Happy Eyeballs Extension for Multiple Interfaces
draft-chen-mif-happy-eyeballs-extension-02

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 January 12, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must
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 method 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 defined framework, interface weighting consideration and and computation process.

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 an indication to identify IPv4 or IPv6 family has been preferred.

2.2. Interface Weighting Consideration

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 by taking various factors into account. The impact factors could be categorized in two groups.

- Hard preconditions: It’s mandatory indications that interface behaviour should comply with such preconditions guidance. The following factors belong to hard preconditions.
  - Operator policies: operators would deliver the customized policies in particular network environments due to charging or area regulation considerations.
  - User preferences: Users might configure to enable a specific interface to access network. For example, user may choose wifi interface to surf Internet considering low cost.

- Soft preconditions: It’s optimal choice for transmitting data packages through a specific interface compared to others. The following factors would contribute soft preconditions justification.
  - Routing policies: DHCPv6 Route Option[draft-ietf-mif-dhcpv6-route-option-01] and RFC4191[RFC4191] allow configuration of specific routes and influence a nodes’ ability to pick an appropriate route to a destination. A weighting for an interface headed to
destination address that matches a specific route would be increased.

* DNS selection: if improved DNS server selection [draft-ietf-mif-dns-server-selection-03] takes effects, the weighting for those interfaces over which DNS suffix matching the requested name should also be increased.

* Other factors: There are many other factors could contribute optimal interface selection. This documents would like to focus on the main ones and treat others in a best effort manner. The key factors are expected to be added in future discussion.

2.3. Interface Selection Process

The selection of a particular interface from the viable set implies a selection of one particular network path in preference to other viable paths. Interface weighting must be computed in advance but also be recomputed during session. The whole process for interface selection could be divided into two stages.

At stage I, upon the connection attempt, interface set should be filtered through the hard preconditions, and then aggregate the results within that kind of "policy group".

At stage II, the soft preconditions should be applied to the resulted interface set. According to particular soft preconditions, the preferred interface would be chosen by increasing I and delaying the connection attempts on the "undesirable" interfaces. This would allow to dial the preference between the different interfaces. The less desirable interface would get penalised a-priori.

To be specific, when one interface defeats others, the corresponding value I will be set to positive value. Other interfaces will be set negative value. A value of 0 indicates equal weight for multiple interfaces.

When interface values I have been configured, the traffic flow targeted to specific destination address or hostname will follow this guidance to choose proper interface. Hence, initial connection attempt would be sent over the interface that has matching particular rules and other interfaces would be tried only if no reply on the preferred one. Network condition may change during the session, interface reselection should be triggered. 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.

2.4. IPv4/IPv6 Selection Algorithm for Individual Interface

For a specific interface in a dual-stack single interface node, the choice of IP address family relies on Happy Eyeballs algorithm, which is defined in [HAPPY-EYEBALLS].

3. Additional Considerations

3.1. Usage Scope

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

3.2. Flow Continuity

Usually, interface changing happens at the beginning of new session. So, there is no flow continuity issues for ongoing TCP session. Dynamic movement of traffic flows are addressed by other IETF protocols as well.

3.3. Default Address Selection

If more than one IPv6 address is assigned to the interface, the native IPv6 address is given preference.

4. IANA Considerations

This memo includes no request to IANA.

5. Security Considerations

TBD

6. Normative References

[HAPPY-EYEBALLS]

[MIF-PS] Blanchet, M., "Multiple Interfaces and Provisioning"
Domains Problem Statement",


[draft-ietf-mif-dhcpv6-route-option-01]
Dec, W., "DHCPv6 Route Option",
draft-ietf-mif-dhcpv6-route-option-01 (work in progress), March 2011.

[draft-ietf-mif-dns-server-selection-03]

Authors’ Addresses

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

Email: chengang@chinamobile.com

Carl Williams
Consultant
El Camino Real
Palo Alto, CA 94306
USA

Email: carlw@mcsr-labs.org
Abstract

This document describes an abstract API that provides the minimal functionality required for a program to communicate effectively with peers and services on the network while running on a host that has more than one active network interface. This API is abstract: we describe the functionality that must be provided, not the bindings that should be used to provide that functionality. The functionality described here provides the building blocks from which higher-level APIs might be built, and is not intended to be used directly by typical applications.
This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.  Introduction . . . . . . . . . . . . . . . . . . . . . . . . .  4
2.  Conventions used in this document . . . . . . . . . . . . . .  4
3.  MIF API Concept . . . . . . . . . . . . . . . . . . . . . . .  5
3.1.  Provisioning Domains . . . . . . . . . . . . . . . . . . .  5
3.2.  Provisioning Domain Agnosticism . . . . . . . . . . . . .  5
3.3.  MIF API Elements . . . . . . . . . . . . . . . . . . . . .  6
3.3.1.  Application Element . . . . . . . . . . . . . . . . . .  6
3.3.2.  High Level API . . . . . . . . . . . . . . . . . . . . .  7
3.3.3.  MIF API . . . . . . . . . . . . . . . . . . . . . . . .  7
3.3.4.  Communications API . . . . . . . . . . . . . . . . . .  7
3.3.5.  Network Link API . . . . . . . . . . . . . . . . . . .  7
3.4.  MIF API communication model . . . . . . . . . . . . . . .  8
3.4.1.  POST MESSAGE call . . . . . . . . . . . . . . . . . .  8
3.4.2.  CHECK MESSAGE call . . . . . . . . . . . . . . . . .  8
3.4.3.  GET MESSAGE call . . . . . . . . . . . . . . . . . . .  8
3.5.  MIF Messages . . . . . . . . . . . . . . . . . . . . . . .  8
3.5.1.  Announce Interfaces . . . . . . . . . . . . . . . . . .  9
3.5.2.  Stop Announcing Interfaces . . . . . . . . . . . . . .  9
3.5.3.  Interface Announcement . . . . . . . . . . . . . . . .  9
3.5.4.  No Interface Announcement . . . . . . . . . . . . . .  9
3.5.5.  Announce Provisioning Domain . . . . . . . . . . . . .  9
3.5.6.  Stop Announcing Provisioning Domains . . . . . . . . . 10
3.5.7.  Provisioning Domain Announcement . . . . . . . . . . 10
3.5.8.  No Provisioning Domain Announcement . . . . . . . . . 10
3.5.9.  Announce Configuration Element . . . . . . . . . . . . 10
3.5.10. Configuration Element Announcement . . . . . . . . . 10
3.5.11. No Configuration Element Announcement . . . . . . . . 11
3.5.12. Announce Address . . . . . . . . . . . . . . . . . . . 11
3.5.13. Address Announcement . . . . . . . . . . . . . . . . . 12
3.5.14. No Address Announcement . . . . . . . . . . . . . . . 12
3.5.15. Get Configuration Data . . . . . . . . . . . . . . . . . 12
3.5.16. Translate Name . . . . . . . . . . . . . . . . . . . . . 12
3.5.17. Stop Translating Name . . . . . . . . . . . . . . . . . 13
3.5.18. Name Translation . . . . . . . . . . . . . . . . . . . . 13
3.5.19. Connect to Address . . . . . . . . . . . . . . . . . . . 13
3.5.20. Connect to Address From Address ........................ 13
3.5.21. Connected ...................................................... 14
3.5.22. Not Connected .................................................... 14
4. Example Usage ....................................................... 14
5. Security Considerations ........................................... 16
6. IANA Considerations ............................................... 16
7. Acknowledgments .................................................... 16
8. References ............................................................. 16
  8.1. Normative References ........................................ 16
  8.2. Informative References ..................................... 16
Authors’ Addresses .................................................... 17
1. Introduction

Traditionally, hosts that communicate on the network have done so over a single network link, which is provided by a single service provider. This simple environment is relatively easy to program to, and relatively predictable.

However, this relatively simple case is no longer the norm. A typical modern host may have one or two wireless interfaces: a wireless interface connected to a broadband network, and possibly another connected to some kind of cellular network. The same host may also have a wired interface which is sometimes connected to another broadband link. It is also quite common for hosts to have VPN links that are configured, for example, for access to corporate networks, or for access to network privacy services.

As a result, it is now quite typical that a program attempting to communicate in such an environment will be presented with conflicting configuration information from more than one provider. In addition, the cost of bandwidth on different links and the power required by those links may require consideration.

The API specified in this document is intended to describe the minimal complete set of API calls required to implement higher level APIs that solve these problems. It is not expected that applications will be implemented to this API, although it should be possible to do so. Rather, we expect this API to be used as a basis for building higher-level APIs that provide domain-specific solutions to these problems. The reason for specifying a lower-level API is to enable any arbitrary domain-specific API to be implemented, since no single higher-level API is likely to satisfy the needs of every application.

The API specified here is an abstract API. This means that we specify the functionality that is required to implement the API, but we do not provide specific bindings for any programming language: these are left up to the implementation. The API is described in terms of messages sent and messages received, rather than in terms of procedure calls, because it is necessary to be able to interleave these messages; a procedure call API necessarily precludes interleaving.

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

The MIF API is intended to deal with situations where more than one interface may be active at a time. It must also deal with situations where a single interface is connected to a link that provides more than one type of network service. The most common example of this that we expect is a dual-stack network configuration.

