MIF Working Group
Internet-Draft
Intended status: Informational
Expires: January 14, 2011

J. Laganier
Qualcomm Inc.
G. Montenegro
Microsoft
J. Korhonen
Nokia Siemens Networks
T. Savolainen
Nokia
Z. Cao
China Mobile

July 13, 2010

MIF Current Practice Analysis
draft-cao-mif-analysis-01

Abstract

This document analyzes whether the problems encountered by a multi-homed host are satisfactorily addressed by mechanisms currently implemented in operating systems.

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 14, 2011.

Copyright Notice

Table of Contents

1. Introduction

A multihomed host have multiple provisioning domains via virtual and/or physical interfaces. A multihomed host receives node configuration information from each of its access networks, through various mechanisms such as DHCP, PPP's IPCP and IPv6 Router Advertisements. When the multihomed host receives various configuration objects (e.g., DNS server address, default gateway, address selection policies) with values that differ from one administrative domain to another, the node has to decide which one to use or how to reconcile them.

Issues regarding how the multi-homed host uses the configuration objects have been addressed in [I-D.ietf-mif-problem-statement]. Current practices of how the various implementations handle these problems are introduced in [I-D.ietf-mif-current-practices]. This document analyzes whether the problems encountered by a multi-homed host are satisfactorily addressed by mechanisms implemented in operating systems.

2. Terminology

The following terms are used throughout this document:

Multihomed Host: A host that is attached to one or more networks via one or more virtual and/or physical interfaces.

3. Problem Analysis

We group the problems raised in [I-D.ietf-mif-problem-statement] into specific categories as per the subsections below.

3.1. Naming and Addressing

1. The operating systems has node-scoped DNS server addresses but the DNS server addresses provided by a given domain are only reachable through that domain.

2. The answers to DNS queries returned by the DNS server of a given domain are only valid and/or reachable within that domain (e.g., split horizon DNS) but the operating system treats these answers as valid on any domain.

3. Private IPv4 addresses [RFC1918] and Unique Local IPv6 Unicast Addresses [RFC4193] are reachable from within a given domain (i.e., they are site-scoped) but the operating system doesn’t know the domain boundary and treats these as reachable on any domain (i.e., they have global scope.)

4. Private IPv4 addresses [RFC1918] are only unambiguous within a given domain but the operating system doesn’t know the domain boundary and cannot associate a Private IPv4 Address to a given domain and thus treats those as valid on any domain.

3.2. Routing

1. Routing tables entries to ambiguous subnet prefixes in [RFC1918] addressing space are only unambiguous within a given domain but the operating system doesn’t distinguish routes to the same prefixes belonging to different communication domains, thus leading to use of the wrong outbound interface and wrong destincation gateway.

2. Routing tables entries with an ambiguous next hop IP address in [RFC1918] addressing space are only to be used within a given domain but the operating system doesn’t necessarily know which was the communication domain thus leading to use of the wrong outbound interface and wrong destination gateway, and/or communication failure if no destination gateway is reachable at the destination address or if the destination gateway has no upstream route to the final destination of the packet.

3. Host implementations usually do not implement the [RFC1122] model where the Type-of-Service are in the routing table which could be use to choose between routes with same longest prefix match and

same metrics but different Type-of-Service characteristics, e.g., low delay, high throughput.

3.3. Reachability

1. Ingress filtering can prevent communication when a node sends packets from a source address allocated from a given domain to a (default) router in another domain.

2. Strong host model implementaion can cause incoming packets to be discarded when they are sent to a destination address assigned to one of the interface of the node that is not the interface on which the packet is incoming.

3. There is no interface between a router and a host for the router to indicate that there is no default route but only specific routes to some prefixes. As a result, a node that discovers a router assumes that any destination is reachable, which might not always be the case: in some case only connectivity to destination in the domain is available, and other destinations are unreachable, e.g., walled gardens, corporate intranets, etc.

3.4. Domain Selection

1. Application usually does not specify to which domain they want to communicate. When the destination has an unambiguous address the domain can sometimes be derived from that. This is however not the case when the destination is an ambiguous address from [RFC1918].

2. Some applications require domain affinity. There should be some way to set it either by the application itself or by the system on behalf of the application. Therefore the system should be cognizant of domains.

3.5. Configuration and Policy

1. Operating system does not keep separate, per domain copies of same configuration objects (e.g., DNS server addresses, NTP server addresses, ..) and thus these are either overwritten by the operating system when received from multiple provisioning domains, or ignored when not received on a so-called primary interface.

2. There’s no way yet to handle multiple policies coming from different domains. E.g., corporate node usage typically means that the corporation issues some policy on that Wi-Fi interface (and others as well). In this case, the carrier and corporation

domains and their policies will overlap over the Wi-Fi interface. Having a common policy language might help to detect and reason about such conflicts, but conflict resolution is another problem. Ultimately, the issue are the different authorities on these domains (e.g., user at home, admin at corporation and carrier for wireless broadband) and how they resolve their conflicts in the overlap situations. Note: Domains and their policies may span multiple interfaces. There is a fixed hierarchy of domains and their authorities, but the top authority may decide to delegate to others certain parts of the system and to their policies, as long as these don’t conflict with his. A conflict resolution that respects the hierarchy is needed.

4. Current Practice Analysis

4.1. Mobile Handset Operating Systems

4.1.1. Nokia S60 3rd Edition, Feature Pack 2

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.1.2. Microsoft Windows Mobile 2003 Second Edition

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.1.3. RIM BlackBerry

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.1.4. Google Android

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.1.5. Qualcomm AMSS

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.1.6. Arena Connection Manager

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.1.7. Access Selection

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.2. Computer Operating Systems

4.2.1. Microsoft Windows

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

4.2.2. Linux and BSD-based Operating Systems

The following problems occurs:

Naming and Addressing: 1, 2, 3, 4

Routing: 1, 2, 3

Reachability: 1, 2, 3

Domain Selection: 1, 2

Configuration and Policy: 1, 2

4.2.3. Apple MacOS X

The following problems occurs:

Naming and Addressing:

Routing:

Reachability:

Domain Selection:

Configuration and Policy:

5. Security Considerations

TBD.

6. IANA Considerations

This document does not require any IANA actions.

7. Informative References

[I-D.ietf-mif-current-practices]
Wasserman, M. and P. Seite, "Current Practices for Multiple Interface Hosts", draft-ietf-mif-current-practices-02 (work in progress), June 2010.

[I-D.ietf-mif-problem-statement]
Blanchet, M. and P. Seite, "Multiple Interfaces Problem Statement", draft-ietf-mif-problem-statement-05 (work in progress), July 2010.

[I.D-MIF-DNS]
Savolainen, T., "DNS Server Selection on Multi-Homed Hosts", February 2010, <draft-savolainen-mif-dns-server-selection-02.txt (work in progress)>.

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

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

Authors' Addresses

Julien Laganier
Qualcomm Incorporated
5775 Morehouse Drive
San Diego, CA 92121
USA

Phone: +1 858 858 3538
Email: julienl@qualcomm.com

Gabriel Montenegro
Microsoft

Email: gmonte@microsoft.com

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
FI-02600 Espoo
FINLAND

Email: jouni.nospam@gmail.com

Teemu Savolainen
Nokia
Hermiankatu 12 D
FI-33720 Tampere
FINLAND

Email: teemu.savolainen@nokia.com

Zhen Cao
China Mobile

Email: zehn.cao@chinamobile.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 8, 2011

G. Chen
China Mobile

March 7, 2011

Happy Eyeballs Extension for Multiple Interfaces
draft-chen-mif-happy-eyeballs-extension-00

Abstract

The memo has been proposed to extend happy eyeballs algorithm to fit into multiple interfaces environment. Based on this extended heuristic algorithm, a client with multiple interface could determine the optimal flow path in which specific interface has been chosen. Furthermore, an appropriate IP address family for each interface can be also identified to guarantee user experiences during IPv6 transition period.

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 September 8, 2011.

Copyright Notice

Table of Contents

1. Introduction

In multiple interface context, the problems raised by hosts with multiple interfaces have been discussed. The MIF problem statement[MIF-PS] has described the various issues when using a MIF node on which multiple interfaces are used and results in wrong domain selection. Happy Eyeballs [HAPPY-EYEBALLS] has described how a dual-stack client can determine the functioning path to a dual-stack server. It's using heuristic algorithm help applications to quickly determine if IPv6 or IPv4 is the most optimal to connect to a server. That is a good practice to achieve intelligent path selection. However, the assumption here is single-homed host. The interaction with multiple interfaces is still waiting for further study.

This memo has been proposed to extend happy eyeballs algorithm to fit into multiple interfaces environment. That could achieve win-win situation. Based on this extended heuristic algorithm, a client with multiple interface could determine the optimal flow path in which specific interface has been chosen. Furthermore, an appropriate IP address family for each interface can be also identified to guarantee user experiences during IPv6 transition period.

2. Heuristic Happy Eyeballs Extension Algorithm

The section details extended Happy Eyeballs algorithm, including new defined data structure, functionalities and example flow.

2.1. The Framework for Extended Algorithm

The Figure 1 shows the proposed framework for extended algorithm.

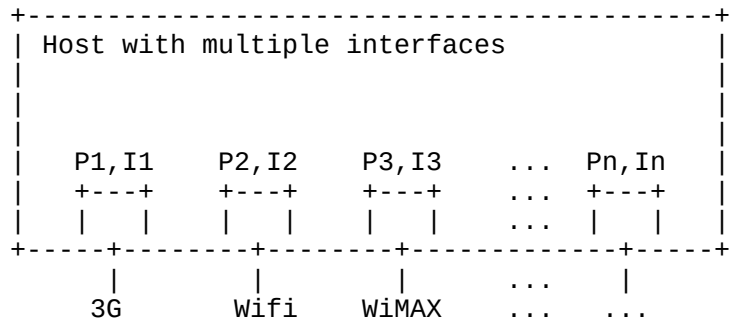


Figure 1: Multiple Interface Mode for Extended Algorithm

Each interface will be configured with weighting coefficient, which is composed of pair values. Apart from value P, which is following current definition in [HAPPY-EYEBALLS], value I is defined to indicate preference of interfaces selection. In general, value I is responsible for interface selection; value P is a indication to identify IPv4 or IPv6 family has been preferred.

2.2. Algorithm for Interface Selection

According to the definition, applications will take account of value I to identify which interface has been chosen before sending out data packages .

Each interface is configured with one value, I. I is served as an indication to identify which interface is preferred for a specific destination or hostname. A positive value indicates preference of specific interface compared to others. The value is justified according to TCP establishment duration. The detailed flow can be found in 2.4. The rule here is that shorter is winner. When one interface defeats others, the corresponding value I will be set to positive value. Other interfaces will be set negative value orderly according to caused time for TCP connection, like -1, -2, etc. A value of 0 indicates equal weight for multiple interfaces. In this case, the heuristic connection is not valid for interface selection.

After several interface values I have been configured, the traffic flow targeted to specific destination address or hostname will follow this guidance to choose proper interface. When connection problems are occurred to preferred connection, the value I need to be adjusted. The adjustment of value I will do polling-based scheme. the value I corresponding to suboptimal interface will be configured as positive. And previously optimal value I will be set to most-negative.

