

Internet Engineering Task Force  
Internet-Draft  
Intended status: Standards Track  
Expires: January 16, 2014

S. Bhandari  
G. Halwasia  
S. Gundavelli  
Cisco Systems  
H. Deng  
China Mobile  
L. Thiebaut  
Alcatel-Lucent  
J. Korhonen  
Renesas Mobile  
I. Farrer  
Deutsche Telekom AG  
July 15, 2013

DHCPv6 class based prefix  
draft-bhandari-dhc-class-based-prefix-05

## Abstract

This document introduces options to communicate property and associate meta data with prefixes. It extends DHCPv6 prefix delegation and address allocation using the meta data for selection of prefixes and addresses.

## 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 16, 2014.

## Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	3
1.1. Motivation . . . . .	3
1.1.1. Mobile networks . . . . .	4
1.1.2. Home networks . . . . .	4
1.2. Terminology . . . . .	5
1.3. Requirements Language . . . . .	5
2. Overview . . . . .	5
2.1. Prefix Property and Class Options . . . . .	5
2.2. Consideration for different DHCPv6 entities . . . . .	6
2.2.1. Requesting Router Behavior for IA_PD allocation . . . . .	7
2.2.2. Delegating Router Behavior for IA_PD allocation . . . . .	8
2.2.3. DHCPv6 Client Behavior for IA_NA allocation . . . . .	9
2.2.4. DHCPv6 Server Behavior for IA_NA allocation . . . . .	9
2.3. Usage . . . . .	10
2.3.1. Class based prefix and IA_NA allocation . . . . .	10
2.3.2. Class based prefix and IA_PD allocation . . . . .	10
2.3.3. Class based prefix and SLAAC . . . . .	10
2.3.4. Class based prefix and applications . . . . .	11
3. Example Application . . . . .	11
3.1. Mobile gateway example . . . . .	11
3.1.1. Class based prefix delegation . . . . .	13
3.1.2. IPv6 address assignment from class based prefix . . . . .	13
3.2. Homenet Example . . . . .	14
3.2.1. Class based prefix delegation to the HGW . . . . .	15
3.2.2. IPv6 Assignment to Homenet hosts using stateful DHCPv6 . . . . .	16
4. Acknowledgements . . . . .	17
5. Contributors . . . . .	17
6. IANA Considerations . . . . .	17
6.1. OPTION_PREFIX_PROPERTY values . . . . .	17
7. Security Considerations . . . . .	18
8. Change History (to be removed prior to publication as an RFC) . . . . .	18
9. References . . . . .	19
9.1. Normative References . . . . .	19
9.2. Informative References . . . . .	20
Authors' Addresses . . . . .	20

## 1. Introduction

In IPv6 a network interface can acquire multiple addresses from the same scope. In such a multi-prefix network each of the multiple prefixes can have a specific property and purpose associated with it. Example: In a mobile network a mobile device can be assigned a prefix from its home network and another from the visiting network that it is attached to. Another example is a prefix may provide free Internet access without offering any quality of service guarantees while another prefix may be charged along with providing quality of service guarantees for network service access. A prefix can have well defined properties that is universal and have a meta data associated with it that communicates its local significance. The properties and meta data of prefix will be relevant for prefix delegation, source address selection as elaborated in the subsequent sections.

This document defines `OPTION_PREFIX_PROPERTY` option that communicates property of the prefix that is universally understood. This document defines `OPTION_PREFIX_CLASS` option to communicate meta data of the prefix that communicates the prefix's local significance.

This document discusses usage of `OPTION_PREFIX_CLASS` to request and select prefixes with specific meta data via `IA_PD` and `IA_NA` as defined in [RFC3633] and [RFC3315] respectively. This document defines the behavior of the DHCPv6 server, the DHCPv6 prefix requesting router and the DHCPv6 client to use `OPTION_PREFIX_CLASS` option for requesting and selecting prefixes and addresses.

The network address can be configured via DHCPv6 as defined in [RFC3315] or via Stateless Address Autoconfiguration (SLAAC) as defined in [RFC4862], additional information of a prefix can be provided via DHCPv6 or via IPv6 Router Advertisement (RA). The information provided in the options defined in this document `OPTION_PREFIX_PROPERTY` and `OPTION_PREFIX_CLASS` can be used for source address selection. Communicated property and meta data information about the prefix via IPv6 Router Advertisement (RA) will be dealt with in separate document [I-D.korhonen-6man-prefix-properties].

### 1.1. Motivation

In this section motivation for class based prefix delegation that qualifies the delegated prefix with additional class information is described in the context of mobile networks and home networks. The property information attached to a delegated prefix helps to distinguish a delegated IPv6 prefix and selection of the prefix by different applications using it.

#### 1.1.1.1. Mobile networks

In the mobile network architecture, there is a mobile router which functions as a IP network gateway and provides IP connectivity to mobile nodes. Mobile router can be the requesting router requesting delegated IPv6 prefix using DHCPv6. Mobile router can assume the role of DHCPv6 server for mobile nodes(DHCPv6 clients) attached to it. A mobile node in mobile network architecture can be associated with multiple IPv6 prefixes belonging to different domains for e.g. home address prefix, care of address prefix as specified in [RFC3775].

The delegated prefixes when seen from the mobile router perspective appear to be like any other prefix, but each prefixes have different meta data referred to as "Prefix Color" in the mobile networks. Some delegated prefixes may be topologically local and some may be remote prefixes anchored on a global anchor, but available to the local anchor by means of tunnel setup in the network between the local and global anchor. Some may be local with low latency characteristics suitable for voice call break-out, some may have global mobility support. So, the prefixes have different properties and it is required for the application using the prefix to learn about this property in order to use it intelligently. There is currently no semantics in DHCPv6 prefix delegation that can carry this information to specify properties of a delegated prefix. In this scenario, the mobile router is unable to further delegate a longer prefix intelligently based on properties of the prefix learnt. Neither is a mobile device able to learn about the property of the prefix assigned to influence source address selection. Example to determine if the prefix is a home address or care of address.