3.1. Provisioning Domains

To properly handle these multiple-service interfaces, we specify the API not in terms of interfaces, but in terms of provisioning domains. So in the case of a dual-stack network attached to a single network interface, there would be two provisioning domains. If the host has a second interface that is connected to a link that only supports IPv6 service, then that host would be connected to a total of two network links, but three provisioning domains.

From the perspective of the MIF API, a provisioning domain consists of a link, plus all the configuration information received on that link for that provisioning domain. So for an IPv4 provisioning domain, that would be whatever information is received from the DHCP server. For an IPv6 provisioning domain, the information received through router advertisements would be combined with the information received via DHCPv6.

**point of discussion: it’s actually possible to have two separate provisioning domains for IPv6 on the same wire. Is this a case that could happen in practice, and that we ought to support? I know that some asian countries have arrangements where the operator of the physical network is distinct from one or more operators who provide transit; I think this is all handled transparently to the host, but I don’t really know the details.

**point of discussion: is IPv4 stateless/Bonjour a separate provisioning domain? What about IPv6 ULA?

3.2. Provisioning Domain Agnosticism

Although it is possible that a high-level API built on top of this API may be able to distinguish between provisioning domains, at the level of this API, no such distinction can be made. Each provisioning domain is treated separately, and it is the responsibility of the higher-level API or of the application to decide which provisioning domain or domains to actually use.
3.3. MIF API Elements

There are a number of different, essentially independent, pieces of software that need to be connected together in order to fully support a successful MIF communication strategy. These elements are shown in figure 3.1.

![Diagram of MIF API Elements]

3.3.1. Application Element

This is an actual application. Applications fall into a variety of broad categories, including network servers, web browsers, peer-to-peer programs, and so on. Although we are focusing here on the mechanisms required to allow these applications to originate connections to remote nodes, it is worth noting that applications must also be able to receive connections from remote nodes.
3.3.2. High Level API

Applications are generally expected to originate connections using some general-purpose high-level API suited to their particular function. It is likely that different applications may use different high-level APIs to communicate, depending on their particular needs. We do not describe the functioning of such high-level APIs; however, one such API under current consideration is the Happy Eyeballs for MIF [reference]. These APIs are expected to be able to be implemented using functionality like that described in the MIF API.

3.3.3. MIF API

This is the API being described in this document. Generally speaking, this API is used by higher-level APIs. However, it is permissible for applications to use the MIF API when it is deemed necessary. Currently, several modern web browsers take this approach to establishing network connections, rather than relying on vendor-provided connection mechanisms.

3.3.4. Communications API

Once an application has originated a connection with a remote node using either a high-level API or the MIF API, it must communicate. Similarly, when an application receives a connection from a remote node, it must communicate with that remote node. The communications API is used for this communication. Popular examples of such APIs include the POSIX socket API and a variety of other related APIs.

It is likely that in some instances, implementations of the MIF API will be done as extensions to the Communications API provided by a particular operating system; the functional separation we show here is intended to allow us to illustrate only those features required in a MIF environment, while relying on existing communications APIs to provide the rest.

3.3.5. Network Link API

This is the software that is responsible for actually managing whatever network links are present on a node, whether these are physical links or tunnels. What precisely this functional box contains may vary greatly from device to device. On a typical modern computer workstation, this functionality would almost certainly reside entirely in the system kernel; however, on an embedded device everything from the Application down to the Network Link API could easily be running together on the bare metal as a single program.

The Network Link API can completely concealed from the Application,
so we don’t show a connection between them on the functional diagram, and indeed we do not talk about the functionality provided by this API. The reason for showing it on the functional diagram is simply to show that there likely is an API in common between MIF and the Communications API.

3.4. MIF API communication model

MIF API requests are made in the form of messages posted to the MIF API, and messages received from it. To accomplish this, several API calls are available. These calls mediate communication between the MIF API and the High Level API, or between the MIF API and the Application. In addition, the CHECK MESSAGE call allows the application to probe for or wait for messages from any of the APIs.

3.4.1. POST MESSAGE call

This call causes a message to be posted to the MIF API. The call posts the message, and then returns.

3.4.2. CHECK MESSAGE call

This call checks to see if there is a message waiting either from the High Level API, the MIF API, or the Communications API. Ideally it should be able to report the availability of any message or event that the application might anticipate receiving, so that the application can simply block waiting for such an event using this call. The application should be able to do a non-blocking probe, wait for some limited period of time, or wait indefinitely.

An example of a function of this type in existing practice is the POSIX poll() system call.

3.4.3. GET MESSAGE call

This call checks to see if there is a message waiting. If there is no message, it returns a status code indicating that there is no message waiting. If there is a message, it returns the message.

3.5. MIF Messages

MIF messages always go in one direction or the other: from the subscriber to the MIF API, or to the subscriber from the MIF API. We use the term "subscriber" here to mean either the Application or the High Level API, since either is permitted to communicate with the MIF API.

Messages described here are grouped according to function.
3.5.1. Announce Interfaces

This message is sent to the MIF API to ask it to send a message announcing the existence of any interface. When the MIF API receives this message from a subscriber, it iterates across the list of all known interfaces; for each known interface, it sends an Interface Announcement message to the subscriber.

In addition, the MIF API sets a flag indicating that the subscriber is interested in learning about new interfaces. When the MIF API detects the presence of a new interface, it sends an Interface Announcement message for that interface to the subscriber. This would happen, for instance, when a new tunnel is configured, or when a USB device that is a network interface is discovered by the Network API.

Also, if a network interface goes away, either because the physical network device is disconnected, or because a tunnel is disabled, the MIF API will send a No Interface Announcement message to the subscriber.

3.5.2. Stop Announcing Interfaces

This message is sent to the MIF API when a subscriber is no longer interested in receiving announcements about new interfaces. Subsequently, the MIF API will no longer send Interface Announcement or No Interface Announcement messages to the subscriber.

3.5.3. Interface Announcement

This message announces the existence of an interface. The announcement includes an interface display name and interface identifier.

3.5.4. No Interface Announcement

This message announces that an interface that had been previously announced is no longer present. The announcement includes the interface identifier.

3.5.5. Announce Provisioning Domain

This message requests the MIF API to announce the availability of any provisioning domains configured on a particular interface. The interface identifier must be specified.

Upon receipt, the MIF API will iterate across the list of Provisioning Domains present for a particular interface, and will
send a Provisioning Domain Announcement for each such Provisioning Domain.

In addition, the MIF API will set a flag indicating that the subscriber wishes to know about new provisioning domains as they appear. Subsequently, when a new Provisioning Domain appears, the MIF API will send a Provisioning Domain Announcement message to the subscriber.

Finally, if a Provisioning Domain expires or is invalidated, the MIF API will send the subscriber a No Provisioning Domain Announcement message for that Provisioning Domain.

In the event that an interface on which provisioning domains has been announced goes away, a No Provisioning Domain Announcement message will be sent for each provisioning domain that had previously been announced on that interface before the No Interface Announcement message is sent.

Once a No Interface Announcement message has been sent, any subscriber that had subscribed to Provisioning Domain announcements for that interface will be automatically unsubscribed.

3.5.6. Stop Announcing Provisioning Domains

This message requests that the MIF API stop sending the subscriber Provisioning Domain Announcement and No Provisioning Domain Announcement messages. The subscriber must indicate the interface for which it no longer wishes to receive Provisioning Domain announcements.

3.5.7. Provisioning Domain Announcement

This message is sent by the MIF API to the subscriber to indicate that a new Provisioning Domain has successfully been configured on an interface. The announcement includes the interface identifier and the provisioning domain identifier.

3.5.8. No Provisioning Domain Announcement

This message is sent by the MIF API to the subscriber to indicate that an existing, previously announced provisioning domain has expired or otherwise become invalid, and can no longer be used.

3.5.9. Announce Configuration Element

This message is sent by the subscriber to request a specific configuration element from a specific provisioning domain. A
provisioning domain identifier must be specified.

The MIF API will respond by iterating across the complete list of configuration elements for a provisioning domain, sending a Configuration Element Announcement message to the subscriber for each one.

Additionally, if any Configuration Elements subsequently complete for a particular provisioning domain, the MIF API will send a Configuration Element Announcement message to the subscriber for each such element. If a Configuration Element becomes invalidated after it has been announced, the MIF API will send a No Configuration Element message.

If a provisioning domain expires or becomes invalid, the MIF API will iterate across the list of remaining configuration elements for that provisioning domain and send a No Configuration Element Announcement message for each such configuration element.

3.5.10. Configuration Element Announcement

The Configuration Element Announcement message includes a Provisioning Domain ID and a Configuration Element Type, which can be one of the following:
- Config Element RA
- Config Element DHCPv6
- Config Element DHCPv4
  ...TBD...

3.5.11. No Configuration Element Announcement

The No Configuration Element Announcement message indicates that a previously valid configuration element for a provisioning domain is no longer valid. The message includes a provisioning domain identifier and a configuration element type.

3.5.12. Announce Address

This message is sent by the subscriber to request announcements of valid IP addresses for a specific provisioning domain. A provisioning domain identifier must be specified.

The MIF API will respond by iterating across the complete list of configuration elements for a provisioning domain, sending a Address Announcement message to the subscriber.