2.3. IPv4/IPv6 Selection Algorithm for Individual Interface

for a specific interface, the choice of IP address family relies on Happy Eyeballs algorithm, which defined in [HAPPY-EYEBALLS].

2.4. Example of Extended Algorithm

The Figure 2 shows example flow for algorithm implemetation,in which a host with three interfaces is demonstrated. It’s easy to apply that to more interfaces situation.

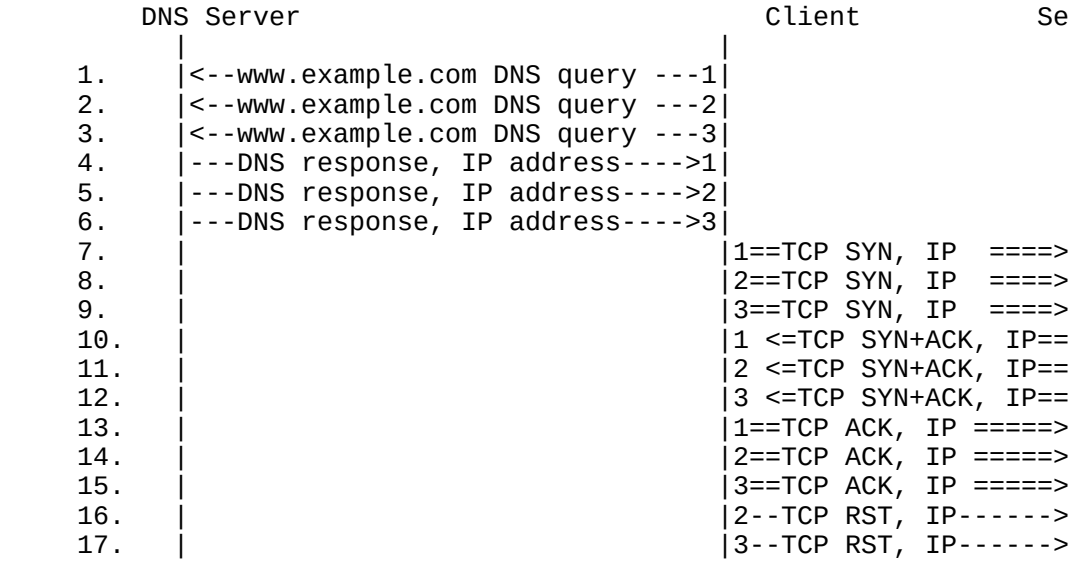


Figure 2: Example of Algorithm Execution

Above example of algorithm execution emphasizes on the computation of coefficient value I. The value P will be calculated as defined in [HAPPY-EYEBALLS].

When client initiates application session, multiple interfaces will do DNS query simultaneously. As a results, DNS server will return IP address related to requested FQDN, as shown from step 1 to 6. Afterwards, three interfaces are trying to establish TCP connection with remote peer IP address synchronously as shown from step 7 to 15. In this example, interface 1 competes with another two and firstly finish TCP connection. So value I will be set to 1. And second one is set to -1; thelast one is set to -2. The interface 2 and 3 will then send RST to remote IP peer for release TCP sessions.

Through the whole process, interface 1 will be set to most-preferred interface for a specific this particular destination or hostname. if there is an accidental connection problem has occurred, the value I with interface 1 will be set to -2. And the values I related to interface 2 and 3 will increase sequentially to be set to 1 and -1. Subsequent traffic will take interface 2 as preferred interface for traffic delivery.

3. Additional Considerations

3.1. Usage Scope

Happy Eyeballs is trageting to HTTP context, but it is useful and applicable to other time-sensitive applications.

3.2. Flow Continuity

Usually, interface changing is happened at beginning of new session. So, there is no flow continuity issues for ongoing TCP sesson.

4. IANA Considerations

This memo includes no request to IANA.

5. Security Considerations

TBD

6. Normative References

[HAPPY-EYEBALLS] Wing, D., "Happy Eyeballs: Trending Towards Success with Dual-Stack Hosts", draft-ietf-v6ops-happy-eyeballs-00.txt (work in progress), March 2011.

[MIF-PS] Blanchet, M., "Multiple Interfaces and Provisioning Domains Problem Statement", draft-ietf-mif-problem-statement-09.txt (work in progress), October 2010.

Author’s Address

Gang Chen
China Mobile
53A,Xibianmennei Ave.,
Xuanwu District,
Beijing 100053
China

Email: chengang@chinamobile.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 15, 2011

W. Dec, Ed.
Cisco Systems
T. Mrugalski
Gdansk University of Technology
T. Sun
China Mobile
B. Sarikaya
Huawei USA
March 14, 2011

DHCPv6 Route Option
draft-ietf-mif-dhcpv6-route-option-01

Abstract

This document describes DHCPv6 Route Options for provisioning IPv6 routes on DHCPv6 client nodes. This is expected to improve the ability of an operator to configure and influence a nodes' ability to pick an appropriate route to a destination when this node is multi-homed and where other means of route configuration may be impractical.

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 [RFC2119].

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 September 15, 2011.

Copyright Notice

Table of Contents

1. Introduction

The Neighbor Discovery (ND) ICMPv6 protocol [RFC4861] provides a mechanism for hosts to discover one or more default routers on a directly connected network segment. Extensions to the Router Advertisement (RA) protocol defined in [RFC4191] allow hosts to discover the preferences for multiple default routers on a given link, as well as any specific routes advertised by these routers. This allows network administrators to better handle multi-homed host topologies and influence the route selection by the host. This ND based mechanism however is sub optimal or impractical in some multi-homing scenarios, where DHCPv6 [RFC3315]is seen to be more viable.

This draft defines the DHCPv6 Route Option for provisioning IPv6 routes on DHCPv6 clients. The proposed option is primarily envisaged for use by DHCPv6 client nodes that are capable of making basic IP routing decisions and maintaining an IPv6 routing table, broadly in line with the capabilities of a generic host as described in [RFC4191].

Throughout the document the words node and client are used as a reference to the device with such routing capabilities, hosting the DHCPv6 client software. The route information is taken to be equivalent to static routing, and limited in the number of required routes to a handful.

2. Problem overview

The solution described in this document applies to multi-homed scenarios including ones where the client is simultaneously connected to multiple access network (e.g. WiFi and 3G). The following scenario is used to illustrate the problem as found in typical multi-homed residential access networks. It is duly noted that the problem is not specific to IPv6, occurring also with IPv4, where it is today solved by means of DHCPv4 classless route information option [RFC3442], or alternative configuration mechanisms.

In multi-homed networks, a given user’s node may be connected to more than one gateways. Such connectivity may be realized by means of dedicated physical or logical links that may also be shared with other users nodes. In such multi-homed networks it is quite common for the network operator to offer the delivery of a particular type of IP service via a particular gateway, where the service can be characterised by means of specific destination IP network prefixes. Thus, from an IP routing perspective in order for the user node to select the appropriate gateway for a given destination IP prefix, recourse needs to be made to classic longest destination match IP

routing, with the node acquiring such prefixes into its routing table. This is typically the remit of dynamic Internal Gateway Protocols (IGPs), which however are rarely used by operators in residential access networks. This is primarily due to operational costs and a desire to contain the complexity of user nodes and IP Edge devices to a minimum. While, IP Route configuration may be achieved using the ICMPv6 extensions defined in [RFC4191], this mechanism does not lend itself to other operational constraints such as the desire to control the route information on a per node basis, the ability to determine whether a given node is actually capable of receiveing/processing such route information. A preferred mechanism, and one that additionally also lends itself to centralized management independent of the management of the gateways, is that of using the DHCP protocol for conveying route information to the nodes.

3. DHCPv6 Based Solution

A DHCPv6 based solution allows an operator an on demand and node specific means of configuring static routing information. Such a solution also fits into network environments where the operator prefers to manage RG configuration information from a centralized DHCP server. [I-D.troan-multihoming-without-nat66] provides additional background to the need for a DHCPv6 solution to the problem.

In terms of the high level operation of the solution defined in this draft, a DHCPv6 client interested in obtaining routing information request the route option using the DHCPv6 Option Request Option (ORO) sent to a server. A Server, when configured to do so, provides the requested route information as part of a nested options structure covering; the next-hop address; the destination prefix; the route metric; any additional options applicable to the destination or next-hop. The overall DHCPv6 design follow a similar approach to that used in the design of the IA_NA, IA_TA and IA_PD options in [RFC3633]

4. DHCPv6 Route Option

A DHCPv6 client interested in obtaining routing information includes the OPTION_IA_RT as par of its DHCPv6 Option Request Option (ORO) in messages directed to a server (as allowed by [RFC3315], ie Solicit, Request, Renew, Rebind, Confirm or Information-request messages). A Server, when configured to do so, provides the requested route information using the OPTION_IA_RT option in messages sent in response (Advertise, and Reply). So as to allow the route option to be both extensible, as well as conveying detailed info for routes, use is made of a nested options structure. An IA_RT conveys one or

more OPTION_NEXT_HOP options that specify the IPv6 next hop addresses. Each OPTION_NEXT_HOP conveys in turn one or more OPTION_RT_PREFIX options that represents the IPv6 destination prefixes reachable via the given next hop. The Formats of the OPTION_IA_RT, OPTION_NEXT_HOP and OPTION_RT_PREFIX are defined in the following sub-sections

The DHCPv6 Route Option format borrows from the principles of the Route Information Option defined in [RFC4191]. One notable exception with respect to [RFC4191] is however that a Route Lifetime element is not defined. The information conveyed by the DHCPv6 Route Option is considered valid until changed or refreshed by general events that trigger DHCPv6 or route table state changes on a node, thus not requiring a specific route lifetime. In the event that it is desired for the client to request a refresh of the route information (and other stateless DHCPv6 options), use of the generic DHCPv6 Information Refresh Time Option, as specified in [RFC4242] is envisaged.

4.1. DHCPv6 Route Option Format

To separate routing information from other options conveyed in a DHCPv6 message, the DHCPv6 Route Option is defined and is used to convey to a client one or more IPv6 routes. Each IPv6 route consists of an IPv6 next hop address, an IPv6 destination prefix (a.k.a. the destination subnet), and a host preference value for the route. Elements of such route (e.g. Next hops and prefixes associated with them) are conveyed in IA_RT's options, rather than in the IA_RT option itself.

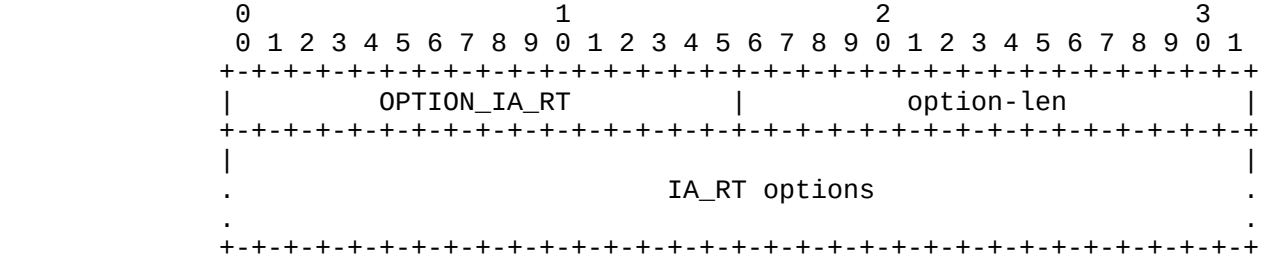


Figure 1: IPv6 Routes Option Format

option-code: OPTION_IA_RT (TBD).

option-len: Length of the IA_RT options field.