#### 1.1.1.2. Home networks

In a fixed network environment, the homenet CPE may also function as both a DHCPv6 client (requesting the IA\_PD(s)) and a DHCPv6 server allocating prefixes from delegated prefix(es) to downstream home network hosts. Some service providers may wish to delegate multiple prefixes to the CPE for use by different services classes and traffic types.

Motivations for this include:

- o Using source prefix to identify the service class / traffic type that is being transported. The source prefix may then reliably be used as a parameter for differentiated services or other purposes. E.g. [I-D.jiang-v6ops-semantic-prefix]

- o Using the specific source prefix as a host identifier for other services. E.g. as an input parameter to a DHCPv4 over IPv6 server [I-D.ietf-dhc-dhcpv4-over-ipv6]

To meet these requirements, when the CPE (functioning as a DHCPv6 server) receives an IA\_NA or IA\_TA request from a homenet host, a mechanism is required so that the correct prefix for requested service class can be selected for allocation. Likewise for DHCPv6 clients located in the homenet, a mechanism is necessary so that the intended service class for a requested prefix can be signalled to the DHCPv6 server.

## 1.2. Terminology

This document uses the terminology defined in [RFC2460], [RFC3315] and [RFC3633].

## 1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

## 2. Overview

This section defines prefix property and prefix class options in IA\_PD and IA\_NA. This section defines the behavior of the delegating router, the requesting router and the DHCPv6 client. It discusses these options in the context of a DHCPv6 information request from a DHCPv6 client to a DHCPv6 server.

### 2.1. Prefix Property and Class Options

The format of the DHCPv6 prefix property and prefix class options are shown below.

```

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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      OPTION_PREFIX_PROPERTY      |      option-length(2)      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Properties      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

option-code:      OPTION_PREFIX_PROPERTY (TBD1)
option-length:    2
Properties:       16 bits maintained as
                  OPTION_PREFIX_PROPERTY in
                  IANA registered namespace.
                  Each value in the registry represents a property.
                  Multiple properties can be represented by bitwise
                  ORing of the individual property values in this
                  field.

```

#### Prefix Property Option

The individual property are maintained in OPTION\_PREFIX\_PROPERTY values enumeration explained in Section Section 6.1.

Along with the OPTION\_PREFIX\_PROPERTY a meta data associated with the prefix that is of local relevance is communicated using OPTION\_PREFIX\_CLASS defined below:

```

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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      OPTION_PREFIX_CLASS      |      option-length      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Prefix Class      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

option-code:      OPTION_PREFIX_CLASS (TBD2)
option-length:    2
Prefix Class:     16 bit integer with the integer value
                  of local significance.

```

#### Prefix Class Option

### 2.2. Consideration for different DHCPv6 entities

The model of operation of communicating prefixes to be used by a DHCPv6 server is as follows. A requesting router requests prefix(es) from the delegating router, as described in Section 2.2.1. A delegating router is provided IPv6 prefixes to be delegated to the requesting router. Examples of ways in which the delegating router is provided these prefixes are:

- o Configuration
- o Prefix delegated via a DHCPv6 request to another DHCPv6 server
- o Using a Authentication Authorization Accounting (AAA) protocol like RADIUS [RFC2865]

The delegating router chooses prefix(es) for delegation, and responds with prefix(es) to the requesting router along with additional options in the allocated prefix as described in Section 2.2.2. The requesting router is then responsible for the delegated prefix(es) after the DHCPv6 REQUEST message exchange. For example, the requesting router may create DHCPv6 server configuration pools from the delegated prefix, and function as a DHCPv6 Server. When the requesting router then receives a DHCPv6 IA\_NA requests it can select the address to be allocated based on the OPTION\_PREFIX\_CLASS option received in IA\_NA request or any of the other method as described in Section 2.3.1.

#### 2.2.1. Requesting Router Behavior for IA\_PD allocation

DHCPv6 requesting router can request for prefixes in the following ways:

- o In the SOLICIT message within the IA\_PD Prefix option, it MAY include OPTION\_PREFIX\_CLASS requesting prefix delegation for the specific class indicated in the OPTION\_PREFIX\_CLASS option. It can include multiple IA\_PD Prefix options to indicate it's preference for more than one prefix class. The class of prefix it requests is learnt via configuration or any other out of band mechanism not defined in this document.
- o In the SOLICIT message include an OPTION\_ORO option with the OPTION\_PREFIX\_CLASS option code to request prefixes from all the classes that the DHCPv6 server can provide to this requesting Router.

The requesting router parses the `OPTION_PREFIX_CLASS` option in the `OPTION_IAPREFIX` option area of the corresponding `IA_PD` Prefix option in the `ADVERTISE` message. The Requesting router **MUST** then include all or subset of the received class based prefix(es) in the `REQUEST` message so that it will be responsible for the prefixes selected.

The requesting router parses and stores `OPTION_PREFIX_PROPERTY` if received with the prefix.

#### 2.2.2. Delegating Router Behavior for `IA_PD` allocation

If the Delegating router supports class based prefix allocation by supporting the `OPTION_PREFIX_CLASS` option and it is configured to assign prefixes from different classes, it selects prefixes for class based prefix allocation in the following way:

- o If requesting router includes `OPTION_PREFIX_CLASS` within the `IA_PD` Prefix option, it selects prefixes to be offered from that specific class.
- o If requesting router includes `OPTION_PREFIX_CLASS` within `OPTION_ORO`, then based on its configuration and policy it **MAY** offer prefixes from multiple classes available.