Additionally, if any new Address is subsequently configured on a particular provisioning domain, the MIF API will send an Address
Announcement message to the subscriber for each such element. If an address becomes invalidated after it has been announced, the MIF API will send a No Address Announcement message.

If a provisioning domain expires or becomes invalid, the MIF API will iterate across the list of remaining configuration elements for that provisioning domain and send a No Address Announcement message for each such address.

3.5.13. Address Announcement

The Address Announcement message includes single IPv4 or IPv6 address and a Provisioning Domain identifier, as well as the valid and preferred lifetimes for that IP address (IPv6 only).

3.5.14. No Address Announcement

The No Address Announcement message indicates that a previously valid address for a provisioning domain is no longer valid. The message includes a provisioning domain identifier and an IPv4 or IPv6 address.

3.5.15. Get Configuration Data

The Get Configuration Data message is sent to the MIF API, and includes a Provisioning Domain ID, a Configuration Element Type, and a Configuration Information Identifier.

Configuration Information Identifiers:

DNS Server List

...TBD...

The MIF API searches the configuration database for the specific type of Configuration Element on the specified Provisioning Domain to see if there is any configuration data of the specified type. If so, the MIF API sends a Configuration Data message to the subscriber; otherwise it sends a No Configuration Data message to the subscriber.

3.5.16. Translate Name

The Translate Name message is sent to the MIF API. It includes a provisioning domain and a name, which is a UTF8 string naming a network node. The message also includes a Translation Identifier, which the subscriber must ensure is unique across all outstanding name service requests.

The MIF API begins a name resolution process. As results come in from the name resolution process, the MIF API sends Name Translation
messages to the subscriber for each such result.

Name resolution can be handled by one or more translations systems such as local host table lookup, Domain Name System, NIS, LLMNR, and is implementation-dependent. **need to think about this

3.5.17. Stop Translating Name

This message is sent to the MIF API to indicate that the subscriber is no longer interested in additional results from a particular name translation process. The message includes the Translation Identifier.

3.5.18. Name Translation

The MIF API sends a Name Translation message to subscribers whenever results come in from a name translation process being performed on behalf of the subscriber. The Name Translation message includes the Translation ID generated by the subscriber, and an IP address returned by the translation process. If a single translation result contains more than one IP address, or IP addresses of different types, the MIF API sends a single Name Translation message for each such IP address.

3.5.19. Connect to Address

The Connect to Address message contains an IP address, a provisioning domain identifier, and a connection identifier which the subscriber must ensure is unique. The MIF API attempts to initiate a TCP connection to the specified IP address using one or more source addresses that are valid for the specified provisioning domain, according to the source address selection policy for that provisioning domain.

If the connection subsequently succeeds, the MIF API will send a Connected message to the subscriber. If it subsequently fails, the MIF API will send a Not Connected message to the subscriber.

3.5.20. Connect to Address From Address

The Connect to Address From Address message contains a source IP address, a destination IP address, a provisioning domain identifier, and a connection identifier which the subscriber must ensure is unique. The MIF API attempts to initiate a TCP connection to the specified IP address using the specified source address.

If the connection subsequently succeeds, the MIF API will send a Connected message to the subscriber. If it subsequently fails, the
MIF API will send a Connection Failed message to the subscriber.

3.5.21. Connected

The Connected message contains the connection identifier that was provided in a previous Connect to Address or Connect to Address From Address message sent by the subscriber. It also contains a token, suitable for use with the connection API, for communicating with the end node to which the connection was established.

3.5.22. Not Connected

The Not Connected message contains the connection identifier that was provided in a previous Connect to Address or Connect to Address From Address message sent by the subscriber. It also contains an indication as to what went wrong with the connection.

4. Example Usage

below is an example that shows how MIF API in use:
As described in the above communication model, the application first invoke the MIF API to query how many interfaces in the host. then, the application invokes MIF API to query how many networks attaches in each interface. application then invoke MIF API to query each DNS
configuration on each interface’s attached network. Application then send DNS query to each DNS server on each network. The DNS servers may return multiple IP address of the queried host name. The application then try to connect to each IP addresses of the host by sending tcp SYN packet to each destination IP addresses through multiple interfaces. Some of the destination IP address may return ACK packet some may not. The application then chose a best connection based on certain criteria. For example, the criteria may based on the quality of the link.

5. Security Considerations

TBD

6. IANA Considerations

None

7. Acknowledgments

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

8. References

8.1. Normative References


8.2. Informative References


Authors’ Addresses

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

Email: liudapeng@chinamobile.com

Ted Lemon
Nominum
Redwood City
CA 94063
USA

Email: Ted.Lemon@nominum.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
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 node’s 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 August 27, 2012.

Copyright Notice
Table of Contents

1. Introduction ........................................... 3
2. Problem overview ...................................... 3
3. Motivation ............................................. 4
  3.1. Use cases .......................................... 4
  3.2. Raised concerns ................................... 9
    3.2.1. Vendor-specific option .......................... 9
    3.2.2. Unicast RA ...................................... 9
    3.2.3. DHCPv6 requires client to use one server ....... 10
    3.2.4. Use VLANs ...................................... 10
4. DHCPv6 Based Solution ................................... 11
  4.1. Default route configuration .......................... 11
  4.2. Configuring on-link routes ............................ 11
  4.3. Deleting obsolete route ............................... 11
  4.4. Applicability to routers ............................. 12
  4.5. Updating Routing Information ......................... 12
  4.6. Limitations ........................................ 13
5. DHCPv6 Route Options .................................. 13
  5.1. Next Hop Option Format ................................ 14
  5.2. Route Prefix Option Format ........................... 15
6. DHCPv6 Server Behavior ................................ 16
7. DHCPv6 Client Behavior ................................ 17
  7.1. Conflict resolution .................................. 18
8. IANA Considerations .................................... 18
9. Security Considerations .................................. 19
10. Contributors and Acknowledgements ........................ 19
11. References ........................................... 20
  11.1. Normative References .............................. 20
  11.2. Informative References ............................. 20
Authors’ Addresses ........................................ 21
1. Introduction

The Neighbor Discovery (ND) 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 provides network administrators with a new set of tools 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 Options 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 gateway. 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 receiving/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. Motivation

The following section enumerates use cases, both in existing networks and as well as in envisaged future deployments. Usage scenarios are specified here in no particular order. As those use cases are described by various network operators, their scenarios may partially overlap.

Discussion: this section is rather long. Nevertheless, there were concerns raised that such option is not needed. Such extensive list can possible solve those concerns. Number of use cases should be limited in future revisions. Alternatively, they can be moved to a separate motivation draft, if needed.

3.1. Use cases

Use case 1: In Broadband network environment where the CPE is multi-homed to two upstream edge routers and each router provides connectivity for different types of services for example internet access and Video on Demand (restricted inside a walled garden) and the Service Provider would like to avoid routing on the CPE, there is a need to provision static route entries on RGs/CPEs. Service Provider requires a centralized control/management point for storing the customer’s related information (IPv6 prefix, IPv6 routes and other provisioned information) and DHCPv6 is a good place for that. Using RA’s would require to manually provision the edge router and this operation is not always possible, for example when router is operated by 3rd party. Broadband Forum document WT-124 issue 3 [BBF-WT-124] calls for this draft to solve the problem.

Use case 2: Operators want (approximate) feature parity so that they
can have (approximate) alignment between their operational procedures for v4 and v6, especially in a dual stack network. Having similar mechanisms for both protocols is desired due to lower operational expenses (OPEX).

Use case 3: In cellular networks, it is efficient for the network to configure routing information in central DHCPv6 server to do unified routing policy information. The gateways (GGSN in cellular network) only need to perform DHCPv6 relay. The Option code sent by clients can be used as an indication that host is MIF capable, so that network need not to do such configuration to host without MIF capabilities.

Use case 4: In cellular network, DHCPv6 is used for IPv6 parameter configuration and RA is used for SLACC of handset. This behavior was introduced in 3GPP Release 8 (or earlier). The network gateway in cellular network (e.g., GGSN) can naturally support DHCPv6 extension since the gateway acts as a DHCPv6 relay. However, it is very hard to update those gateways to use RA announcing the route information. The handsets with MIF feature need to visit subscribed/operator provided service. Some traffic is routed to the operator’s network through 3G interface instead of to Internet through WiFi. DHCPv6 will be used to configure these specific routes. This use case is described in [THREEGPP-23.853].

Use case 5: PMIPv6 use case in LTE network. In LTE cellular network, both GTP and PMIPv6 are used for mobility management. In GTP, it is a point-to-point link between mobile host and PGW (PDN Gateway). However, in PMIPv6 case, the point-to-point link is between mobile host and SGW(Serving Gateway). The PGW sends /64 prefix to SGW through PBA. The SGW sends RA to mobile host. Route option may be needed when the host is multi-homed if it is simultaneously connected to the cellular network and WiFi or it simultaneously connects to multiple APNs in the cellular network. If RA is used for route configuration, both PGW and SGW(whose number is larger than PGW) need to be updated. Moreover, since a host can only connect to one SGW at a time, the SGW have to keep multiple route information received from different PGWs for one host and send them by RA to the host separately. This makes RA is not favorable in this use case.

Use case 6: WiFi networks. Some WiFi hotspots provide local services ("walled garden"). The route configuration on hosts or RGS is needed to direct some traffic to local network, while other traffic to the Internet. While this can be achieved using Route Information Option (RIO) in RA for all nodes that support [RFC4191], it does not allow doing so on a per-host basis.