IA_RT options: Options associated with this IA_RT. This includes, but is not limited to, OPTION_NEXT_HOP options that specify next hop addresses.

The Route option MUST NOT appear in the following DHCPv6 messages: Solicit, Request, Renew, Rebind, Information-Request. The Route Option MAY appear in ADVERTISE and REPLY messages.

Discussion: Traditionally, grouping options (IA_NA, IA_TA and IA_RD) contain an identifier field (IAID) that must be unique among identifiers generated by one client. It is used to differentiate between several options of the same type (e.g. several IA_NA options) that may be used simultaneously. However, it is assumed that client will never use more than one IA_RT option therefore such an identifier is not needed.

4.2. Next Hop Option Format

The Next Hop Option defines the IPv6 address of the next hop, usually corresponding to a specific next-hop router. For each next hop address there can be one or more prefixes reachable via that next hop.

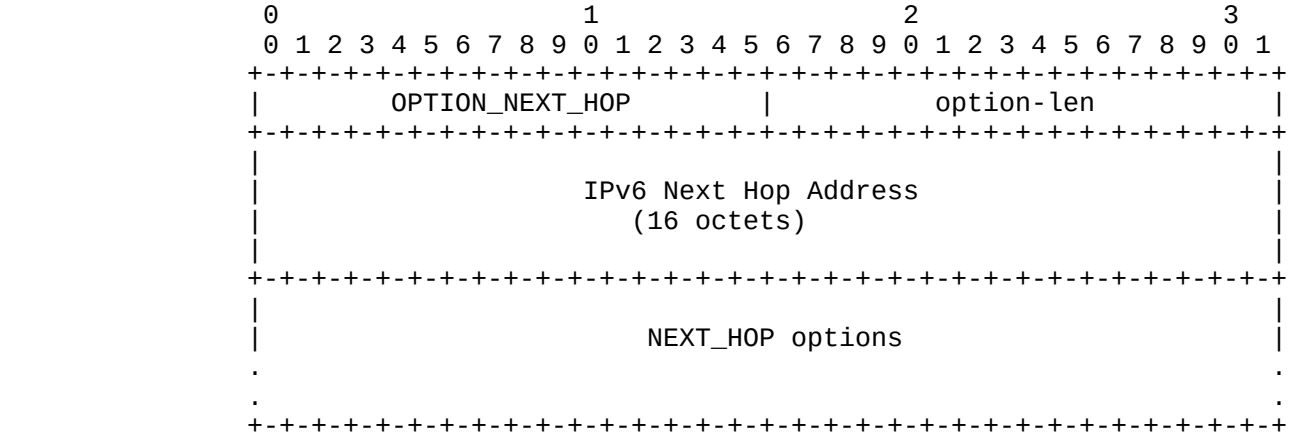


Figure 2: IPv6 Route Option Format

option-code: OPTION_NEXT_HOP (TBD).

option-len: 16 + Length of NEXT_HOP options field.

IPv6 Next Hop Address: 16 octet long field that specified IPv6 address of the next hop.

NEXT_HOP options: Options associated with this Next Hop. This includes, but is not limited to, one or more OPTION_RT_PREFIX options that specify prefixes reachable through the given next hop.

4.3. Route Prefix Option Format

The Route Prefix Option is used to convey information about a single prefix that represents the destination network. The Route Prefix Option is used as a sub-option in the previously defined Next Hop Option.

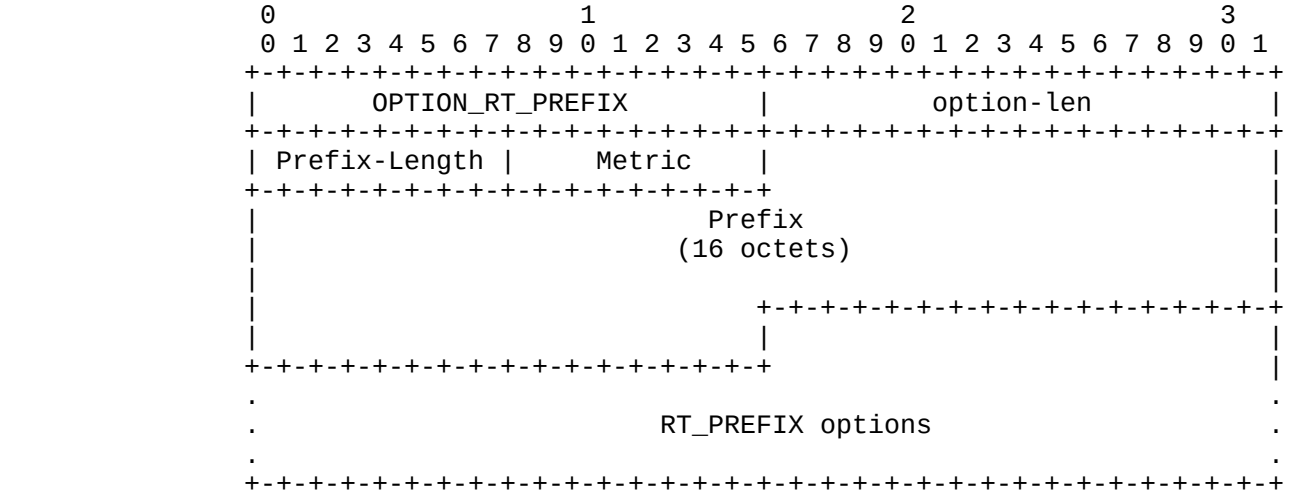


Figure 3: Route Prefix Option Format

option-code: OPTION_RT_PREFIX (TBD).

option-len: 18 + length of RT_PREFIX options.

Prefix Length: 8-bit unsigned integer. The length in bits of the IP Prefix. The value ranges from 0 to 128. This field represents the number of valid leading bits in the prefix.

Metric: Route Metric. 8-bit signed integer. The Route Metric indicates whether to prefer the next hop associated with this prefix over others, when multiple identical prefixes (for different next hops) have been received.

Prefix: Fixed length 16 octet field containing an IPv6 prefix.

RT_PREFIX options: Options specific to this particular prefix.

5. DHCPv6 Server Behavior

When configured to do so s DHCPv6 server shall provide the Routes Option in ADVERTISE and REPLY messages sent to a client that requested the route option. Each Next Hop Option sent by the server must convey at least one Route Prefix Option.

Servers SHOULD NOT send Route Option to clients that did not explicitly requested it, using the ORO.

Servers MUST NOT send Route Option in messages other than ADVERTISE or REPLY.

Servers MAY also include Status Code Option, defined in Section 22.13 of the [RFC3315] to indicate the status of the operation.

Servers MUST include the Status Code Option, if the requested routing configuration was not successful and SHOULD use status codes as defined in [RFC3315] and [RFC3633].

The maximum number of routing information in one DHCPv6 message depend on the maximum DHCPv6 message size defined in [RFC3315]

Discussion: How should server indicate that there are no specific routes for this particular client? The reasonable behavior is to return empty IA_RT option, possibly with Status Code indicating Success. Another approach could be to simply not return any IA_RT option.

6. DHCPv6 Client Behavior

A DHCPv6 client compliant with this specification MUST request the Route Option (option value TBD) in an Option Request Option (ORO) in the following messages: Solicit, Request, Renew, Rebind, Information-Request or Reconfigure. The messages are to be sent as and when specified by [RFC3315].

When processing a received Route Option a client MUST substitute a received 0::0 value in the Next Hop Option with the source IPv6 address of the received DHCPv6 message. It MUST also associate a received Link Local next hop addresses with the interface on which the client received the DHCPv6 message containing the route option. Such a substitution and/or association is useful in cases where the DHCPv6 server operator does not directly know the IPv6 next-hop

address, other than knowing it is that of a DHCPv6 relay agent on the client LAN segment. DHCPv6 Packets relayed to the client are sourced by the relay using this relay's IPv6 address, which could be a link local address.

The Client MAY refresh assigned route information periodically. The generic DHCPv6 Information Refresh Time Option, as specified in [RFC4242], can be used when it is desired for the client to periodically refresh of route information.

The routes conveyed by the Route Option should be considered as complimentary to any other static route learning and maintenance mechanism used by, or on the client with one modification: The client MUST flush DHCPv6 installed routes following a link flap event on the DHCPv6 client interface over which the routes were installed. This requirement is necessary to automate the flushing of routes for clients that may move to a different network.

7. IANA Considerations

A DHCPv6 option number of TBD for the introduced Route Option. IANA is requested to allocate three DHCPv6 option codes referencing this document: OPTION_IA_RT, OPTION_NEXT_HOP and OPTION_RT_PREFIX.

8. Security Considerations

The overall security considerations discussed in [RFC3315] apply also to this document. The Route option could be used by malicious parties to misdirect traffic sent by the client either as part of a denial of service or man-in-the-middle attack. An alternative denial of service attack could also be realized by means of using the route option to overflowing any known memory limitations of the client, or to exceed the client's ability to handle the number of next hop addresses.

Neither of the above considerations are new and specific to the proposed route option. The mechanisms identified for securing DHCPv6 as well as reasonable checks performed by client implementations are deemed sufficient in addressing these problems.

9. Contributors and .Acknowledgements

This document would not have been possible without the significant contribution provided by: Arifumi Matsumoto, Hui Deng, Richard Johnson, Zhen Cao.

The authors would also like to thank Alfred Hines, Ralph Droms, Ted Lemon, Ole Troan, Dave Oran and Dave Ward for their comments and useful suggestions.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[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.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

10.2. Informative References

[I-D.troan-multihoming-without-nat66] Troan, O., Miles, D., Matsushima, S., Okimoto, T., and D. Wing, "IPv6 Multihoming without Network Address Translation", draft-troan-multihoming-without-nat66-01 (work in progress), July 2010.

[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.

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

[RFC4242] Venaas, S., Chown, T., and B. Volz, "Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 4242, November 2005.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

Authors' Addresses

Wojciech Dec (editor)
Cisco Systems
Haarlerbergweg 13-19
1101 CH Amsterdam
The Netherlands

Email: wdec@cisco.com

Tomasz Mrugalski
Gdansk University of Technology
Storczykowa 22B/12
Gdansk 80-177
Poland

Phone: +48 698 088 272
Email: tomasz.mrugalski@eti.pg.gda.pl

Tao Sun
China Mobile
Unit2, 28 Xuanwumenxi Ave
Beijing, Xuanwu District 100053
China

Phone:
Email: suntao@chinamobile.com

Behcet Sarikaya
Huawei USA
1700 Alma Dr. Suite 500
Plano, TX 75075
United States

Phone: +1 972-509-5599
Fax:
Email: sarikaya@ieee.org
URI:

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: September 12, 2011

T. Savolainen
Nokia
J. Kato
NTT

March 11, 2011

Improved DNS Server Selection for Multi-Homed Nodes
draft-ietf-mif-dns-server-selection-01

Abstract

A multi-homed node can be connected to multiple networks that may utilize different DNS namespaces. The node commonly receives DNS server configuration information from all connected networks. Some of the DNS servers may have information about namespaces other servers do not have. When the multi-homed node needs to utilize DNS, it has to choose which of the servers to contact to. This document describes a policy based method for helping on selection of DNS server, for both forward and reverse DNS lookup procedures, with help of DNS suffix and IPv6 prefix information received via DHCPv6.