The delegating router responds with an `ADVERTISE` message after populating the `IP_PD` option with prefixes from different classes. Along with including the `IA_PD` prefix options in the `IA_PD` option, it **MAY** include the `OPTION_PREFIX_CLASS` option in the `OPTION_IAPREFIX` option area of the corresponding `IA_PD` prefix option with the class information of the prefix.

If neither the `OPTION_ORO` nor the `IA_PD` option in the `SOLICIT` message include the `OPTION_PREFIX_CLASS` option, then the delegating router **MAY** allocate the prefix as specified in [RFC3633] without including the class option in the `IA_PD` prefix option in the response.

If `OPTION_ORO` option in the `Solicit` message includes the `OPTION_PREFIX_CLASS` option code but the delegating router does not support the solution described in this specification, then the delegating router acts as specified in [RFC3633]. The requesting router **MUST** in this case also fall back to the behavior specified in [RFC3633].



If both delegating and requesting routers support class-based prefix allocation, but the delegating router cannot offer prefixes for any other reason, it MUST respond to requesting router with appropriate status code as specified in [RFC3633]. For e.g., if no prefixes are available in the specified class then the delegating router MUST include the status code NoPrefixAvail in the response message.

In addition if the delegating router has additional property associated with the prefix it will be provided in OPTION\_PREFIX\_PROPERTY option.

#### 2.2.3. DHCPv6 Client Behavior for IA\_NA allocation

DHCPv6 client MAY request for an IA\_NA address allocation from a specific prefix class in the following way:

- o In the SOLICIT message within the IA\_NA option, it MAY include the OPTION\_PREFIX\_CLASS requesting address to be allocated from a specific class indicated in that option. The class information to be requested can be learnt via configuration or any other out of band mechanism not described in this document.

If DHCPv6 client receives OPTION\_PREFIX\_CLASS, OPTION\_PREFIX\_PROPERTY options in the IAaddr-options area of the corresponding IA\_NA but does not support one or both of these options, it SHOULD ignore the received option(s).

#### 2.2.4. DHCPv6 Server Behavior for IA\_NA allocation

The DHCPv6 server parses OPTION\_PREFIX\_CLASS option received and when it supports allocation within the requested OPTION\_PREFIX\_CLASS responds with an ADVERTISE message after populating the IA\_NA option with address information from the requested prefix class. Along with including the IA Address options in the IA\_NA option, it also includes the OPTION\_PREFIX\_CLASS option in the corresponding IAaddr-options area.

Even though the IA\_NA option in the SOLICIT message does not include the OPTION\_PREFIX\_CLASS option, based on local policies, the DHCP server MAY select a default OPTION\_PREFIX\_CLASS value for the client and then SHOULD include the OPTION\_PREFIX\_CLASS option in the IAaddr-options area of the corresponding IA\_NA it sends to the client. If both DHCP client and server support class based address allocation, but the DHCP server cannot offer addresses in the specified Usage class then the DHCP server MUST include the status code NoAddrsAvail (as defined in [RFC3315]) in the response message. If the DHCP server cannot offer addresses for any other reason, it MUST respond to client with appropriate status code as specified in [RFC3315]. In

addition if the server has additional property associated with the prefix by means of configuration or learnt from DHCPv6 prefix delegation or derived via any other means it MUST be sent as OPTION\_PREFIX\_PROPERTY option.

### 2.3. Usage

Class based prefix delegation can be used by the requesting router to configure itself as a DHCPv6 server to serve its DHCPv6 clients. It can allocate longer prefixes from a delegated shorter prefix it received, for serving IA\_NA and IA\_PD requests. Prefix property and class can be used for source address selection by applications using the prefix for communication.

#### 2.3.1. Class based prefix and IA\_NA allocation

The requesting router can use the delegated prefix(es) from different classes (for example "video" (1), "guest"(2), "voice" (3) etc), for assigning the IPv6 addresses to the end hosts through DHCPv6 IA\_NA based on a preconfigured mapping with OPTION\_PREFIX\_CLASS option, the following conditions MAY be observed:

- o It MAY have a pre-configured mapping between the prefix class and OPTION\_USER\_CLASS option received in IA\_NA.
- o It MAY match the OPTION\_PREFIX\_CLASS if the IA\_NA request received contains OPTION\_PREFIX\_CLASS.
- o It MAY have a pre-configured mapping between the prefix class and the client DUID received in DHCPv6 message.
- o It MAY have a pre-configured mapping between the prefix class and its network interface on which the IA\_NA request was received.

The requesting router playing the role of a DHCPv6 server can ADVERTISE IA\_NA from a class of prefix(es) thus selected.

#### 2.3.2. Class based prefix and IA\_PD allocation

If the requesting router, receives prefix(es) for different classes (for example "video"(1), "guest"(2), "voice"(3) etc), it can use these prefix(es) for assigning the longer IPv6 prefixes to requesting routers it serves through DHCPv6 IA\_PD by assuming the role of delegating router, its behavior is explained in Section 2.2.2.

#### 2.3.3. Class based prefix and SLAAC

DHCPv6 IA\_NA and IPv6 Stateless Address Autoconfiguration (SLAAC as defined in [RFC4862]) are two ways by IPv6 addresses can be dynamically assigned to end hosts. Making SLAAC class aware is outside the scope of this document, it is specified in [I-D.korhonen-6man-prefix-properties].

#### 2.3.4. Class based prefix and applications

Applications within a host can do source address selection based on the class of the prefix learnt in OPTION\_PREFIX\_PROPERTY and OPTION\_PREFIX\_CLASS using rules defined in [RFC6724]. The internal data structure and interface for source address selection used by application to choose source prefix with specific property and class in a host is beyond the scope of this document.