Use case 7: VPN network. When a user connects to enterprise VPN
network, the routing of VPN traffic need to be configured. Due to
the large number of such VPN networks, we cannot assume all the VPN
network only use RA. DHCPv6 provides another choice which may be
preferred by the VPN network. This situation is described in
[RFC4191], Section 5.2. Hosts that do not support RFC4191 will not
operate properly.

Use case 8: Selective walled garden. Figure 1 illustrates the case
of two clients connected to a shared link. Both clients are assumed
to have global IPv6 addresses and obtain their Internet connectivity
via Router2 by means of a configured or a discovered default route.
Client 1 however, unlike Client 2, is intended to run a specific
application, e.g. VoIP, that is meant to access ServerA by means of
Router1 with Server A being otherwise not reachable from the
Internet. In addition to the global IP address Client1 may be
assigned with another IP address of a more restricted scope for the
purpose of communicating with Server A.

\[---\text{Router1}---\langle\text{IP Cloud}\rangle---\text{ServerA}\]
\[---\text{Client1}---\]
\[---\text{Client2}---\]
\[---\text{Router2}---\langle\text{IP Cloud}\rangle---\text{Internet}\]

Figure 1: Walled garden scenario

The problem in the above scenario comes down to the fact that in
order to reach Server A, Client1 requires to use a more specific
route whose next-hop address is Router1. An ICMPv6 based mechanism
for disseminating more specific route information, as defined in
[RFC4191], disseminates this information via the shared link also to
Client2. Often the operator wants to avoid this redundant
dissemination to passing to Client2. In addition many operators
prefer to be able to manage specific client route information from a
centralized repository instead of managing directly such
configuration on a router, as is required with the ICMPv6 based
scheme. The former requirement is driven by the desire to provide to
each client only the information required for their intended role
which may be tied to a specific service, as well as to allow the
possibility to introduce other routers into the scenario for purposes
of load sharing. The requirement for more centralized configuration
management is often due to administrative boundaries within an
operator’s organization as well as an existing operational practice
that are in place for IPv4, all of which make router based
configuration difficult.
Use case 9: Multihoming problem. A multihomed IPv6 host or gateway needs to solve at least 3 problems to operate properly when more than one link is operational:

1. Source address selection
2. Next-hop selection
3. DNS server selection

Problems one and three are solved by [I-D.ietf-6man-addr-select-opt] and [I-D.ietf-mif-dns-server-selection], respectively. It should be noted that both mechanisms use DHCPv6 as well. This draft attempts to solve problem two. Below is a brief explanation of the problem. See draft [I-D.ietf-v6ops-ipv6-multihoming-without-ipv6nat] for detailed problem analysis, background information and additional discussion regarding the need for a DHCPv6 solution to route information problem and IPv6 multihoming in general (with focus on aforementioned 3 problems).

In multihoming environment, server can restrict assignment of additional prefixes only to hosts that support more advanced next-hop and address selection requirements. (See Section 5.2 of [I-D.ietf-v6ops-ipv6-multihoming-without-ipv6nat]). Obviously this MUST be done on a per-host basis. Information about node capability is obtained via Option Request Option (ORO) in Solicit message, so support for Route Options is also used as means to report node capabilities to a network.

Use case 10: In static networks (i.e. networks that have static routers that are not changing over time, like home network with), such as some enterprise, hosting provider networks or even home network with a single router, it may be possible to stop using RA mechanism and deliver all configuration parameters to hosts using DHCPv6 only. This approach solves the rogue RA problem (i.e. a node that is not an approved router starts announcing RA in a network may hijack traffic from other hosts). This approach may be appealing in some cases, but not in all. For example if there is security association shared between clients and a DHCPv6 server, it may be useful to trust DHCP and disable RA mechanism. Also, environments that need DHCP for extended information, including but not limited to communicating information like DNS servers, hostnames, NTP servers, TFTP boot information and so on are forced to run two protocols increasing complexity and troubleshooting, where we have proof of concept in IPv4 that only one protocol (DHCP) should be needed.

Use case 11: It also has been proposed that route information option may be used as tie breaker in networks that deploy both DHCPv6 route
option and RA. DHCPv6 server could announce routing information along with RA. Legitimate router is also announced over DHCPv6. Host that receives conflicting information over RA may use additional information received from DHCPv6 as a tie breaker. This proposal [nanog-beijnum] was not investigated further.

Use case 12: DHCP-based configuration provides different failure mode than RA. While RA-based configuration works better in networks that offer redundant uplink using separate routers (second router can quickly take over upstream traffic), there are many deployments that cannot use that advantage, because of a single uplink. Current home networks with a single uplink as most obvious example. On the other hand, RA is more severely impacted by rogue entity problem. New rogue RA device may instantly break all other devices on the network. New rogue DHCP server will cause no immediate harm, may cause slow breakage over time, and may in fact never cause any breakage. This is due to the fundamental design choices of each protocol and it is hard to make either work the other way.

Use case 13: DHCP-based configuration may use mostly unicast traffic, while RA-based configuration mostly uses multicast. In some environments implementing multicast traffic may be cumbersome, e.g. in WiMAX environment not every subscriber station (SS) supports multicast channels and multicast capability must be emulated by base station (BS) using redundant transmissions. Classic, stateless, multicasted RA is in disadvantage compared to DHCP with standard unicast option enabled. While it is possible to selectively send unicast RAs to selected subscribers, such architecture is essentially a stateful RA, thus forfeiting major benefit of RA being stateless.

Use case 14: Separated networks. In networks that do not have any routers, two DHCPv6 clients get a global address from DHCPv6 server. They cannot ping each other due to the fact that they do not know prefix that is available on-link. While it is tempting to suggest that separated networks should use link-local addressing, other factors should be taken into consideration. A stateful DHCPv6 may be used as a node monitoring tool, thus having advantage over link-local address usage. The also may be sensor networks that have outside connectivity only sporadically, e.g. uplink is established periodically to gather readings, but most of the time router is powered down for power reasons. Route Option in DHCPv6 could be used to configure on-link routes, while router could announce itself using short-lived RA.

Those requirements and use cases can be summarized as following:
1. In view of the DHCPv6 requirements in several fields, vendor-specific options lead to several segmented definitions. An IETF defined general option is a better choice.

2. Per user/host configuration makes DHCPv6 be used for the on-demand configuration.

3. As there is no well-defined central management system for prefix delegation and routing options via RA, it seems that DHCPv6 is the only available solution. It is better to have a generic option then a bunch of competing vendor options.

4. While this work was initially started with multihoming in mind, it is useful for single interface devices as well.

In a sense this route configuration mechanism makes DHCPv6 complete. Without it, this protocol cannot fully provision all configuration parameters to a host on its own.

3.2. Raised concerns

Opponents of this option proposed several alternative approaches. This section attempts to address raised issues.

3.2.1. Vendor-specific option

Claim: During discussion about route configuration, some opponents say that routing information should be defined as vendor specific option.

Response: There are many ISPs, cellular and BBF network operators, CPE vendors, hardware vendors, DHCP implementors that want to implement and deploy this mechanism. Using vendor-specific option would severly limit interoperability and would make adoption and deployment much more complicated.

This solution is not a technology-specific requirement, it is requested by wide variety of companies, so it is not a vendor specific.

3.2.2. Unicast RA

Claim: Some proponents insist that instead of using DHCPv6 solution, RA should be used instead. Some propose to send unicast RA with RIO option on a per-host basis.

Response: While this approach technically does not violate existing specs, it uses RA in a stateful way, thus the benefit of RA being
stateless is lost. Furthermore, it would require deploying
additional mechanism, like RADIUS to deliver necessary information
about hosts to routers. Authors consider deploying such stateful RA
server with RADIUS support more complicated to deploy than the
solution it tries to avoid (DHCPv6).

As there is no well-defined central management system for prefix
delegation and routing options via RA, it seems that DHCPv6 is the
only available solution. It is better to have a generic option than
a bunch of competing vendor options.

Another concern raised is that RIO is not mandatory nor optional in
3GPP system and there is currently not support in 29.061 RADIUS or
Diameter profile, so use of that alternative is somewhat limited in
some cases.

3.2.3. DHCPv6 requires client to use one server

Claim: DHCPv6 has less rich semantics as client has to pick one out
of all available servers.

Response: While that is how currently most clients are implemented,
there is nothing in [RFC3315] that mandates that. It is true that
DHCPv6 was not designed with several provisioning domains. On the
contrary, section 17.1.3 states that "Upon receipt of one or more
valid Advertise messages, the client selects one or more Advertise
messages based upon the following criteria.". This means that DHCPv6
client can obtain parameters from all available DHCPv6 servers, not
just selected one. As such, DHCPv6 may work with overlapping
provisioning domains. Authors acknowledge that this possibility is
currently rather theoretical, as most known implementations do not
take advantage of that possibility.

3.2.4. Use VLANs

Claim: There was a proposal to use VLANs as a solution to lack of
per-host capability in RA mechanism.