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 September 12, 2011.

Copyright Notice

Table of Contents

1. Introduction

A multi-homed node faces several problems a single-homed node does not encounter, as is described in [I-D.ietf-mif-problem-statement]. This document studies in detail the problems local namespaces may cause for multi-homed nodes and provides a solution for IPv6 domain. The node may be implemented as a host or as a router.

When multiple namespaces are visible for a node, some DNS servers have information other servers do not have. Because of that, a multi-homed node cannot assume every DNS server is able to properly answer for any query, but instead the node must be able to ask right server for the information it needs.

An example of an application that benefits from multi-homing is a web browser that commonly accesses many different destinations and needs to be able to dynamically communicate over different network interfaces.

In deployments where multiple namespaces are present, selection of correct route and destination and source addresses for the actual IP connection is crucial as well, as the resolved destination’s IP addresses may be only usable on the network interface over which the name was resolved on. Hence solution described in this document is assumed to be commonly used in combination with tools for delivering additional routing and source and destination address selection policies.

The Appendix A describes best current practices possible with tools preceding this document and on networks not supporting the solution described in this document. As it is possible to solve the problem with less efficient and less explicit manners, the new solution may be considered as an optimization. However, in some environments this solution is considered essential.

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 [RFC2119].

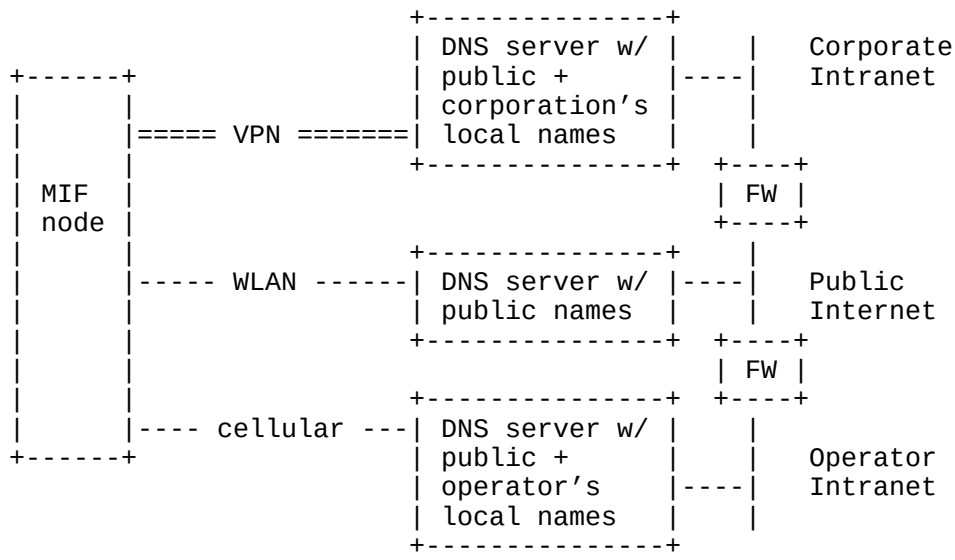
2. Problem description for local namespaces with multi-homed nodes

This chapter describes two host multi-homing related local namespace scenarios for which the procedure described in chapter 3 provides a solution for. Essentially the same challenges may be faced by Consumer Premises Equipment as is described in

[I-D.ietf-v6ops-multihoming-without-nat66]. This chapter additionally describes a related problem for which this document provides only partial solution.

2.1. Fully qualified domain names with limited scopes

A multi-homed node may be connected to one or more networks that are using local namespaces. As an example, the node may have simultaneously open a wireless LAN (WLAN) connection to the public Internet, cellular connection to an operator network, and a virtual private network (VPN) connection to a corporate network. When an application initiates a connection establishment to an FQDN, the host needs to be able to choose the right DNS server for making a successful DNS query. This is illustrated in the figure 1. An FQDN for a public name can be usually resolved with any DNS server, but for an FQDN of corporation's or operator's service's local name the node needs to be able to correctly select the right DNS server for the DNS resolution, i.e. do also network interface selection already before destination's IP address is known.

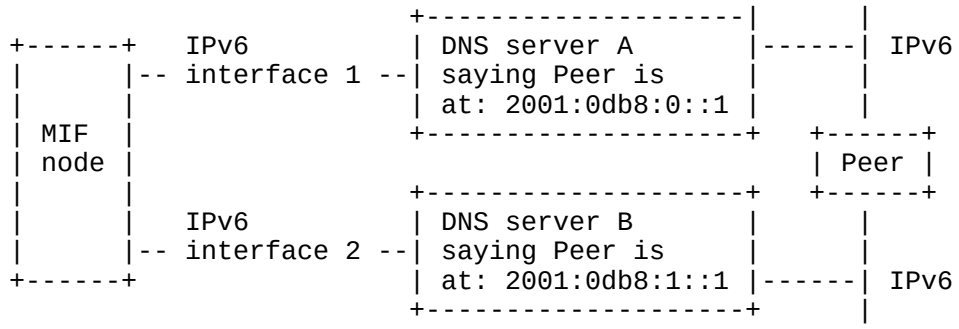


Local DNS namespaces illustrated

Figure 1

2.2. Network interface specific IP addresses

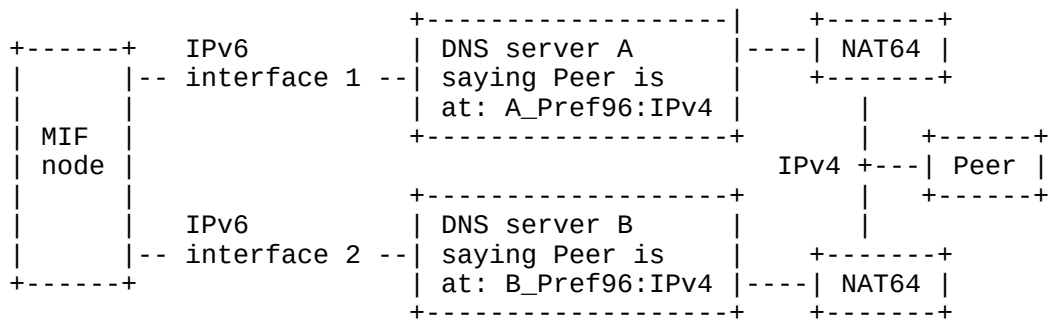
In the second problem an FQDN is valid and resolvable via different network interfaces, but to different and not necessarily globally reachable IP addresses, as is illustrated in the figure 2. Node's routing and source and destination address selection mechanism must ensure the destination's IP address is only used in combination with source IP addresses of the network interface the name was resolved on.



Local DNS namespaces and different IP addresses for an FQDN on interfaces 1 and 2.

Figure 2

Similar situation can happen with IPv6 protocol translation and AAAA record synthesis [I-D.ietf-behave-dns64]. A synthesised AAAA record is guaranteed to be valid only on a network interface it was synthesized on. Figure 3 illustrates a scenario where the peer's IPv4 address is synthesized into different IPv6 addresses by DNS servers A and B.



AAAA synthesis results in interface specific IPv6 addresses.

Figure 3

A thing worth noting is that interface specific IP addresses can cause problems also for a single-homed host, if the host retains its DNS cache during movement from one network interface to another. After the interface change a host could have both positive and negative DNS cache entries no longer valid on the new network interface. Because of this the cached DNS information should be considered network interface local instead of node global.

2.3. A problem not fully solved by the described solution

A more complex scenario is an FQDN, which in addition to possibly resolving into network interface specific IP addresses, identifies on different network interfaces completely different peer entities with potentially different set of service offerings. In even more complex scenario, an FQDN identifies unique peer entity, but one that provides different services on its different network interfaces. The solution described in this document is not able to tackle these higher layer issues. In fact, these problems may be solvable only by manual user intervention.

However, when DNSSEC is used, the DNSSEC validation procedure may provide assistance for selecting correct responses for some, but not all, use cases. A node may prefer to use the DNS answer that validates with the preferred trust anchor.

3. Deployment scenarios

This document has been written with three particular deployment scenarios in mind. First being a Consumer Premises Equipment (CPE) with two or more uplink VLAN connections. Second scenario involves a

cellular device with two uplink Internet connections: WLAN and cellular. Third scenario is for VPNs, where use of local DNS server may be preferred for latency reasons, but corporate DNS server must be used to resolve private names used by the corporation.

3.1. CPE deployment scenario

A home gateway may have two uplink connections leading to different networks, as is described in [I-D.ietf-v6ops-multihoming-without-nat66]. In this scenario only first uplink connections lead to Internet, while second uplink connection leads to a private network utilizing private namespace.

It is desirable that the CPE does not have to send DNS queries over both uplink connections, but instead CPE should send default queries to the DNS server of the interface leading to the Internet, and queries related to private namespace to the DNS server of the private network.

Depending on actual deployments, all VLAN connections may be considered secure.

3.2. Cellular network scenario

A cellular device may have both WLAN and cellular network interfaces up. In such a case it is often desirable to use WLAN by default, except for those connections cellular network operator wants to go over cellular interface. The cellular network may utilize private names and hence the cellular device needs to ask for those through the cellular interface.

In this scenario cellular interface can be considered secure and WLAN often insecure.

3.3. VPN scenario

Depending on a deployment, there may be need to use VPN only for traffic destined to a corporate network. The corporation may be using private namespace, and hence related DNS queries should be send over VPN to the corporate DNS server, while by default a DNS server of a local access network may be used.

In this scenario VPN interface can be considered secure and local access network insecure.

4. Improved DNS server selection

This chapter describes a procedure a (stub / proxy) resolver may utilize for improved DNS server selection in face of multiple namespaces and multiple simultaneously active network interfaces.

4.1. Procedure for prioritizing DNS servers and handling responses

A resolver SHALL build a priority list of DNS servers it will contact to depending on the query. To build the list in safe and optimal way, a node SHOULD ask with DHCPv6 which DNS servers of each network interface are most likely able to successfully serve forward lookup requests matching to specific DNS suffixes or reverse (PTR record) lookup requests matching to specific IPv6 prefixes.

A resolver lacking more explicit information shall assume that all information is available from any DNS server of any network interface. The DNS servers learnt by other DNS server address configuration methods MUST be handled as medium priority default servers.

When a DNS query needs to be made, the resolver SHOULD give highest precedence to the DNS servers explicitly known to serve matching suffixes or prefixes. However, the resolver MUST take into account different trust levels of pieces of DNS server selection information the resolver may have received from node’s network interfaces. The resolver MUST generally prefer more trusted DNS servers and less trusted DNS servers MAY be of highest priority only if trusted interfaces specifically configure DNS servers to be of low priority. The non-exhaustive list on figure 4 illustrates how the different trust levels of received DNS server selection information MUST influence the DNS server selection logic.