### 3. Example Application

#### 3.1. Mobile gateway example

The following sub-sections provide examples of class based prefix delegation and how it is used in a mobile network. Each of the examples will refer to the below network:

The example network consists of :

**Mobile Gateway** It is network entity anchoring IP traffic in the mobile core network. This entity allocates an IP address which is topologically valid in the mobile network and may act as a mobility anchor if handover between mobile and Wi-Fi is supported.

**Mobile Nodes (MN)** A host or router that changes its point of attachment from one network or subnetwork to another. A mobile node may change its location without changing its IP address; it may continue to communicate with other Internet nodes at any location using its (constant) IP address, assuming link-layer connectivity to a point of attachment is available.

**Access Point (AP)** A wireless access point, identified by a MAC address, providing service to the wired network for wireless nodes.

**Access Router (AR)** An IP router residing in an access network and connected to one or more Access Point(AP)s. An AR offers IP connectivity to MNs.

**WLAN controller (WLC)** The entity that provides the centralized forwarding, routing function for the user traffic.

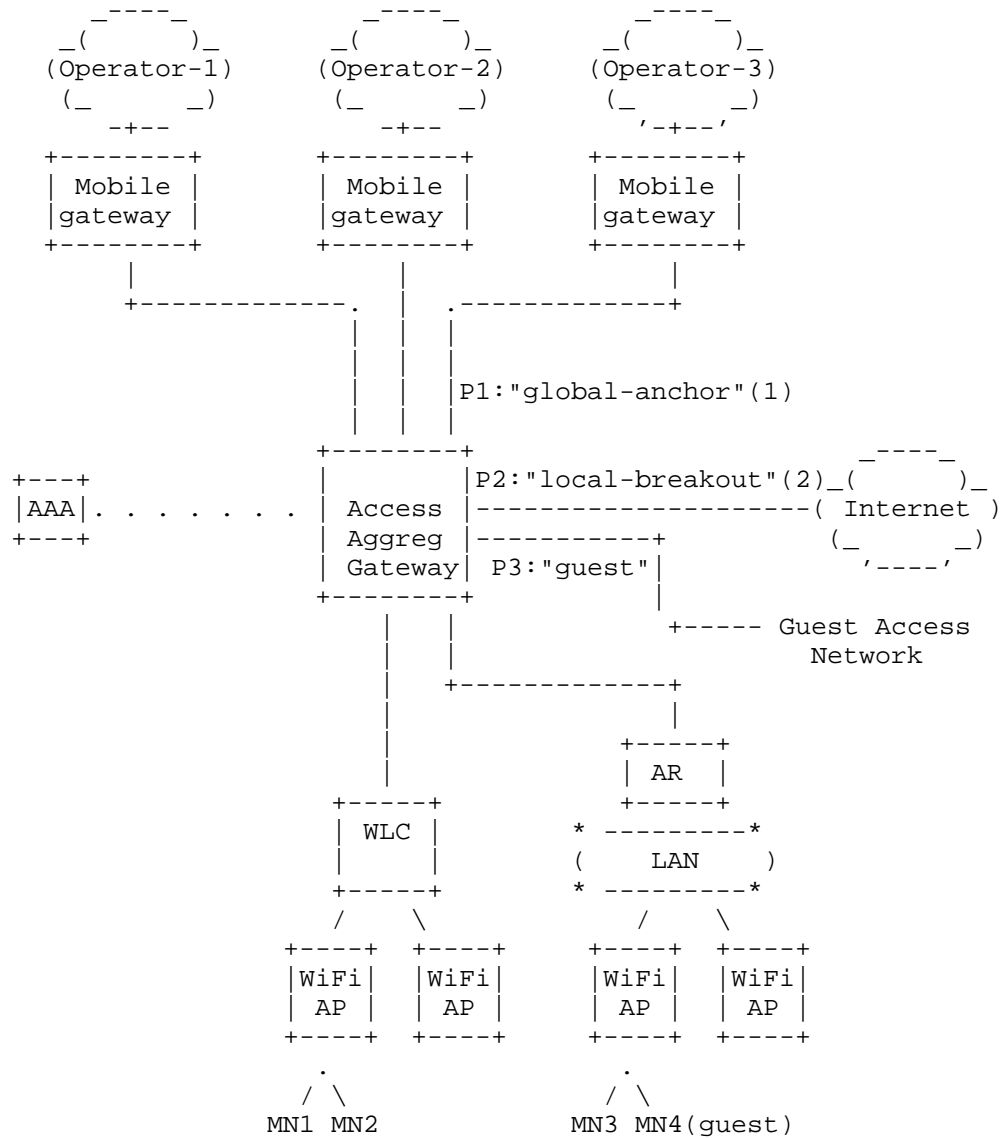


Figure 1: Example mobile network

### 3.1.1. Class based prefix delegation

The Access Aggregation Gateway requests for Prefix delegation from Mobile gateway and associates the prefix received with class "global-anchor"(1). The Access Aggregation Gateway is preconfigured to provide prefixes from the following classes: "global-anchor" (1), "local-breakout"(2), "guest"(3). It has a preconfigured policy to advertise prefixes to requesting routers and mobile nodes based on the service class supported by the service provider for the requesting device. In the example mobile network, the Access Router(AR) requests class based prefix allocation by sending a DHCPv6 SOLICIT message and include OPTION\_PREFIX\_CLASS in the OPTION\_ORO.

The Access Router (AR) receives an advertise with following prefixes in the IA\_PD option:

1. P1: IA\_PD Prefix option with a prefix 3001:1::/64 containing OPTION\_PREFIX\_CLASS set to "global-anchor"(1)
2. P2: IA\_PD Prefix option with a prefix 3001:2::/64 containing OPTION\_PREFIX\_CLASS set to "local-breakout"(2)
3. P3: IA\_PD Prefix option with a prefix 3001:3::/64 containing OPTION\_PREFIX\_CLASS set to "guest"(3)

It sends a REQUEST message with all of above prefixes and receives a REPLY message with prefixes allocated for each of the requested class.