Response: Deploying VLANs complicates network topology much more than
adding a single DHCPv6 option. Furthermore in many cases it is not
possible to deploy VLANs in any reasonable way, e.g. in multihost
environment. Also, low cost devices (e.g. CPE) often do not offer
VLAN capabilities, but they are very much capable of supporting
DHCPv6. Another objection of esthetic nature. Using layer 2
mechanisms to work around limitations in layer 3 is not elegant.
4. 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 Residential Gateway (RG) configuration information from a centralized DHCP server. [I-D.ietf-v6ops-ipv6-multihoming-without-ipv6nat] 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 options 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.

4.1. Default route configuration

Defined mechanism may be used to configure default route. Default route is configured using RT_PREFIX option that specifies ::/0 route, included as suboption in NEXT_HOP.

Server MUST NOT define more than one default route.

4.2. Configuring on-link routes

Server may also configure on-link routes, i.e. routes that are available directly over the link, not via routers. To specify on-link routes, server MAY include RTPREFIX option directly in Advertise and Reply messages.

4.3. Deleting obsolete route

There are two mechanisms that allow removing a route. Each defined route has a route lifetime. If specific route is not refreshed and its timer reaches 0, client MUST remove corresponding entry from routing table.

In cases, where faster route removal is needed, server SHOULD return RT_PREFIX option with route lifetime set to 0. Client that receives RT_PREFIX with route lifetime set to 0 MUST remove specified route immediately, even if its previous lifetime did not expire yet.
4.4. Applicability to routers

Contrary to Router Adverisement mechanism, defined in [RFC4861] that explicitly limits configuration to hosts, routing configuration over DHCPv6 defined in this document may be used by both hosts and routers. (This limitation of RA mechanism was partially lifted by W-1 requirement formulated in [RFC6204].)

One of the envisaged usages for this solution are residential gateways (RG) or Customer Premises Equipment (CPE). Those devices very often perform routing. It may be useful to configure routing on such devices over DHCPv6. One example of such use may be a class of premium users that are allowed to use dedicated router that is not available to regular users.

4.5. Updating Routing Information

Network configuration occasionally changes, due to failure of existing hardware, migration to newer equipment or many other reasons. Therefore there a way to inform clients that routing information have changed is required.

There are several ways to inform clients about new routing information. Every client SHOULD periodically refresh its configuration, according to Information Refresh Time Option, so server may send updated information the next time client refreshes its information. New routes may be configured at that time. As every route has associated lifetime, client is required to remove its routes when this timer expires. This method is particularly useful, when migrating to new router is undergoing, but old router is still available.

Server MAY also announce routes via soon to be removed router with lifetimes set to 0. This will cause the client to remove its routes, despite the fact that previously received lifetime may not yet expire.

Aforementioned methods are useful, when there is no urgent need to update routing information. Bound by timer set by value of Information Refresh Time Option, clients may use outdated routing information until next scheduled renewal. Depending on configured value this delay may be not acceptable in some cases. In such scenarios, administrators are advised to use RECONFIGURE mechanism, defined in [RFC3315]. Server transmits RECONFIGURE message to each client, thus forcing it to immediately start renewal process.

See also Section 4.6 about limitations regarding dynamic routing.
4.6. Limitations

Defined mechanism is not intended to be used as a dynamic routing protocol. It should be noted that proposed mechanism cannot automatically detect routing changes. In networks that use dynamic routing and also employ this mechanism, clients may attempt using routes configured over DHCPv6 even though routers or specific routes ceased to be available. This may cause black hole routing problem. Therefore it is not recommended to use this mechanism in networks that use dynamic routing protocols. This mechanism SHOULD NOT be used in such networks, unless network operator can provide a way to update DHCP server information in case of router availability changes.

Discussion: It should be noted that DHCPv6 server is not able to monitor health of existing routers. As there are currently more than 60 options defined for DHCPv6, it is infeasible to implement mechanism that would monitor huge set of services and stop announcing its availability in case of service outage. Therefore in case of prolonged unavailability human intervention is required to change DHCPv6 server configuration. If that is considered a problem, network administrators should consider using other alternatives, like RA and ND mechanisms (see [RFC4861]).

User is also encouraged to read Section 3.2.

5. DHCPv6 Route Options

A DHCPv6 client interested in obtaining routing information includes the NEXT_HOP and RT PREFIX options as part of its Option Request Option (ORO) in messages directed to a server (as allowed by [RFC3315]), i.e. Solicit, Request, Renew, Rebind or Information-request messages). A Server, when configured to do so, provides the requested route information using zero, one or more NEXT_HOP options in messages sent in response (Advertise, and Reply). So as to allow the route options to be both extensible, as well as conveying detailed info for routes, use is made of a nested options structure. Server sends one or more NEXT_HOP options that specify the IPv6 next hop addresses. Each NEXT_HOP option conveys in turn zero, one or more RT PREFIX options that represents the IPv6 destination prefixes reachable via the given next hop. Server includes RT PREFIX directly in message to indicate that given prefix is available directly on-link. Server MAY send a single NEXT_HOP without any RT PREFIX suboptions or with RT PREFIX that contains ::/0 to indicate available default route. The Formats of the NEXT_HOP and RT PREFIX options are defined in the following sub-sections.
The DHCPv6 Route Options format borrows from the principles of the Route Information Option defined in [RFC4191].

5.1. Next Hop Option Format

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 NEXT_HOP option that contains RT_PREFIX suboptions.

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 zero, one or more prefixes reachable via that next hop.

```
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------------+---------------------------+
|  IPv6 Next Hop Address | NEXT_HOP options          |
| (16 octets)           |                           |
+---------------------+---------------------------+
```

Figure 2: IPv6 Next Hop Option Format

option-code: OPTION_NEXT_HOP (TBD1).