Information from more trusted interface A	Information from less trusted interface B	Resulting DNS server priority selection
1. Medium priority default	Medium priority default	Default: A, then B
2. Medium priority default	High priority default High priority specific	Default: A, then B Specific: A, then B
3. Low priority default	Medium priority default	Default: B, then A
4. Low priority default High priority specific	Medium priority default	Default: B, then A Specific: A, then B

Figure 4: DNS server selection in case of different trust levels

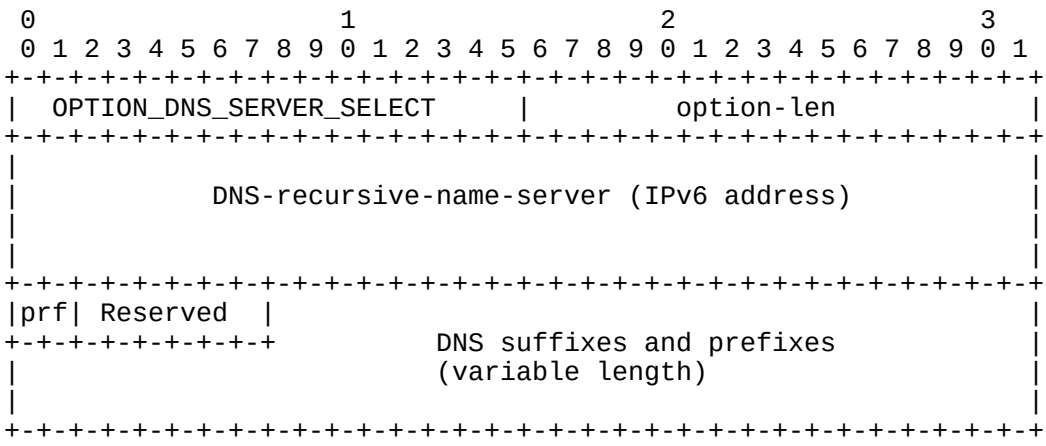
The resolver SHOULD avoid sending queries to different interfaces in parallel as that may waste resources, sometimes significantly, and would also unnecessary reveal information about ongoing communications. Independently of whether DNS queries are sent in series or parallel, replies for DNS queries MUST be waited until acceptable positive reply is received, all replies are received, or time out occurs. Specifically, the resolver MUST NOT proceed with a positive reply from less trusted server that cannot be validated with DNSSEC if the DNS query sent to more trusted server is still pending. If the reply does not validate with a preferred trust anchor, the node may send queries also to other servers in an attempt to receive a reply that validates with more preferred trust anchor. (DISCUSS: What about those DNS servers that instead of negative answer always return positive reply with an IP address of some captive portal?)

When local namespaces are present a negative reply from a DNS server implies only that the particular DNS server was not able to serve the query. However, it is not probable that the secondary DNS servers on the same network interface, on a same administrative domain, would be able to serve either. Therefore, the next DNS server resolver contacts SHOULD be from another network interface.

Resolver SHOULD use DNSSEC to validate all received DNS replies. In the case DNSSEC validation fails the resolver MUST resend the query to the next preferred DNS server.

4.2. DNS server selection option

A DHCPv6 option described below is used to inform resolvers which DNS server should be contacted when initiating forward or reverse DNS lookup procedures.



option-code: OPTION_DNS_SERVER_SELECT (TBD)

option-len: Lenght of the option in octets

DNS-recursive-name-server: An IPv6 address of a DNS server

prf: DNS server preference, for selecting between equally trusted DNS servers:
- 01 High
- 00 Medium
- 11 Low
- 10 Reserved

Reserved: Flags reserved for the future. MUST be set to zero.

DNS suffixes and prefixes: The list of DNS suffixes for forward DNS lookup and prefixes for reverse DNS lookup the DNS server has special knowledge about. Field MUST be encoded as specified in section "Representation and use of domain names" of [RFC3315].
Special suffix of "." is used to indicate capability to resolve global names and act as a default name server. Lack of "." suffix on the list indicates DNS server only has information related to listed suffixes and prefixes.
Prefixes for reverse mapping are encoded as defined for ip6.arpa [RFC3152].

DHCPv6 option for explicit DNS suffix configuration

Figure 5

A node SHOULD include an OPTION_ORO option in a DHCPv6 request with the OPTION_DNS_SERVER_SELECT option code to inform the DHCPv6 server about the support for the improved DNS server selection logic. DHCPv6 server receiving this information MAY then choose to provision DNS server addresses only with the OPTION_DNS_SERVER_SELECT.

The OPTION_DNS_SERVER_SELECT contains one or more DNS suffixes the related DNS server has particular knowledge of. The option can occur multiple times in a single DHCPv6 message, if multiple DNS servers are to be configured.

A resolver SHOULD prioritize between equally trusted DNS servers with help of the preference field. The resolver MUST NOT prioritize less trusted DNS servers higher than trusted, even in the case of less trusted server would apprently have additional information. In the case of all other things being equal the resolver shall make the prioritization decision based on its internal preferences.

IPv6 prefixes should cover all the DNS suffixes configured in this option. Prefixes should be as long as possible to avoid potential collision with information received on other option instances or with options received from DHCPv6 servers of other network interfaces. Overlapping IPv6 prefixes are interpreted so that the resolver can use any of the DNS servers for queries mathing the prefixes.

If the OPTION_DNS_SERVER_SELECT contains a DNS server address already learned from other DHCPv6 servers and possibly through other network interfaces, the node MAY append new prefixes and suffixes to the information received earlier. The node MUST NOT remove previously obtained information. However, the node SHOULD NOT extent lifetime of earlier information either. In the case conflicting DNS server address and related information is learned from less trusted interface, the node MAY choose to ignore the option.

As the DNS options of [RFC3646], the OPTION_DNS_SERVER_SELECT option MUST NOT appear in any other than the following DHCPv6 messages: Solicit, Advertise, Request, Renew, Rebind, Information-Request, and Reply.

The node SHOULD create a host specific route for the DNS server address. The route must point to the interface DNS server address was learned on. This is required to ensure DNS queries are sent out via the right interface.

In the case of a trusted DNS server replying negatively to a question having matching suffix, it will be for implementation to decide whether to consider that as a final response, or whether to ask also from other DNS servers. The implementation decision may be based,

for example, on deployment or trust models.

The information conveyed in OPTION_DNS_SERVER_SELECT is considered valid until changed or refreshed by general events that trigger DHCPv6 action. In the event that it is desired for the client to request a refresh of the information, use of generic DHCPv6 Information Refresh Time Option, as specified in [RFC4242] is envisaged.

4.3. Coexistence with RFC3646

The OPTION_DNS_SERVER_SELECT is designed to coexist with OPTION_DNS_SERVERS defined in [RFC3646]. The DNS servers configured via OPTION_DNS_SERVERS MUST BE considered as default name servers with medium preference. When both options are received from the same network interface and the OPTION_DNS_SERVER_SELECT contains default DNS server address, the resolver MUST make the decision which one to prefer based on preferences. If OPTION_DNS_SERVER_SELECT defines medium preference then DNS server from OPTION_DNS_SERVER_SELECT SHALL be selected. All default servers are assumed to be able to resolve queries for global names.

If both OPTION_DNS_SERVERS and OPTION_DNS_SERVER_SELECT contain the same DNS server(s) IPv6 address(es), only one instance of each DNS servers' IPv6 addresses shall be added to the DNS server list.

If a node had indicated support for OPTION_DNS_SERVER_SELECT in DHCPv6 request, the DHCPv6 server may choose to omit sending of OPTION_DNS_SERVERS. This enables offloading use case where network administrator wishes to only advertise low priority default DNS servers.

4.4. Interactions with OPTION_DOMAIN_LIST

A node may be configured with DNS search list with OPTION_DOMAIN_LIST. Resolution for the name containing any dots SHOULD first be attempted with DNS servers of all interfaces as described earlier. Only if the resolution fails the node SHOULD append the name with search list suffix(es) and then utilize improved DNS server selection algorith again to decide which DNS server(s) to contact next. A name without any dots SHALL immediately be appended with suffix(es) and improved DNS server selection be utilized on resolution.

4.5. CNAME/DNAME record considerations

If a node receives a reply with a canonical name (CNAME) or delegation name (DNAME) the follow-up queries MUST be sent to the

same DNS server irrespectively of the FQDN received. Otherwise referrals may fail.

5. Example of a node behavior

Figure 6 illustrates node behavior when it initializes two network interfaces for parallel usage and learns DNS suffix and prefix information from DHCPv6 servers.

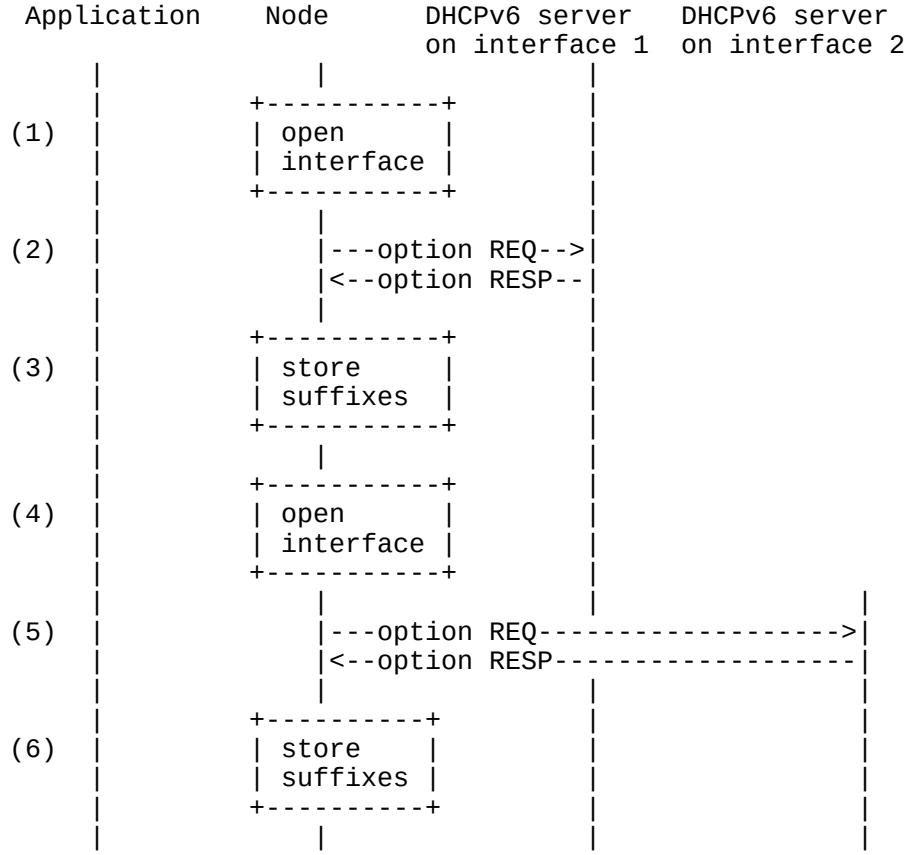


Illustration of learning DNS suffixes

Figure 6

Flow explanations:

1. A node opens its first network interface

2. The node obtains DNS suffix and IPv6 prefix information for the new interface 1 from DHCPv6 server

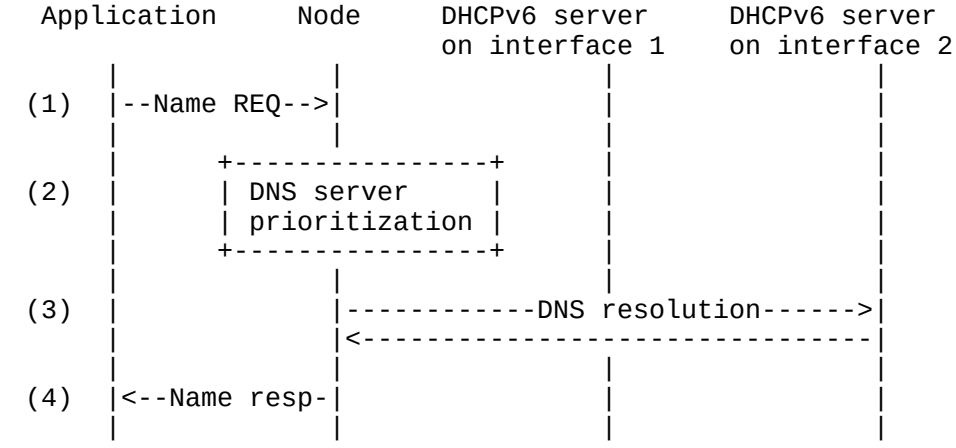
3. The node stores the learned DNS suffixes and IPv6 prefixes for later use

4. The node opens its seconds network interface 2

5. The node obtains DNS suffix, say 'example.com', and IPv6 prefix information, say '8.b.d.0.1.0.0.2.ip6.arpa' for the new interface 2 from DHCPv6 server

6. The node stores the learned DNS suffixes and prefixes for later use

Figure 7 below illustrates how a resolver uses the learned suffix information. Prefix information use for reverse lookups is not illustrated, but that would go as the figure 7 example.



Example on choosing interface based on DNS suffix

Figure 7

Flow explanations:

1. An application makes a request for resolving an FQDN, e.g. 'private.example.com'

2. A node creates list of DNS servers to contact to and uses configured DNS server information and stored DNS suffix information on priorization decisions.

3. The node has chosen interface 2, as from DHCPv6 it was learned earlier that the interface 2 has DNS suffix 'example.com'. The node then resolves the requested name using interface 2's DNS server to an IPv6 address

4. The node replies to application with the resolved IPv6 address

6. Scalability considerations

The size limitations of DHCPv6 messages limit the number of suffixes and prefixes that can be carried in a configuration option. Including the suffixes and prefixes in a DHCPv6 option is best suited for deployments where relatively few carefully selected suffixes and prefixes are adequate.

7. Considerations for network administrators

Network administrators deploying private namespaces should assist advanced hosts in the DNS server selection by providing information described in this document for nodes. To ensure nodes' routing and source and destination IP address selection also works correctly, network administrators should also deploy related technologies for that purpose.

The solution described herein is best for selecting a DNS server having knowledge of some namespaces. The solution is not able to make the right decision in a scenario where the same name points to different services on different network interfaces, as described in section 2.3. Network administrators are recommended to avoid overloading of namespaces in such manner.

To mitigate against attacks against local namespaces, administrators utilizing this tool should deploy DNSSEC for their zone.

8. Acknowledgements

The author would like to thank following people for their valuable feedback and improvement ideas: Mark Andrews, Jari Arkko, Marcelo Bagnulo, Stuart Cheshire, Lars Eggert, Tomohiro Fujisaki, Peter Koch, Suresh Krishnan, Edward Lewis, Kurtis Lindqvist, Arifumi Matsumoto, Erik Nordmark, Steve Padgett, Fabien Rapin, Dave Thaler, Margaret

Wasserman, Dan Wing, and Dec Wojciech. Ted Lemon and Julien Laganier receive special thanks for their contributions to security considerations.

This document was prepared using xml2rfc template and the related web-tool.

9. IANA Considerations

This memo includes a new DHCPv6 option that requires allocation of a new code point.

10. Security Considerations

It is possible that attackers might try to utilize OPTION_DNS_SERVER_SELECT option to redirect some or all DNS queries sent by a resolver to undesired destinations. The purpose of an attack might be denial-of-service, preparation for man-in-the-middle attack, or something akin.

Attackers might try to lure specific traffic by advertising DNS suffixes and prefixes from very small to very large scope or simply by trying to place attacker’s DNS server as the highest priority default server.

The main countermeasure against these attacks is to systematically prioritize more trusted DNS servers higher than less trusted ones. It is also implementation possiblity to listen for DNS server selection rules only on trusted interfaces. Additionally, resolvers SHOULD implement validating DNSSEC aware resolvers in order to be able to validate DNS responses received via any of its interfaces. Trusting on validation done by a DNS server is a possibility only if a host trusts the DNS server and can use a secure channel for DNS messages.

Decision on trust levels of network interfaces depends very much on deployment scenario and types of network interfaces. For example, unmanaged WLAN may be considered less trustworthy than managed cellular or VPN connections.

A node MAY also choose, or be configured, to obtain DNS server selection rules only from selected trusted interfaces, in which case it would be in the hands of administrators of these trusted interfaces whether or not to allow redirection, offloading, of DNS queries to untrusted interfaces (case 4 of figure 4).

A node that accepts DNS server selection rules from non-trusted interfaces and also implements DNSSEC validation SHOULD send queries also to (all) other known DNS servers in case a non-validatable response is received from the preferred DNS server. This protects against possible redirection attacks.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3152] Bush, R., "Delegation of IP6.ARPA", BCP 49, RFC 3152, August 2001.

[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.

[RFC3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, April 2004.

[RFC4242] Venaas, S., Chown, T., and B. Volz, "Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 4242, November 2005.

11.2. Informative References

[I-D.ietf-behave-dns64]
Bagnulo, M., Sullivan, A., Matthews, P., and I. Beijnum, "DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", draft-ietf-behave-dns64-11 (work in progress), October 2010.

[I-D.ietf-mif-problem-statement]
Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement", draft-ietf-mif-problem-statement-09 (work in progress), October 2010.

[I-D.ietf-v6ops-multihoming-without-nat66]
Troan, O., Miles, D., Matsushima, S., Okimoto, T., and D. Wing, "IPv6 Multihoming without Network Address Translation", draft-ietf-v6ops-multihoming-without-nat66-00 (work in

progress), December 2010.

[I-D.wing-behave-dns64-config]
Wing, D., "IPv6-only and Dual Stack Hosts on the Same Network with DNS64", draft-wing-behave-dns64-config-03 (work in progress), February 2011.

[RFC3397] Aboba, B. and S. Cheshire, "Dynamic Host Configuration Protocol (DHCP) Domain Search Option", RFC 3397, November 2002.

[RFC3646] Droms, R., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, December 2003.

[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.

[RFC5006] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Option for DNS Configuration", RFC 5006, September 2007.

Appendix A. Best Current Practice for DNS server selection

On some local namespace deployments explicit policies for DNS server selection are not available. This section describes ways for hosts to mitigate the problem by sending wide-spread queries and by utilizing possibly existing indirect information elements as hints.

A.1. Sending queries out on multiple interfaces in parallel

A possible current practice is to send DNS queries out of multiple interfaces and pick up the best out of the received responses. A host SHOULD implement DNSSEC in order to be able to reject responses that cannot be validated. Selection between legitimate answers is implementation specific, but replies from trusted servers should be preferred.

A downside of this approach is increased consumption of resources. Namely power consumption if an interface, e.g. wireless, has to be brought up just for the DNS query that could have been resolved also via cheaper interface. Also load on DNS servers is increased. However, local caching of results mitigates these problems, and a node might also learn interfaces that seem to be able to provide

'better' responses than other and prefer those - without forgetting fallback required for cases when node is connected to more than one network using local namespaces.

Another downside is revealing to all DNS servers the names a host is connecting to. For example, a DNS server of a public hotspot could learn all the private names host is trying to connect on other interfaces.

A.2. Search list option for DNS forward lookup decisions

A host can learn the special DNS suffixes of attached network interfaces from DHCP search list options; DHCPv4 Domain Search Option number 119 [RFC3397] and DHCPv6 Domain Search List Option number 24 [RFC3646]. The host behavior is very similar as is illustrated in the example at section 3.3. While these DHCP options are not intented to be used in DNS server selection, they may be used by the host as hints for smarter DNS server prioritization purposes in order to increase likelyhood of fast and successful DNS query.

Overloading of existing DNS search list options is not without problems: resolvers would obviously use the DNS suffixes learned from search lists also for name resolution purposes. This may not be a problem in deployments where DNS search list options contain few DNS suffixes like 'example.com, private.example.com', but can become a problem if many suffixes are configured.

A.3. More specific routes for reverse lookup decision

[RFC4191] defines how more specific routes can be provisioned for hosts. This information is not intented to be used in DNS server selection, but nevertheless a host can use this information as a hint about which interface would be best to try first for reverse lookup procedures. A DNS server configured via the same interface as more specific routes is more likely capable to answer reverse lookup questions correctly than DNS server of an another interface. The likelyhood of success is possibly higher if DNS server address is received in the same RA [RFC5006] as the more specific route information.

A.4. Longest matching prefix for reverse lookup decision

A host may utilize the longest matching prefix approach when deciding which DNS server to contact for reverse lookup purposes. Namely, the host may send a DNS query to a DNS server learned over an interface having longest matching prefix to the address being queried. This approach can help in cases where ULA [RFC4193] addresses are used and when the queried address belongs to a host or server within the same

network (for example intranet).

Authors' Addresses

Teemu Savolainen
Nokia
Hermiankatu 12 D
TAMPERE, FI-33720
FINLAND

Email: teemu.savolainen@nokia.com

Jun-ya Kato
NTT
9-11, Midori-Cho 3-Chome Musashino-Shi
TOKYO, 180-8585
JAPAN

Email: kato@syce.net

Multiple Interfaces (Mif)
Internet-Draft
Updates: 3484bis
(if approved)
Intended status: Experimental
Expires: September 14, 2011

J. Korhonen
Nokia Siemens Networks
T. Savolainen
Nokia

March 13, 2011

Controlling Traffic Offloading Using Neighbor Discovery Protocol
draft-korhonen-mif-ra-offload-01.txt

Abstract

This specification defines an extension to IPv6 Neighbor Discovery Protocol, which allows management of IPv6 traffic offloading to IPv4 and moving IPv4 traffic away from a specific interface. The specification updates the source and destination algorithms described in RFC 3484bis.

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 September 14, 2011.

Copyright Notice

Table of Contents

1. Introduction

This specification defines an extension to Neighbor Discovery Protocol [RFC4861], which allows management of IPv6 traffic offloading to IPv4 and moving IPv4 traffic away from a specific network connection.

The described solution is intended to be used during transition towards IPv6, during which time multi-interfaced hosts are often likely to have network interfaces with IPv4-only capability. A common scenario where coexistence of IPv4 and IPv6 network interfaces is expected to occur is when a smartphone has IPv6-enabled cellular connection and IPv4-only WLAN connection active at the same time.