### 3.1.2. IPv6 address assignment from class based prefix

When the Access Router(AR) receives a DHCPv6 SOLICIT requesting IA\_NA from the mobile node that has mobility service enabled, it offers an IPv6 address from the prefix class "global-anchor"(1). For MN3 it advertises 3001:1::1 as the IPv6 address in OPTION\_IAADDR in response to the IA\_NA request.

The Mobile Node(MN4) Figure 1 sends a DHCPv6 SOLICIT message requesting IA\_NA address assignment with OPTION\_USER\_CLASS option containing the value "guest" towards the CPE. The Access Router(AR) assumes the role of the DHCPv6 server and sends an ADVERTISE to the MN with OPTION\_IA\_NA containing an IPv6 address in OPTION\_IAADDR from the "guest"(3) class. The IPv6 address in the OPTION\_IAADDR is set to 3001:3::1. The "guest" class can also be distinguished based on a preconfigured interface or SSID advertised for MNs connecting to it.

When the Access Aggregation Gateway receives a DHCPv6 SOLICIT requesting IA\_NA from MNs through WLC and it has a preconfigured

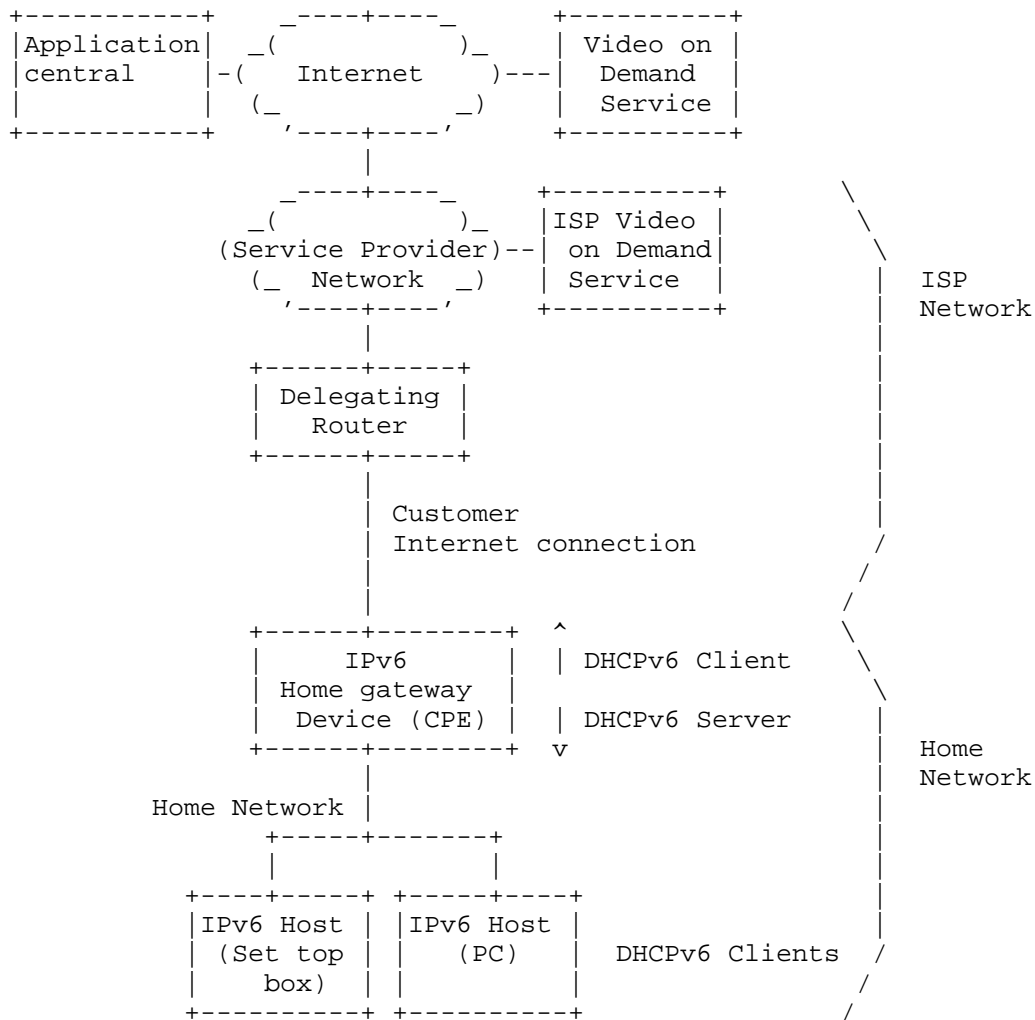
profile to provide both local-breakout Internet access and global-anchor, it offers an IPv6 address from the class "local-breakout" (2) and "global-anchor"(1). For MN1 it advertises 3001:2::1 and 3001:1::2 as the IPv6 address in OPTION\_IAADDR in response to the IA\_NA request. Applications within MN1 can choose to use the appropriate prefix based on the mobility enabled or local-breakout property attached to the prefix based on source address selection policy.

The prefixes that are globally anchored and hence have mobility can be advertised with OPTION\_PREFIX\_PROPERTY set to 0x0002 to convey that the prefix provides network based mobility as listed in Section 6.1. If the prefix also provides security guarantees OPTION\_PREFIX\_PROPERTY can be set to 0x000A to indicate mobility and security guarantees by bitwise ORing of both the properties.