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, zero, one or more RT_PREFIX options that specify prefixes reachable through the given next hop.
5.2. 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. It may also be sent directly in a message to indicate that route is available directly on-link.

```
<table>
<thead>
<tr>
<th>OPTION_RT_PREFIX</th>
<th>option-len</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix-Length</td>
<td>Resvd</td>
</tr>
<tr>
<td>-----------------</td>
<td>----</td>
</tr>
<tr>
<td>Prefix</td>
<td></td>
</tr>
<tr>
<td>(up to 16 octets)</td>
<td></td>
</tr>
<tr>
<td>RT_PREFIX sub-options</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 3: Route Prefix Option Format

option-code: OPTION_RT_PREFIX (TBD2).

option-len: Length of the Route Prefix option including all its sub-options.

Route lifetime 32-bit unsigned integer. Specifies lifetime of the route information, expressed in seconds (relative to the time the packet is sent). There are 2 special values defined. 0 means that route is no longer valid and must be removed by clients. A value of all one bits (0xffffffff) represents infinity.

Prefix Length: 8-bit unsigned integer. The length in bits of the IPv6 Prefix. The value ranges from 0 to 128. This field represents the number of valid leading bits in the prefix.
Resvd:      Reserved field. Server MUST set this value to zero and 
            client MUST ignore its content.

Prf(Route Preference):  2-bit signed integer. The Route Preference 
            indicates whether to prefer the router associated with this 
            prefix over others, when multiple identical prefixes (for 
            different routers) have been received. If the Reserved 
            (10) value is received, the Route Information Option MUST 
            be ignored.

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:    a variable size field that specifies Rule IPv6 prefix. 
            Length of the field is defined by prefix6-len field and is 
            rounded up to the nearest octet boundary (if case when 
            prefix6-len is not divisible by 8). In such case 
            additional padding bits must be zeroed.

RT PREFIX options: Options specific to this particular prefix.

Values for preference field have meaning identical to Route 
Information Option, defined in [RFC4191], Section 2.1:

01 High

00 Medium (default)

11 Low

10 Reserved - MUST NOT be sent

6.  DHCPv6 Server Behavior

When configured to do so, a DHCPv6 server shall provide the Next Hop 
and Route Prefix Options 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.

Server includes NEXT_HOP option with possible RT_HINT suboptions to 
designate that specific routes are available via routers. Server 
includes RT_PROXY options directly in Advertise and Reply messages 
to inform that specific routes are available directly on-link.

If there is more than one route available via specific next hop,
server MUST send only one NEXT_HOP for that next hop, which contains multiple RT_PREFIX options. Server MUST NOT send more than one identical (i.e. with equal next hop address field) NEXT_HOP 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]

7. DHCPv6 Client Behavior

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

When processing a received Route Options 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 SHOULD 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.

Client MUST confirm that routers announced over DHCPv6 are reachable,
using one of methods suitable for specific network type. The most
common mechanism is Neighbor Unreachability Detection (NUD),
specified in [RFC4861]. Client SHOULD use NUD to verify that
received routers are reachable before adjusting its routing tables.
Client MAY use other reachibility verification mechanisms specific to
used network technology. To avoid potential long-lived routing black
holes, client MAY periodically confirm that router is still
reachable.

7.1. Conflict resolution

Information received via Route Options over DHCPv6 MUST be treated
equally to routing information obtained via other sources. In
particular, from the RA perspective, DHCPv6 provisioning should be
treated as if yet another RA was received. Preference field should
be taken into consideration during route information processing. In
particular, administrators are encouraged to read [RFC4191], Section
4.1 for guidance.

To facilitate information merge between DHCPv6 and RA, DHCPv6 option
conveys the same information as RIO, specified in [RFC4191], albeit
on-wire format is slightly different. The differences are:

Metric field (available in previous version of this draft) has been replaced with 2-bit preference field that is in line with RIO
information.

RIO uses 128-length prefix field, while DHCPv6 option uses variable
prefix length. That difference is used to minimize packet size as it
avoid transmitting zeroed octets. Despite slightly different
encoding, delivered information is exactly the same.

If prefix is available directly on-link, Route Prefix option is
conveyed directly in DHCPv6 message, not withing Next Hop option.
That feature is considered a superset, compared to RIO.

8. IANA Considerations

IANA is kindly requested to allocate DHCPv6 option code TBD1 to the
OPTION_NEXT_HOP and TBD2 to OPTION_RT_PREFIX. Both values should be
added to the DHCPv6 option code space defined in Section 24.3 of [RFC3315].

9. 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.

It is essential that clients verify that announced routers are indeed reachable, as specified in Section 7. Failing to do so may create black hole routing problem.

This mechanism may introduce severe problems if deployed in networks that use dynamic routing protocols. See Section 4.6 for details.

DHCPv6 becomes a complete provisioning protocol with this mechanism, i.e. all necessary configuration parameters may be delivered using DHCPv6 only. It was suggested that in some cases this may lead to decision of disabling RA. While RA-less networks could offer lower operational expenses and protection against rogue RAs, they would not work with nodes that do not support this feature. Therefore such decision is not recommended, unless all effects are carefully analyzed. It is worth noting that disabling RA support in hosts would solve rogue RA problem, it would in fact only change the issue into rogue DHCPv6 problem. That is somewhat beneficial, however, as rogue RA may affect all nodes immediately while rogue DHCPv6 server will affect only new nodes, that boot up after rogue server manifests itself.

Reader is also encouraged to read DHCPv6 security considerations document [I-D.ietf-dhc-secure-dhcpv6].

10. Contributors and Acknowledgements

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

The authors would also like to thank Alfred Hines, Ralph Droms, Ted Lemon, Ole Troan, Dave Oran, Dave Ward, Joel Halpern, Marcin Siodelski, Alexandru Petrescu, Roberta Maglione, Tim Chown, Brian Carpenter, Dave Thaler, Lorenzo Colitti and Leo Bicknell for their comments and useful suggestions.

This work has been partially supported by Department of Computer Communications (a division of Gdansk University of Technology) and the Polish Ministry of Science and Higher Education under the European Regional Development Fund, Grant No. POIG.01.01.02-00-045/09-00 (Future Internet Engineering Project).

11. References

11.1. Normative References


11.2. Informative References


[I-D.ietf-mif-dns-server-selection]

[I-D.ietf-mif-dns-server-selection] Troan, O., Miles, D., Matsushima, S., Okimoto, T., and D. Wing, "IPv6 Multihoming without Network Address Translation", draft-ietf-v6ops-ipv6-multihoming-without-ipv6nat-04 (work in progress), February 2012.


Authors’ Addresses

Wojciech Dec  
Cisco Systems  
Haarlerbergweg 13-19  
1101 CH Amsterdam  
The Netherlands  
Email: wdec@cisco.com

Tomasz Mrugalski (editor)  
Internet Systems Consortium, Inc.  
950 Charter Street  
Redwood City, CA  94063  
USA  
Phone: +1 650 423 1345  
Email: tomasz.mrugalski@gmail.com

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:
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.

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 May 3, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as
Table of Contents

1. Introduction ......................................................... 3
2. Requirements and Terminology ................................. 3
3. Problem Background ............................................. 3
4. Solution ............................................................. 4
   4.1. Neighbor Discovery Offload Option ......................... 4
   4.2. Lowering IPv4 Router Preference ......................... 5
   4.3. IPv4 Offloading to Specific Routes ......................... 6
   4.4. IPv4 Offloading to Default Gateway ......................... 7
   4.5. Offload Lifetime ............................................ 7
5. Router Behavior ..................................................... 7
6. Host Behavior ....................................................... 7
7. Security Considerations ........................................ 8
8. IANA Considerations ............................................. 8
9. References .......................................................... 8
   9.1. Normative References ........................................ 8
   9.2. Informative References ...................................... 9
Appendix A. Address Selection Approach .......................... 9
   A.1. Modification to Default Address Selection ................ 9
   A.2. Address selection examples ................................ 10
      A.2.1. Case 1: IPv6-only cellular and IPv4-only WLAN
              accesses .................................................. 10
      A.2.2. Case 2: WLAN access with multiple prefixes ........... 10
      A.2.3. Case 3: WLAN and cellular interface with
              cellular’s IPv4 not default route ...................... 11
      A.2.4. Case 4: Dual-stack cellular access .................... 11
      A.2.5. Case 5: Dual-stack cellular and single stack WLAN .. 11
      A.2.6. Case 6: Coexistence with RFC4191 ...................... 12
Authors’ Addresses ................................................... 12
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.

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 scenario can become a problem, for example, when a smartphone has simultaneously IPv6-enabled cellular connection ([I-D.ietf-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 interface that provides better performance at a lower price 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 the level of router’s willingness to act as a router for IPv4 traffic.

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 can be used together with the Route Information option defined in [RFC4191] to communicate offloading information for specific routes.

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.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length=2    |D|          Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Gateway                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Lifetime              |           Padding             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Padding                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Figure 1: Router Advertisement Offload Option

Type

TBD by IANA.

Length

MUST be set to 2.
D (IPv4 Gateway Preference)

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

Reserved

A 15-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, then this field MUST be set to unspecified address (0.0.0.0) or the Neighbor Discovery Offload option MUST be omitted in all Router Advertisements originated by the router.

Lifetime

16-bit unsigned integer. The Lifetime in seconds limits the validity of state changes caused by this new option. The value of Lifetime in this option SHOULD be smaller than the value of Route Lifetime contained in the Route Information option [RFC4191], if present, in the same Router Advertisement.

Padding

The padding MUST be initialized to zero by the sender and MUST be ignored by the receiver.

The behavior of ‘IPv4 Gateway Preference’ (see Section 4.2) is discussed in more detail in the following sections. The usage of 'Gateway' for offloading is discussed in Section 4.3 and Section 4.4. The Offload option is only used in Router Advertisement messages.

4.2. Lowering IPv4 Router Preference

The 'D' flag bit in the Offload option indicates the willingness of Dual-Stack capable router originating the Router Advertisement to
