFC 5221 requirements, considerations about requirement #9, #10, #11 are added for each approach.

00:
Approved as a 6man working group item.

Authors' Addresses

Arifumi Matsumoto
NTT PF Lab
Midori-Cho 3-9-11
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 3334
Email: arifumi@nttv6.net

Tomohiro Fujisaki
NTT PF Lab
Midori-Cho 3-9-11
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 7351
Email: fujisaki@syce.net

Ruri Hiromi
Intec Netcore, Inc.
Shinsuna 1-3-3
Koto-ku, Tokyo 136-0075
Japan

Phone: +81 3 5665 5069
Email: hiromi@inetcore.com