### 3.2. Homenet Example

The following sub-section describes an example of class based prefix delegation in a home network environment. The network consists of the following elements:

- o Home Gateway (HGW) device: a routing device located in the customer's premises that provides connectivity between the customer and the service provider. In this example, the HGW is functioning as both a DHCP client towards the service provider's DHCP infrastructure and a DHCP server towards hosts located in the home network.
- o IPv6 Set Top Box (STB): A dedicated, IPv6 attached, video on demand device.
- o IPv6 PC: An IPv6 attached personal computer
- o Delegating Router: The router in the ISPs network acting as a DHCP server for the IA\_PD request.
- o ISP Video On Demand (ISP-VOD) service: An ISP provided service offering unicast based streaming video content to subscribers.
- o Video On Demand (VOD) service: A server providing unicast based streaming video content to subscribers
- o On demand Video Application: Application hosted on the IPv6 PC
- o Application Central: Application server hosted in the Internet that the On demand Video Application communicates with to access VOD service



Simple home network with Data and Video devices

### 3.2.1. Class based prefix delegation to the HGW

In this example, three different services are being run on the same network. The service provider wishes that traffic is sourced from different prefixes by the home network clients [I-D.jiang-v6ops-semantic-prefix]. The HGW (requesting router) has been configured to request prefix delegation from the ISPs delegating router with the usage classes "video" (1) and "internet"(2) and "video-app" (3) the meaning of these being of relevance to the ISP

operating this and application that are configured out of band to utilize it.

The delegating router is preconfigured to advertise prefixes with these service classes. The HGW sends three IA\_PD options within the SOLICIT message, one with OPTION\_PREFIX\_CLASS "video" (1), the second with "internet" (2) and a third with "video-app" (3). The HGW receives an advertise with the following prefixes in the IA\_PD option:

1. P1: IA\_PD Prefix option with a prefix 3001:5::/56 containing OPTION\_PREFIX\_CLASS set to "video" (1) with OPTION\_PREFIX\_PROPERTY set to 0x0001 indicating there is no internet reach
2. P2: IP\_PD Prefix option with a prefix 3001:6::/56 containing OPTION\_PREFIX\_CLASS set to "internet" (2)
3. P3: IP\_PD Prefix option with a prefix 3001:7::/56 containing OPTION\_PREFIX\_CLASS set to "video-app" (3) with property set to 0x0040 indicating the prefix provides Internet service SLA

It sends a REQUEST message with all of the above prefixes and receives a REPLY message with prefixes allocated for each of the requested classes. The HGW then configures a /64 prefix from each of the delegated prefixes on its LAN interface [RFC6204] and sends out router advertisements (RAs) with the "M" and "O" bits set.

### 3.2.2. IPv6 Assignment to Homenet hosts using stateful DHCPv6

1. STB sends a DHCPv6 SOLICIT message with the OPTION\_PREFIX\_CLASS option set to "video" (1) within the IA\_NA. The HGW checks the requested prefix class against the available prefixes it has been delegated and advertises 3001:5::1 to the STB. The STB then configures this address on its LAN interface and uses it for sourcing requests to the VOD service.
2. The PC sends a DHCPv6 SOLICIT message requesting for IA\_NA with the OPTION\_PREFIX\_CLASS option in ORO indicating support for prefix class. The HGW checks the available prefixes it has been delegated and advertises IA\_NA from P1 (3001:5:2 with property set to 0x0001) , P2 (3001:6::1) and P3 (3001:7::1) to the PC or in absence of OPTION\_PREFIX\_CLASS in the solicit HGW is preconfigured to assign from the "internet"(2) class as the default. The PC then configures these addresses on its LAN interface and uses it for sourcing requests to the Internet.
3. The On demand Video Application on the PC communicates with its well known Application Central using the "internet" prefix and is



directed to use "video-app" prefix if available based on agreement between service provider and on demand video application service provider. The On demand Video Application then connects using the address assigned from P3 that will offer better experience based on the SLA between the providers.

4. If the homenet hosts use SLAAC prefix delegation with the class will use the options and procedure defined in [I-D.korhonen-6man-prefix-properties]

#### 4. Acknowledgements

The authors would like to acknowledge review and guidance received from Frank Brockners, Wojciech Dec, Richard Johnson, Erik Nordmark, Hemant Singh, Mark Townsley, Ole Troan, Bernie Volz, Maciek Konstantynowicz

#### 5. Contributors

Authors would like to thank contributions to use cases and text for various sections received from Sindhura Bandi.

#### 6. IANA Considerations

IANA is requested to assign an option code to OPTION\_PREFIX\_PROPERTY (TBD1) and OPTION\_PREFIX\_CLASS (TBD2) from the "DHCPv6 and DHCPv6 options" registry (<http://www.iana.org/assignments/dhcpv6-parameters/dhcpv6-parameters.xml>).

##### 6.1. OPTION\_PREFIX\_PROPERTY values

IANA is requested to reserve and maintain registry of OPTION\_PREFIX\_PROPERTY values and manage allocation of values as per as per policy defined in [RFC5226] with IETF assigned values requiring IETF consensus, RFC Required policy, any other values other than the ones listed below are not valid.

1. 0x0001 The prefix cannot be used to reach the Internet
2. 0x0002 The prefix provides network based mobility
3. 0x0004 The prefix requires authentication
4. 0x0008 The prefix is assigned on an interface that provides security guarantees
5. 0x0010 Usage is charged

6. 0x0020 The prefix provides multi-homed redundancy
  7. 0x0040 The prefix provides Internet service SLA, based on associated OPTION\_PREFIX\_CLASS
  8. 0x0080 Unassigned
  9. 0x0100 Unassigned
  10. 0x0200 Unassigned
  11. 0x0400 Unassigned
  12. 0x0800 Unassigned
  13. 0x1000 Unassigned
  14. 0x2000 Unassigned
  15. 0x4000 Unassigned
  16. 0x8000 Unassigned
7. Security Considerations
- Security issues related to DHCPv6 which are described in section 23 of [RFC3315] and [RFC3633] apply for scenarios mentioned in this draft as well.
8. Change History (to be removed prior to publication as an RFC)
- Changes from -00 to -01
- a. Modified motivation section to focus on mobile networks
  - b. Modified example with a mobile network and class based prefix delegation in it
- Changes from -01 to -02
- a. Modified option format to be enumerated values
  - b. Added IANA section to request managing of registry for the enumerated values
  - c. Added initial values for the class