serve as a gateway for IPv4 traffic. If 'D' is set (1), the router indicates that it SHOULD NOT be used as a gateway for IPv4 traffic, if other gateways are present in the same or other available interfaces. If 'D' is unset (0), the router does not indicate any preference of being or not being a gateway for IPv4 traffic. When 'D' is unset (0), the decision of temporarily modifying the routing status is left for hosts that receive the Offload option (see Section 4.3 and Section 4.4). 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.

4.3. IPv4 Offloading to Specific Routes

To enable offloading of IPv4 traffic to specific routes, both Offload option and Route Information option [RFC4191] MUST present in the same Router Advertisement. A host receiving such Router Advertisement need to maintain a set of status including specific route, Router Preference, and Lifetime. A specific route consists of an IPv4 gateway from the Offload option and an IPv4 prefix from the Route Information option. The Prefix field in the Route Information option SHOULD follow the IPv4-mapped IPv6 address format defined in [RFC4291]. The Prefix Length in the Route Information option is used to indicate the IPv4 prefix length. The Router Preference in the Route Information option indicates whether to prefer the IPv4 router associated with this prefix over others. The Lifetime in the Offload option determines how long the temporarily added specific route will be valid. The Lifetime field in Route Information option SHOULD be ignored.

When 'D' flag is unset (0) in the Offload option, the advertised specific route shall be added by hosts if there is no duplicated prefix matching to the advertised prefix and the advertised lifetime in Offload option is valid. If there is a matching prefix, such specific route will be updated or deleted according to the status of Lifetime and Router Preference. The Lifetime in Offload option determines whether the route will be deleted or updated depending on the existing routing status of the hosts. If the advertised Lifetime is set to 0, any matched prefix and the corresponding route MUST be removed. If Lifetime is valid, the Router Preference further determines whether the gateway of the existing route, if matched, will be substituted to the advertised one, or the lifetime for existing route will be updated.

When 'D' flag is set (1) in the Offload option, any existing specific routes with the next-hop router matching to the advertised gateway SHOULD be removed.
To avoid misconfiguration of offloading operation, only one Offload option is allowed in a single Router Advertisement.

4.4. IPv4 Offloading to Default Gateway

If there is no Route Information options containing IPv4-mapped IPv6 addresses in the same Router Advertisement, the default gateway for offloading can be added, updated, or deleted depending on the ‘D’ flag, Lifetime, and existing routing status on the hosts. When ‘D’ is set (1), the existing default gateway matching to the advertised one SHOULD be removed if there are other usable gateways present in the same or other available interfaces.

When ‘D’ is unset (0) and there is no default gateway present for the receiving interface, the advertised gateway with valid lifetime can be added. If the advertised gateway matches to the existing one on the host, depending on the advertised lifetime, the existing default gateway shall be updated to the advertised lifetime in Offload option or deleted if the lifetime is set 0. If there is a default gateway existing on the receiving interface, which does not match to the advertised gateway, the advertised one SHOULD be ignored.

4.5. Offload Lifetime

The lifetime in the Offload option determines the valid period of temporary routing changes including IPv4 gateway preferences and offloading of IPv4 traffic to specific routes and default gateway. 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). For specific route offloading, the prefix(es) advertised in the Route Information option SHOULD follow IPv4 mapped IPv6 address (e.g. ::ffff:1.2.3.4) as described in 4.3.

6. Host Behavior

A multi-interface capable host SHOULD monitor presence of Offload
option in received Router Advertisement messages. When the Offload option is received, the IPv4 gateway preferences and offloading to default gateway shall temporarily be updated as described in 4.2 and 4.4. Depending on the presence of Route Information in the same Router Advertisement, the offloading to specific IPv4 routes shall temporarily be updated as described in 4.3. Hosts SHOULD uses the lifetime value in the Offload option to determine the valid time of all routing changes caused by the Router Advertisement received.

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 IPv4 gateway preferences and offloading modifications from the previous Router Advertisement. The removals concerns the prefixes configured from router where the router advertisement was received.

7. 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]. The security of utilizing the Offload option is at the equal level to solution in [RFC4191].

8. IANA Considerations

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

9. References

9.1. Normative References

[I-D.ietf-6man-rfc3484-revise]


9.2. Informative References


Appendix A. Address Selection Approach

A.1. 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.

A.2. 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.2.1. 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.2.2. 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::3) == true and Default(D) == true).
A.2.3. 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.2.4. 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.2.5. 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.2.6. 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

Yi Ding (editor)
University of Helsinki
P.O. Box 68
FI-00014 University of Helsinki
Finland

Email: yi.ding@cs.helsinki.fi
Abstract

ISPs wants to take advantage of MIF Transport protocols like SCTP, MPTCP to enhance their End User’s experience when the End User has been offloaded on WLAN. In addition, WLAN are untrusted so ISPs MUST Secure at least some of their End Users’s communications. For various reasons IPsec is the protocol they choose to secure the communications. Currently, IPsec is not adapted to Multiple Interfaces Environment. IPsec can hardly be configured in a proper way which may result in breaking End Users’ communications. At least, it makes it very hard for the End Users to combine Security with MIF enhancements. MOBIKE partly address the problem for a single Interface. This draft provides the problem statement and defines the IPsec Security Requirements for MIF.
Table of Contents

1. Requirements notation .......................... 3
2. Introduction ..................................... 3
3. Problem Statement ................................. 3
   3.1. Adding Interfaces Dynamically ............... 3
   3.2. Removing Interfaces Dynamically ............ 4
   3.3. Multihoming .................................. 5
   3.4. Hard Handover Mobility ....................... 5
   3.5. Soft Handover Mobility ....................... 5
   3.6. Selecting Traffic ............................. 6
   3.7. Conclusion .................................... 6
4. Multiple Interfaces Offload Security Requirements .......... 7
5. Position toward MOBIKE ............................ 9
6. Security Considerations ........................... 10
7. IANA Considerations ................................ 10
8. Acknowledgment ................................... 10
9. Normative References ............................. 10
Authors’ Addresses .................................. 11
1. Requirements notation

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].

2. Introduction

Current Radio Access Network (RAN) infrastructure will not be able to deal with the next future traffic increase. Consequently ISPs are willing to offload the RAN traffic on alternate networks like WLAN. RAN and WLAN have different characteristics, and compared to RAN, WLAN may be untrusted, unreliable and the Network Interface management is performed by the End User (EU). As a consequence, when a EU switches a non-secured communication from RAN to WLAN, it MUST be able to secure it. Then communications on WLAN takes advantage of Multiple Interfaces to enhance the EU experience on WLAN. Thus, such communications MUST have their security appropriately configured to keep the communication secured and avoid that Security breaks the communication.

Section 3 describes the Problem Statement: an IPsec secured communication cannot benefit from MIF features. Then, section 4 provides the IPsec Security Requirements for Multiple Interfaces, Mobility and Multihoming. Section 5 positions MOBIKE [RFC4555] toward the Security Requirements, and provides the additional features MUST be defined for MOBIKE.

3. Problem Statement

3.1. Adding Interfaces Dynamically

The EU may be connected through multiple WLAN Access Points for bandwidth aggregation. Eventually, splitting flows among various Access Points may also be one way to overcome WLAN Access Points unreliability. The EU may be able to add or remove an Interface on a given communication. Protocols like SCTP or MPTCP have especially been designed for that purpose. In fact, SCTP through AS-CONF message is able to dynamically add Interfaces to a given SCTP association.

When the EU is being offloaded, the communication may be secured with IPsec. In this draft we consider two scenarios: (1) One where the communication is encapsulated to a Security Gateway through multiple IPsec tunnels (one per Interface). This scenarios may not require the Server to see the EU with Multiple Interfaces. (2) The other
scenario considers a communication where the EU is connected via Multiple Interfaces directly to the Server. In that case, the communication is secured with IPsec transport mode. The main motivation for using End-to-End security is to limit Security Gateway latencies and limit the security overhead.

When the nodes discovers a new Interface, we expect that IPsec adds this Interface. From the existing IPsec Security Associations related to the communication, IPsec MUST be able to derive for both the EU and the Server the IPsec configuration for the ADDED Interface. More specifically, if the EU is connected to a Security Gateway, the EU MUST configure a new IPsec Tunnel so that the communication can be tunnelled from the new Interface to the Security Gateway. With communication, we mean that the EU may send or receive packets related to the communication. If the EU is directly connected to the Server, the EU MUST configure IPsec so that the communication can be also protected by using the new Interface. Note that IPsec does not define which interface SHOULD be used. IPsec is configured so that other protocols in charge of carrying the traffic may be able to choose one or the other Interface.

Currently IPsec does not provide such mechanisms. This means that any time the EU discovers an Interface, it will have to initiate an IKEv2 negotiation that authenticates the EU and the Server and derives the key material. We want to avoid multiple negotiations for a given communication.

An alternative would be to use MOBIKE Multihoming, which provides the opportunity to the EU to add the new Interface with the ADDITIONAL_IP_ADDRESS Notify Payload. This would make the new Interface being considered as an Alternate Interface. In other words, this Interface could be used only if the EU would become unreachable on the running Interface. This does not provide Multiple Interfaces. A single Interface is used at a time, and this is what MOBIKE has been designed for. Furthermore MOBIKE only considers the Tunnel mode, which would only address the Security Gateway scenario.

### 3.2. Removing Interfaces Dynamically

The EU may use Multiple connections on WLAN, section Section 3.1 explains why the EU may be able to dynamically ADD interfaces to a given communication. Similarly, this section shows that the EU MUST also be able to REMOVE Interfaces from a communication. There may be multiple reasons to REMOVE an Interface. The Interface may not be reachable, the EU may not want to use this Interface anymore... On a security point of view, when an Interface is not used for a secure communication, IPsec MUST explicitly DISCARD all traffic on that Interface.
Currently IKEv2 provides the possibility to DELETE a Security Association. However, this requires a per Security Association Negotiation. With frequent Interface changes, and the Multiple Interfaces of the EU, this negotiations require too many Notify Payload. The EU simply wants to advertise the Server to REMOVE an Interface with a single Notify Payload.

MOBIKE overcomes this management issue by using a single Interface. Consequently there is only one active Interface.

3.3. Multihoming