This specification updates the source and destination algorithm described in RFC 3484bis [I-D.ietf-6man-rfc3484-revise]

2. Requirements and Terminology

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].

3. Problem Background

Current Internet hosts generally prefer IPv6 addresses over IPv4 addresses when performing source and destination address selections, as is recommended in [I-D.ietf-6man-rfc3484-revise].

A multi-interfaced host may have IPv6 enabled on a more ’expensive’ interface and a ’cheaper’ interface may have support only for IPv4. In such a scenario it might be desirable for hosts to prefer IPv4 in communication instead of IPv6.

The above mentioned problem can occur, for example, when a smartphone has simultaneously IPv6-enabled cellular connection ([I-D.korhonen-v6ops-3gpp-eps]) and IPv4-only WLAN connectivity active. When connecting to dual-stack capable destinations it would oftentimes be generally more efficient to use WLAN network interface. Furthermore, a cellular network operator may want hosts to offload traffic away from cellular network whenever hosts have alternate network accesses available.

Similar issue can arise also when a host has multiple interfaces with IPv4 connectivity. The cheaper interface should oftentimes be used for the communication, but it may not be clear for a host which one

of the available interfaces it should prefer.

4. Solution

This document introduces a new Neighbor Discovery option that a network can use to communicate 'lower-than-IPv4' preference for advertised prefix(es), and hence for host's IPv6 address, and also the level of router's willingness to act as an IPv4 default router.

The new Neighbor Discovery option was chosen to support hosts without DHCPv6 [RFC3315] support and also to work on networks not utilizing DHCPv6.

The new Neighbor Discovery option shall be phased out when IPv4 usage diminishes.

4.1. Neighbor Discovery Offload Option

This specification defines a new Neighbor Discovery [RFC4861] option called Offload (Type TBD) to be used in Router Advertisements. The option is illustrated in Figure 1. Router and hosts implementing this specification MUST understand the Offload option.

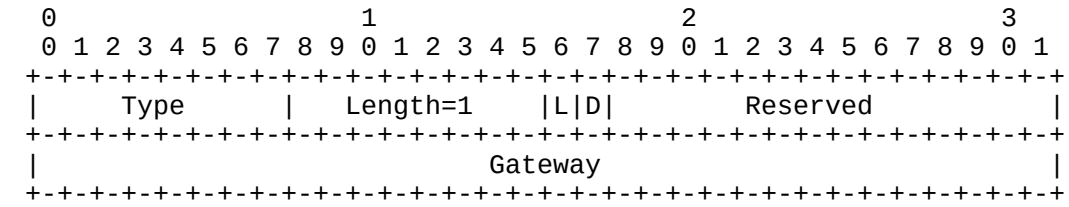


Figure 1: Router Advertisement Offload Option

Type

TBD by IANA.

Length

MUST be set to 1.

L (Lower-than-IPv4 Preference)

In addition to [RFC4191] defined Prf handling, the additional 'L' flag bit indicates 'Lower-than-IPv4' preference. From the [RFC4191] point of view, the 'Lower-than-IPv4' does not have any affect. The 'L' bit affects the source and destination address

selection for IPv6 addresses configured from prefixes advertised by the Router Advertisement containing the Offload option.

[*Discussion*: The 'L' flag has partially the same effect as setting the preferred lifetime to zero in the Prefix Information option. However, we did not want to change Prefix Information option configuration in the router as this option can then be used independently to control preferences, whether they are on or off.]

D (Default IPv4 Gateway Preference)

Indicates the willingness of the Dual-Stack capable router (who originated the Router Advertisement) to serve as a default gateway for the IPv4 traffic. If 'D' is unset (0) then the router indicates no specific to be or not to be a default gateway for IPv4 traffic. If 'D' is set (1) then the router explicitly indicates it is not willing to serve as a default gateway for IPv4 traffic if there are other usable gateways present in the same or other available interfaces.

Reserved

A 14-bit unused field. It MUST be initialized to zero by the sender and MUST be ignored by the receiver.

Gateway

The address of the dual-stack router's IPv4 interface used as the next-hop from hosts point of view for sending and receiving IPv4 traffic on this link. The IPv4 address MUST belong to the same interface that originated the Router Advertisement containing this option. If the router is IPv6 only or the 'D' bit unset (0), then this field MUST be set to unspecified address (0.0.0.0).

The behavior of 'lower-than-IPv4 Preference' (see Section 4.2) and 'Default IPv4 Gateway Preference' (see Section 4.3) are discussed in more detail in the following sections. The Offload option is only used in Router Advertisement messages.

4.2. Lowering the Preference of IPv6 Default Addresses

Router originating a Router Advertisement (RA) with the 'L' bit set in the Offload option indicates that it SHOULD NOT be used for forwarding IPv6 traffic for destinations that are also reachable with IPv4 (via other interfaces) or IPv6 destinations are also reachable using other interfaces.

If a host implements this specification, the 'L' bit required

behavior can be achieved either by host implementation dependent means (especially relevant in cellular hosts that have a per application 'silo view' of the networking stack) or by modifying the [I-D.ietf-6man-rfc3484-revise] policy table and the default address selection algorithm.

In the latter case, the [I-D.ietf-6man-rfc3484-revise] is modified so that the addresses configured from the prefixes advertised by the 'lower-than-IPv4 Preference' router MUST be treated as 'deprecated' if there is no more specific route for the intended IPv6 destination (i.e. other than default router for the intended destination would be selected). The modification concerns both the source address selection algorithm and the destination address selection algorithm. The expected behavior is that other available preferred IPv6 addresses get selected over 'lower-than-IPv4' IPv6 addresses and even IPv4 destinations are preferred over IPv6 destinations when only 'lower-than-IPv4' IPv6 addresses are available as IPv6 source addresses.

4.3. Lowering IPv4 Default Router Preference

The 'D' flag bit in the Offload option indicates the willingness of the Router Advertisement originating Dual-Stack capable router to serve as a default gateway for IPv4 traffic. When 'D' is unset (0), the router does not indicate any preference of being or not being a default gateway for IPv4 traffic. If 'D' is set (1), the router indicates that it SHOULD NOT be used as a default gateway for IPv4 traffic, if other default gateways are present in the same or other available interfaces. The 'Gateway' field in the Offload option contains the IPv4 address of the Dual-Stack interface that originated the Router Advertisement. The address serves as the identification of the next-hop IPv4 routers.

4.4. Offload Lifetime

The lifetime of the [I-D.ietf-6man-rfc3484-revise] modifications and IPv4 default gateway preferences caused by the Offload option are tied to the lifetime indicated in the Router Advertisement. Also, if the router sends a new Router Advertisement without the Offload option before the router lifetime expires, it is an indication to the receiving hosts that any existing Offload option caused state/ information MUST be removed.

5. Router Behavior

A router configuration SHOULD allow network administrator to add and configure this option into Router Advertisement messages. The

configuration can be selectively enabled (the Offload option is included in the Router Advertisement) or disabled (the Offload option is not included in the Router Advertisement).

6. Host Behavior

A multi-interface capable host SHOULD monitor presence of this option in received Router Advertisement messages. When the Offload option is received, the source and destination selection algorithms defined in [I-D.ietf-6man-rfc3484-revise] shall be temporarily modified as described in Section 4.2. The IPv4 source address selection and default gateway preferences shall temporarily be updated as described in 4.3.

If the host receives a Router Advertisement without the Offload option and there is an existing state created by an earlier received Offload option, then the host MUST remove all default address selection algorithm and IPv4 default gateway preferences modifications. The removals concerns the prefixes configured from router where the router advertisement was received.

7. Modification to Default Address Selection

The 'lower-than-IPv4 Preference' affects the Source Address Selection Rule 3. The notation Lower(SA) returns true if the address SA was configured from the prefixes advertised by a 'lower-than-IPv4 Preference' router. Lower(SA) returns false is the address SA was configured from prefixes advertised by other than 'lower-than-IPv4 Preference' router. The notation Default(D) returns false if the address D has more specific routes (i.e. other than the default route). Default(D) returns true if the address D points only to a default route. The modified Rule 3 would be as follows:

Rule 3: Avoid deprecated addresses.

> The addresses SA and SB have the same scope. If Lower(SA) == true and Default(D) == true, then mark SA temporarily as "deprecated". If Lower(SB) == true and Default(D) == true, then mark SB temporarily as "deprecated". If one of the two source addresses is "preferred" and one of them is "deprecated" (in the [RFC4862] sense), then prefer the one that is "preferred."

Similar modification also concerns the Destination Address Selection Rule 3 when checking whether a candidate source address for a given destination is deprecated.

8. Security Considerations

The Offload option allows malicious hosts and routers to affect a victim host's next hop and default address selection if spoofing of Router Advertisements are possible on the access link. This is a well-known and understood security threat [RFC3756] and can be mitigated using, for example, Secure Neighbor Discovery [RFC3971].

9. IANA Considerations

This specification defines a new Neighbor Discovery option described in Section 4.1.

10. References

10.1. Normative References

[I-D.ietf-6man-rfc3484-revise]
Matsumoto, A., Kato, J., and T. Fujisaki, "Update to RFC 3484 Default Address Selection for IPv6", draft-ietf-6man-rfc3484-revise-01 (work in progress), October 2010.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

10.2. Informative References

[I-D.korhonen-v6ops-3gpp-eps]
Korhonen, J., Soininen, J., Patil, B., Savolainen, T., Bajko, G., and K. Iisakkila, "IPv6 in 3GPP Evolved Packet System", draft-korhonen-v6ops-3gpp-eps-06 (work in progress), February 2011.

[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.

[RFC3756] Nikander, P., Kempf, J., and E. Nordmark, "IPv6 Neighbor

Discovery (ND) Trust Models and Threats", RFC 3756, May 2004.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

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

Appendix A. Address selection examples

Link-local addresses are omitted in all following examples. The assumption is that possible destinations have a global scope and all IPv6 enabled interfaces have at least one global scope IPv6 address. Therefore, the default address selection would always output global scope addresses over link-local addresses.

A.1. Case 1: IPv6-only cellular and IPv4-only WLAN accesses

A host has obtained global IPv6 address, 2001:db8::2, on a cellular interface and with it has received Neighbor Discovery option with 'lower-than-IPv4' preference. The host also has global IPv4 address, 192.0.2.2, on a WLAN interface.

When connecting to a dual-stack enabled destination, both 2001:db8::2 and 192.0.2.2 are considered as source addresses candidates. IPv4 address is selected, because 2001:db8::2 is considered deprecated. Hence host uses WLAN for communication.

When connecting to IPv6-only destination, 2001:db8::2 is selected and cellular network used, as there are no other IPv6 addresses available.

A.2. Case 2: WLAN access with multiple prefixes

A host has obtained two global IPv6 addresses, one of which was from a router indicating 'lower-than-IPv4' preference. For example, 2001:db8:1::2 from router with 'lower-than-IPv4' preference and 2001:db8:2::3 from router without any special preferences.

When connecting to IPv6-only destination, both addresses are considered as source address candidates. Source address selection chooses 2001:db8:2::3 as 2001:db8:1::2 is considered deprecated (Lower(2001:db8::2) == true and Default(D) == true).