- d. Added section for applications to select address with a specific property

Changes from -02 to -03

- a. Added server behaviour for IA\_NA and IA\_PD allocation
- b. Added Class based Information-Request usage

Changes from -03 to -04

- a. Added homenet use case
- b. Split usage class into a fixed IANA maintained properties registry and a prefix class

Changes from -04 to -05

- a. Added on demand video application use case and modified the example section
- b. Added additional properties and reference for SLAAC/ND procedure

## 9. References

### 9.1. Normative References

- [I-D.ietf-dhc-dhcpv4-over-ipv6]  
Cui, Y., Wu, P., Wu, J., and T. Lemon, "DHCPv4 over IPv6 Transport", draft-ietf-dhc-dhcpv4-over-ipv6-06 (work in progress), March 2013.
- [I-D.jiang-v6ops-semantic-prefix]  
Jiang, S., Sun, Q., Farrer, I., and Y. Bo, "A Framework for Semantic IPv6 Prefix", draft-jiang-v6ops-semantic-prefix-03 (work in progress), May 2013.
- [I-D.korhonen-6man-prefix-properties]  
Korhonen, J., Patil, B., Gundavelli, S., Seite, P., and D. Liu, "IPv6 Prefix Properties", draft-korhonen-6man-prefix-properties-02 (work in progress), July 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

- [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.
- [RFC3775] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6204] Singh, H., Beebe, W., Donley, C., Stark, B., and O. Troan, "Basic Requirements for IPv6 Customer Edge Routers", RFC 6204, April 2011.
- [RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.

## 9.2. Informative References

- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

## Authors' Addresses

Shwetha Bhandari  
Cisco Systems  
Cessna Business Park, Sarjapura Marathalli Outer Ring Road  
Bangalore, KARNATAKA 560 087  
India

Email: shwethab@cisco.com

Gaurav Halwasia  
Cisco Systems  
Cessna Business Park, Sarjapura Marathalli Outer Ring Road  
Bangalore, KARNATAKA 560 087  
India

Phone: +91 80 4426 1321  
Email: ghalwasi@cisco.com

Sri Gundavelli  
Cisco Systems  
170 West Tasman Drive  
San Jose, CA 95134  
USA

Email: sgundave@cisco.com

Hui Deng  
China Mobile  
53A, Xibianmennei Ave., Xuanwu District  
Beijing 100053  
China

Email: denghui02@gmail.com

Laurent Thiebaut  
Alcatel-Lucent  
France

Email: laurent.thiebaut@alcatel-lucent.com

Jouni Korhonen  
Renesas Mobile  
Linnoitustie 6  
FIN-02600 Espoo  
Finland

Email: jouni.nospam@gmail.com

Ian Farrer  
Deutsche Telekom AG  
GTN-FM4, Landgrabenweg 151  
Bonn 53227  
Bonn 53227

Email: [ian.farrer@telekom.de](mailto:ian.farrer@telekom.de)

DHC WG  
Internet-Draft  
Updates: 2131 (if approved)  
Intended status: Standards Track  
Expires: December 27, 2013

I. Farrer  
Deutsche Telekom AG  
June 25, 2013

Dynamic Allocation of Shared IPv4 Addresses using DHCPv4 over DHCPv6  
draft-farrer-dhc-shared-address-lease-00

Abstract

This memo describes an update to [RFC2131], which enables the dynamic leasing of shared IPv4 addresses and layer 4 source ports to DHCPv4 over DHCPv6 clients [I-D.ietf-dhc-dhcpv4-over-dhcpv6]. Shared addressing allows a single IPv4 address to be allocated to multiple, active clients simultaneously. Clients sharing the same address are then differentiated by unique L4 source ports.

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 December 27, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Functional Overview . . . . .	3
3. Client-server Interaction . . . . .	4
3.1. Allocating a Shared, Dynamic IPv4 Address . . . . .	4
3.2. Reusing a Previously Allocated Shared, Dynamic IPv4 address . . . . .	5
4. Client Usage of a Shared Address . . . . .	5
5. Additional Changes to [RFC2131] Behaviour . . . . .	6
6. DHCPv4 Port Parameters Option . . . . .	6
7. Specific Behaviour for Softwire Clients . . . . .	7
8. DHCPv4 over DHCPv6 Source Address Option . . . . .	9
9. Security Considerations . . . . .	9
10. IANA Considerations . . . . .	9
11. Acknowledgements . . . . .	10
12. References . . . . .	10
12.1. Normative References . . . . .	10
12.2. Informative References . . . . .	10
Author's Address . . . . .	11

## 1. Introduction

[I-D.ietf-dhc-dhcpv4-over-dhcpv6] introduces a "Unified Server" - a DHCP server capable of servicing both DHCPv6 [RFC3315] and DHCPv4 over DHCPv6 requests. This enables the provisioning of DHCPv4 based configuration to IPv6 connected clients over IPv6 only transport networks.

One of the benefits of the DHCPv4 over DHCPv6 based approach is that it allows the dynamic leasing of IPv4 addresses to clients, based on existing mechanisms for address lease management available in DHCPv4 servers. This can make much more efficient use of remaining public IPv4 addresses than static pre-allocation based approaches as only IPv4 clients which are currently active need to be allocated addresses.



Shortages of available public IPv4 addresses mean that it is not always possible for operators to allocate a full IPv4 address to every active customer simultaneously. This problem may be particularly acute whilst the operator is in the migration phase from a native IPv4 network to a native IPv6 network with IPv4 provided as an overlay service. Such migrations are likely to increase the requirement on public IPv4 addresses so that both existing and transition networks can be provided for.