Multihoming is the ability to provision Interfaces in case the running Interface is not reachable anymore. For a secure communication, the EU wants to provide one or a range of Alternate IP addresses that MUST be used in case the Primary Interface is not reachable. The difference with ADDing an interface to an given communication is that with Multihoming the Alternate MUST be used only if the Primary Interface is not reachable. On an IPsec point of view, it means that IPsec MUST be configured to DISCARD any packets of the communication unless the Primary Interface is not reachable. When the Primary Interface is not reachable, then IPsec MUST be configured to PROTECT or BYPASS the traffic for the given communication.

Currently MOBIKE provides Multihoming. However, MOBIKE does not make possible to assign a list of Alternate Interfaces to a specific communication. The reason is that MOBIKE only considers a single working interface.

3.4. Hard Handover Mobility

Hard Handover Mobility is the ability for a host to update an Interface with another. This generates the packets of the Network to be discarded. In an IPsec point of view, updating the Security Association results in DISCARDing packets sent or received on the new Interface, and accepting (BYPASSing or PROTECTing) packets on the old Interface not anymore used.

IPsec with MOBIKE provides this facility. However, it is only provided for the Tunnel mode.

3.5. Soft Handover Mobility

Soft Handover is the ability to switch from an old Interface to the a new Interface with a state where both old and new Interfaces can send or receive traffic so to avoid loosing the packets in the network. Soft Handover can be done with a combination of ADD and REMOVE
operations described in section Section 3.1 and section Section 3.2

As mentioned in section Section 3.1 and section Section 3.2, they are currently NOT handled by MOBIKE.

3.6. Selecting Traffic

The EU MUST be able to ADD / REMOVE an Interface, to provide Alternate Interface for Multihoming, or perform some Mobility with Soft Handover or Hard Handover. However in the previous sections such operations have been considered as a global policy for the EU. In fact the EU may not have the same policy for all its traffic. Thus such operations MUST be provided for a given traffic. Motivations may be that the EU may keep some corporate traffic inside a corporate network (private IP addresses, confidentiality...) whereas Internet traffic can use any Interface and especially the one providing the highest bandwidth.

MOBIKE does not provide this kind of facility since it considered a single Interface in use.

3.7. Conclusion

This section address common scenario for an EU being offloaded on the a WLAN. The EU may be connected to a Security Gateway or directly connected to the Service. In both cases, the EU MUST be able to:
- ADD an Interface: When the EU has discovered a new Interface, it MUST be able to add this Interface to its current configuration. This means, that IPsec MUST be configured to be able to receive or send traffic on all its interfaces.
- REMOVE an Interface: When the EU notice that one Interface is not active, it MUST be able to remove this Interface to its current configuration. This means that IPsec MUST NOT PROTECT any traffic on this Interface.
- Mobility: The EU MUST be able to perform Hard Handover as well as Soft Handover.
- Multihoming: When one link fails, the EU MUST be able to automatically switch the traffic to an Alternate IP address. This means that IPsec MUST be configured to be able to receive or send traffic on that Interface.
- Traffic Selectors: The EU MUST be able to perform all the above operations globally or for a given traffic. Thus, it MUST be able to indicate which traffic the operation MUST be applied to.
4. Multiple Interfaces Offload Security Requirements

Then follows the Multiple Interfaces Offload Security Requirements. Note they only concern the Security layer. The only purpose of those Requirements is to properly configure the EU Security Layer so that the Security Layer does not stall or affect the EU communication. Since this draft considers IPsec [RFC4301] and IKEv2 [RFC5998], Multiple Interfaces, Multihoming and Mobility address two different channels:

- The DATA channel: i.e. EU communication. In that case, Security Requirements means how to secure properly the IPsec Security Policy Database and Security Association Database, so that IPsec do not block the EU communication. This is like configuring a firewall.
- IKEv2 channel i.e. IKEv2 application. IKEv2 is the IPsec application that configures the IPsec Databases. The application MUST be Multiple Interfaces, Multihoming and Mobility aware so to configure properly the IPsec Databases for the DATA channel.

Here are the following Security Requirements:
- Multiple Interfaces:
  - DATA channel: For the DATA channel, Multiple Interfaces means that the EU MUST be able to ADD or REMOVE an IP address to a given secured communication. Suppose an EU has established a communication with a Server using an Interface I_OLD. When it detects an new Interface I_NEW, the EU MUST be able to configure IPsec Databases so that the communication can go through I_OLD or I_NEW without being discarded. Note that how the DATA traffic is handled and effectively routed on one or the other or both Interfaces is out of scope of the draft. Similarly, when the EU is communicating to the Server with Multiple Interfaces, it MUST be able to configure IPsec Databases so that one or multiple interfaces MUST NOT accept / handle any traffic.
  - IKEv2 channel: For the IKEv2 channel, we suppose using one interface is sufficient. The IKEv2 channel only carries signalization messages. If the EU wants to change the Interface for IKEv2, then it SHOULD perform a Mobility.
- Multihoming:
  - DATA channel: For the DATA channel, Multihoming means that the EU MUST be able to provide Alternate Interfaces to the Server. In the case the Primary (or running) Interface fails, the communication with the Server MUST be able to go on on the Alternate Interface. More specifically, this means that when the Primary Interface is detected as being down, the EU and the Server MUST
configure the IPsec Databases so that the communication can use the Alternate Interface. The difference with ADDing and Interface in the Multiple Interfaces case is that until the Primary Interface is down, the Alternate Interface does not receive or transmit any traffic. Alternate Interfaces DISCARD such traffic.

- IKEv2 channel: For the IKEv2 channel, Multihoming means that when the Primary Interface is down, IKEv2 MUST be able to switch to the Alternate Interface to send IKEv2 signalization messages to the Server. Once IKEv2 has recovered from the Primary Interface crash-down, it can proceed to the DATA channel IPsec configuration.

- Mobility:
  - DATA channel: For the DATA channel, Mobility means that the EU MUST be able to UPDATE the IPsec Databases and change an old Interface (I_OLD) by a new Interface (I_NEW). There are two ways to do so. With a Hard Handover, I_OLD is replaced by I_NEW. Packets that are in the network or in the network stack of the Server and EU when the update occurs will be DISCARDED by the EU. With Soft Handover, the EU ADDs I_NEW and configures its IPsec Databases to receive / send traffic on both I_OLD and I_NEW. Then it REMOVES I_OLD when no traffic is anymore expected on that Interface. Note that Soft Handover is performed according to the Multiple Interfaces Requirements.
  - IKEv2 channel: For the IKEv2 channel, as mentioned in the Multiple Interfaces item, Hard Handover may be sufficient, since the channel only carries signalization messages. Once IKEv2 has moved the IKEv2 channel, it configures IPsec Databases for the DATA channel.

- Traffic Selector:
  - DATA channel: For DATA channel Traffic Selector MUST specify which traffic the Mobility, Multihoming, Multiple Interface action MUST be performed.
  - IKEv2 channel: For the IKEv2, Mobility and Multiple Interface operation may be done with a Hard Handover. However, for Multihoming the channel SHOULD be considered as a specific traffic.

Note that when this draft considers Mobility, Multiple Interfaces or Mobility, only the IPsec configuration is affected. However, in some cases, the configuration of the IPsec Databases may affect the communication of the EU. In fact, if the EU is securing its communication with IPsec and the Tunnel mode, a modification of the outer Interface results in moving the communication. In that case, communication mobility results as a side effect of IPsec Database configuration and this is what is used in MOBIKE [RFC4555]. This case does not happen with the IPsec Transport mode, and the
communication mobility MUST be handled by other protocols then IPsec (application, SHIM6, SCTP, MPTCP...)

5. Position toward MOBIKE

Multihoming Security Requirements are partly handled by IPsec MOBIKE [RFC4555] extension. MOBIKE has been designed for the VPN Mobility and Multihoming use case with a single interface. Thus MOBIKE only addresses the Security Gateway, with the IPsec Tunnel mode. More specifically, MOBIKE does neither address the Transport mode, nor the case of Multiple Interfaces.

Here are the Mobility and Multihoming MOBIKE features:
- MOBIKE Mobility: MOBIKE provides Mobility by UPDATING the outer IP address. Because MOBIKE considers a single interface, the UPDATE occurs for both the IKEv2 channel and the DATA channel. Furthermore, Because MOBIKE only considers the Tunnel mode, UPDATING the IPsec Databases results in moving the communication as a side effect. Because the EU has a single interface, Mobility is always a Hard Handover.
- MOBIKE Multihoming: MOBIKE provides Multihoming mechanism. The two peers are able to exchange Alternate IP addresses. In case the Primary IP address is not reachable, IKEv2 tests the Alternate IP address is still reachable with a COOKIE2 exchange. If the Alternate IP address is still reachable, MOBIKE triggers a MOBILITY and UPDATES the Primary Address by the Alternate IP address. Because the EU has only a single interface, both DATA and IKEv2 channels are updated. Because MOBIKE only considers the Tunnel mode, only communications with Tunnel mode will be updated.

MOBIKE provides Mobility and Multihoming features. However, MOBIKE currently partly addresses the Security Requirements:
- Multiple Interfaces: This is NOT addressed by MOBIKE. This means that currently EU with communications involving Multiple Interfaces will need to establish an IKE channel on each Interface. This also means that there is no Security Interface Management facilities, and for example Soft Handover is NOT possible.
- Mobility: MOBIKE addresses Mobility only for Hard Handover with IPsec Tunnel mode protection. As a result the Security Gateway Scenario is partly addressed. To completely address it with Soft Handover, MOBIKE needs to be extended for Multiple Interfaces. Furthermore, to address End-to-End security with the Server, MOBIKE also needs to be extended for the Transport mode.
- Multihoming: MOBIKE Multihoming features currently address the Security Requirements at least for the IKEv2 channel. For the DATA channel, Multihoming may be extended for Multiple Interface by providing Alternate IP addresses for each Interface.

As a result, MOBIKE requires the following extensions:
- Mobility for Transport: to support all offload architecture, especially those with End-to-End Security.
- Mobility for Soft Handover: to make possible Soft Handover for both Transport and Tunnel mode. Note that Soft Handover is related to Multiple Interfaces Management.
- Multihoming for Multiple Interfaces: Multihoming SHOULD be provided with different Alternate IP addresses depending on the network the connection is currently working. Note that it is also related to Multiple Interface Management.
- Multiple Interfaces Management: MOBIKE MUST consider Multiple Interfaces Management for operations it has been designed for like Mobility and Multihoming. It MUST also provide generic extension to make Multiple Interface Management, such as ADDing or REMOVing an Interface.
- Traffic Selector: the EU MUST be able to explicitly specify which traffic the operation applies.

6. Security Considerations

The whole draft is about security.

7. IANA Considerations

There is no IANA consideration here.

8. Acknowledgment

We would like to thank Daniel Palomares, Pierrick Seite, Brian Carpenter, Hui Deng and Jong-Hyouk Lee for their useful comments.

9. Normative References


Authors’ Addresses

Daniel Migault
Francetelecom - Orange
38 rue du General Leclerc
92794 Issy-les-Moulineaux Cedex 9
France

Phone: +33 1 45 29 60 52
Email: mglt.ietf@gmail.com

Carl Williams
MCSR Labs
Philadelphia, PA 19103
USA

Phone: 650-279-5903
Email: carlw@mcsr-labs.org