A.3. Case 3: WLAN and cellular interface with cellular's IPv4 not default route

A host has obtained IPv6 address, 2001:db8::2, and IPv4 address, 192.0.2.2, from cellular network. The network has indicated 'lower-than-IPv4' preference for IPv6 and 'not your default router' for IPv4. The host also has dual-stack WLAN access with 2001:db8:1::3 and 192.0.2.30 addresses.

When connecting to IPv4-only destination, host selects 192.0.2.30 as source address because default gateway on the interface of 192.0.2.2 address is 'not default gateway'. WLAN is used for communication.

When connecting to IPv6-only destination, host selects 2001:db8:1::3 from WLAN interface as the 2001:db8::2 is considered deprecated (Lower(2001:db8::2) == true and Default(D) == true). WLAN is used for communication.

When connecting to dual-stack destination, host selects from the four candidate addresses 2001:db8:1::3, as IPv6 is preferred in general and as that address is not deprecated. WLAN is used for communication.

A.4. Case 4: Dual-stack cellular access

A host has obtained IPv6 address, 2001:db8::2, and IPv4 address, 192.0.2.2, from cellular network. The network has indicated 'lower-than-IPv4' preference.

When connecting to a dual-stack enabled destination, both addresses are considered as candidate source addresses. IPv4 address is chosen, because IPv6 address is considered deprecated.

A.5. Case 5: Dual-stack cellular and single stack WLAN

A host has obtained IPv6 address, 2001:db8::2, and IPv4 address, 192.0.2.2, from cellular network. The network has indicated 'lower-than-IPv4' preference for IPv6 and 'not your default router' for IPv4. The host also has WLAN access with 192.0.2.30 address.

When connecting to dual-stack destination, all three addresses are considered as source address candidates. The IPv4 address from WLAN, 192.0.2.30, is selected as the IPv6 address, 2001:db8::2, is considered deprecated and as the IPv4 default route points to WLAN. Hence WLAN is used for communication.

A.6. Case 6: Coexistence with RFC4191

A host has obtained IPv6 address, 2001:db8:1::2/64 from cellular network. The network has indicated 'lower-than-IPv4' preference for IPv6 and a more specific route to 2001:db8:2::/48. The host also has IPv6 WLAN access with 2001:db8:3::3/64 address.

When connecting to 2001:db8:2::1 the host selects 2001:db8:1::2 from cellular interface as a source address, because Lower(2001:db8:1::2) == true and Default(2001:db8:2::1) == false and hence the 2001:db8:1::2 is not considered as deprecated address even though 'lower-than-IPv4' preference was advertised.

When connecting to 2001:db8:4::1 the host selects 2001:db8:3::3 from WLAN interface as a source address, because Lower(2001:db8:2::1) == true and Default(2001:db8:3::3) == true) and hence 2001:db8:2::1 is considered as deprecated address.

Authors' Addresses

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
FI-02600 Espoo
Finland

Email: jouni.nospam@gmail.com

Teemu Savolainen
Nokia
Hermiankatu 12 D
FI-33720 Tampere
Finland

Email: teemu.savolainen@nokia.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 14, 2011

D. Liu
China Mobile
Yuri. Ismailov
Ericsson
Z. Cao
China Mobile
March 14, 2011

Socket API Extension for MIF Host
draft-liu-mif-api-extension-04

Abstract

API (Application Program Interface) is an interface implemented by a software program that enables it to interact with other software. In IP network communication area, Socket API is the de facto standard and is widely used in many systems. Recently, multiple interfaces or multiple connection capable devices become more and more common. But due to the limitation of the default route of the host and other issues such as DNS selection etc prevent the utilization of multiple interfaces/connections benefit. Moreover, there is no API level support for the application developer to utilize the benefit of the host's multiple interfaces/connections. Starting with the requirement of MIF API extension, this document describes a new set of abstraction APIs to provide additional services to applications running on hosts attached to multiple provisioning domains. These services could assist advanced applications in having greater control over first-hop, source address and/or DNS selection issues.

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 September 8, 2011.

Copyright Notice

Table of Contents

1. Introduction

API is an software interface which is used to interact with other softwares. In IP network area, Socket API is widely used in many operating systems.

Socket API has been extended according to the evolution of TCP/IP protocol stack. For example, Socket API was extended to support IPv6 in RFC 3493 [RFC3493]. Nowadays, there are more and more devices that have the ability to attach to multiple provisioning domains. Due to the limitation of Socket API and other issues such as default route and DNS selection, the application can not directly control the multiple interfaces/connection capability of the devices. This document describes a new set of abstraction APIs to provide additional services to applications running on hosts attached to multiple provisioning domains. These services could assist advanced applications in having greater control over first-hop, source address and/or DNS selection issues.

2. Conventions used in this document

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 [RFC2119].

3. MIF API Concept

Figure 1 demostrates the MIF API concept. MIF API is mostly extended based on current Socket API. MIF API is an abstract API which describes the abstract function and not depend on specific platform. MIF provides the following function:

1.Address selection

MIF host may has multiple IP addresses. for example, different interface may have different IP addresses been configured. Further more, some protocols, for example: mobile IP protocol may assign home address to the host. Application need to control which IP address it want to use. MIF API may provide the address selection function in this case.

2.Routing selection

Noramlly, host only has a default route which points to one default gateway. for MIF host, there may be different connections using multiple interfaces and each connection has its own default gateway.

MIF API provide the capability for the MIF host to control the first hop route.

3. DNS Selection

The MIF host may have the DNS split issues. MIF API may provide the capability for the application to select and control which DNS server it want to use.

4. Connection Manager API

Connection Manager is widely used in current mobile terminals. The lack of standard guide of connection manager will lead to non-consistency of user experience. Connection manager API aims to provide a set of abstract function description of connection manager.

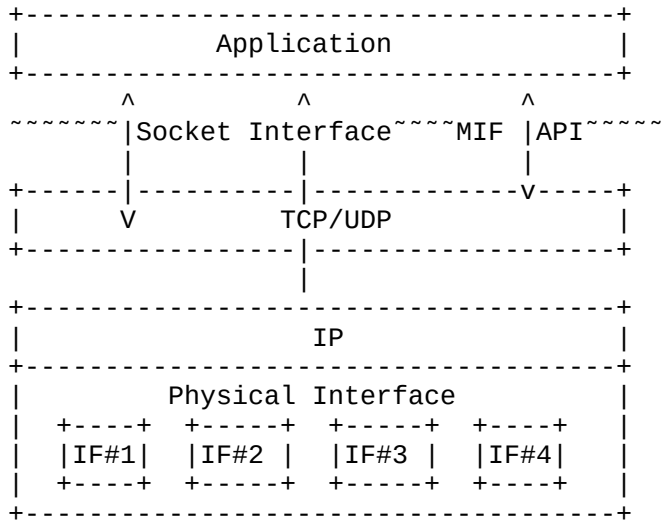


Figure 1. MIF API Concept

Figure 1

4. API extension for MIF host

This section discusses the possible ways to design the MIF API.

There are works in IETF relatd to socket API extensions,such as: IPv6 socket API extension, HIP socket API extension, shim6 socket API

extensions. This section gives initial proposals regarding to how this MIF Socket APIs should be designed.

4.1. MIF API Requirement

R1: Compatible with existing Socket APIs

The extension of Socket APIs should remain compatible with current existing Socket APIs which means this extension will not affect the existing application and the application developer can chose whether to use this extension or not.

R2: Support for multiple Connections

The newly introduced APIs can support the host applications to utilize multiple connections. If some connections fail to work, applications can failover to other available interfaces.

R3: No Changes to Existing Socket APIs

The newly introduced APIs cannot change the behavior of existing network APIs available on the host operating system.

R4: Easy for Implementation

The implementation details for the newly introduced APIs should be as neat as possible. It keeps the underlying implementation details simple, and makes the application developers easy to use these APIs.

4.2. Definition of New Socket APIs

This section gives the definition and description of MIF API.

Address Selection API

In current Socket API definition, setsockopt() is used to set the option of socket. It has the following definition:

```
int setsockopt(int socket, int level, int option_name, const void
*option_value, socklen_t option_len);
```

The setsockopt() function shall set the option specified by the option_name argument, at the protocol level specified by the level argument, to the value pointed to by the option_value argument for the socket associated with the file descriptor specified by the socket argument.

The option_name argument specifies a single option to set. The

option_name argument and any specified options are passed uninterpreted to the appropriate protocol module for interpretations. The options are as follows:

SO_DEBUG

SO_DONTROUTE

SO_SNDBUF

.....

This document proposes to extend the current setsockopt() function as follows:

Extend new option type used for address selection:

SO_ADDR

When setsockopt() set this parameter, the application will use the IP address specified by option_value as the source IP address.

Route Selection API

Extend new option type parameter for setsockopt() function:

SO_FSTHOP

When this option is set, the packet will send via the gateway that specified by option_value.

DNS Selection API

gethostbyname() and gethostbyaddr() is the APIs used for DNS queries. Current version of gethostbyname() has the following definition:

struct hostent *gethostbyname(const char *name)

This document proposes to extend this function as follows:

struct hostent *gethostbyname(const char *name, struct sockaddr *DNS_addr)

DNS_addr is used to specify which DNS server the application want to use for the DNS query.

4.3. Connection Manager API extension

Connection manager is a common practice for MIF host to perform provisioning domain selection and interface configuration [I-D.draft-seite-mif-connection-manager]. High level API of connection manager may differ across different operating systems and/or platforms. It is hence preferred to have more consistent connection manager API specification and this will be beneficial for both applications developers,users and operators.

Get Interface status/configuration

Connection Manger may need to get the interface's status and its configration parameters, for example, the connection manager in MIF host may need to know all the interface's current status(e.g. on/ off), interface type,IP address etc.

A new structure is defined in this document:

Struct InterfaceInfo {

char * name; /*interface name*/

struct parameter; /*IP address, address type, address scope(e.g. HoA, CoA, temp address),MAC address,Interface type (WiFi, cellular) etc*/

uint32_t status; /*interface status*/

}

function struct Interfaceinfo getInterfaceInfo()is used to get the status and parameters of an interface.

Function setInterfaceInfo(struct InterfaceInfo parameter) is used to set parameters associated with a specific interface.

5. Using the extended APIs

To enjoy the benefit of multiple interfaces, the developers can write programs according to the APIs introduced in this document. Detail example need to be further specified.

6. Security Considerations

TBD

7. IANA Considerations

None

8. Acknowledgments

The authors want to thank Teemu Savolainen from Nokia, Dayi Zhao from Bitway and others for their useful suggestions and discussions.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[I-D.scharf-mptcp-api] Scharf, M. and A. Ford, "MPTCP Application Interface Considerations", draft-scharf-mptcp-api-02 (work in progress), July 2010.

[RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, February 2003.

Authors' Addresses

Dapeng Liu
China Mobile
Unit2, 28 Xuanwumenxi Ave,Xuanwu District
Beijing 100053
China

Email: liudapeng@chinamobile.com

Yuri Ismailov
Ericsson
Stockholm
Sweden

Email: yuri@ismailov.eu

Zhen Cao
China Mobile
Unit2, 28 Xuanwumenxi Ave,Xuanwu District
Beijing 100053
China

Email: caozhen@chinamobile.com