IPv4 address sharing provides a way of easing this problem. A shared IP address is a single IPv4 address which is allocated to a number of clients simultaneously. The clients differentiate themselves through the use of layer 4 source ports, which are unique for each client sharing a single IPv4 address, known as a Port Set ID (PSID).

The client will generally utilize these restricted source ports by implementing a NAT44 function, translating traffic from the original source address and unrestricted port range to the allocated shared IPv4 address and unique restricted port range.

This technique is also referred to as "extended" or "A+P" addressing [RFC6346].

Due to the nature of address sharing in this manner, it is only suitable for some very specific architectures, such as those described in [I-D.ietf-softwire-map] and [I-D.ietf-softwire-lw4over6]. Use of shared addressing in other, more traditional deployment architectures must be avoided due to the fundamental incompatibilities of assigning a the same /32 IPv4 address to multiple clients such as when they are attached to the same layer 2 segment. Some rules for the use of the allocated shared dynamic address are provided in Section 4 below.

[RFC2131] describes DHCPv4, providing a method for dynamically allocating IPv4 addresses to clients based on three different mechanisms:

- o Automatic Address Allocation
- o Dynamic Address Allocation
- o Manual Address Allocation

This memo describes how the dynamic allocation mechanism can be adapted for allocating shared IPv4 addresses for use by DHCPv4 over DHCPv6 clients and Unified Servers. The approach is referred to as "shared, dynamic" addressing throughout this memo.

## 2. Functional Overview

From a functional perspective, shared, dynamic allocation by the unified server is quite similar to the normal DHCPv4 server dynamic allocation process. The underlying difference is that the unified server MAY allocate the same IPv4 address to more than one DHCPv4 over DHCPv6 client simultaneously, providing that each address allocation also includes a range of layer 4 source ports unique to that address (i.e. each PSID may only be allocated once per /32 address).

To enable this, the DHCPv4 over DHCPv6 client needs to be extended to implement `OPTION_PORTPARAMSV4` (described below). This is used to indicate to the Unified server that it is capable of supporting shared, dynamic addressing and also for conveying the allocated PSID.

The server must be extended to implement `OPTION_PORTPARAMSV4` so that clients able to support shared, dynamic address leasing can be identified and so that shared, dynamic addresses can be allocated and their leases maintained. The server must also manage unique client leases based on the address and PSID tuple, instead of just IPv4 address.

### 3. Client-server Interaction

The following sections describe the changes to the client and server necessary to implement dynamic, shared address allocation.

#### 3.1. Allocating a Shared, Dynamic IPv4 Address

Section 3.1 of [RFC2131] describes the client-server interaction for allocating an IPv4 address. The process described below detail the required changes for the dynamic, shared addressing mechanism.

The following message flow is transported within the DHCPv6 `OPTION_BOOTP_MSG` message.

1. The client constructs its DHCPv4 `DHCPDISCOVER` message (transported within the DHCPv6 `BOOTPRELAYV6` message). The `DHCPDISCOVER` message MUST include the following options:

- \* A client Identifier (constructed as per [RFC4361])
- \* `OPTION_PORTPARAMSV4` (described below)

The client MAY insert a non-zero value in the PSID-Len field within `OPTION_PORTPARAMSV4` to indicate the preferred size of the restricted port range allocation to the unified server.

2. Each Unified server which receives the `DHCPDISCOVER` message containing `OPTION_PORTPARAMSV4` within the `BOOTPRELAYV6` message may respond with a `DHCPOFFER` message which contains an available

IPv4 address in the 'yiaddr' field. The response MUST also include OPTION\_PORTPARAMSV4 containing a restricted port-range. If the received OPTION\_PORTPARAMSV4 field contains a non-zero PSID-Len field, the Unified server MAY allocate a port set of the requested size to the client (depending on policy).

3. The client evaluates all received DHCPPOFFER messages and selects one based on the configuration parameters received (e.g. the size of the offered port set). The client then sends a DHCPREQUEST containing a server identifier and the corresponding OPTION\_PORTPARAMSV4 received in the DHCPPOFFER message.
4. The server identified in the DHCPREQUEST message (via the siaddr field) creates a binding for the client. The binding includes the client identifier, the IPv4 address and the PSID. These parameters are used by both the server and the client to identify the lease referred to in any future DHCP messages. The server responds with a DHCPACK message containing the configuration parameters for the requesting client.
5. The client receives the DHCPACK message with the configuration parameters. The client MUST NOT perform a final check on the address, such as ARPing for a duplicate allocated address.
6. If the client chooses to relinquish its lease by sending a DHCPRELEASE message, the client MUST include the original client identifier, the leased network address and the allocated restricted source ports included in OPTION\_PORTPARAMSV4.

### 3.2. Reusing a Previously Allocated Shared, Dynamic IPv4 address

If the client remembers the previously allocated address and restricted port range, then the process described in section 3.2 of [RFC2131] must be followed. OPTION\_PORTPARAMSV4 MUST be included in the message flow, with the client's requested port set being included in the DHCPDISCOVER message.

## 4. Client Usage of a Shared Address

As a single IPv4 address is being shared between a number of different clients, the allocated shared address is only suitable for certain functions. The client MUST implement a function to ensure that only the allocated layer 4 ports of the shared IPv4 address are used for sourcing new connections.

The client MUST apply the following rules for any traffic to or from the shared /32 IPv4 address:

- o Only port-aware protocols MUST be used.
- o All connections originating from the shared IPv4 address MUST use a source port taken from the allocated restricted port range.

- o The client MUST NOT accept inbound connections with destination ports outside of the allocated restricted port range.

In order to prevent addressing conflicts which could arise from the allocation of the same IPv4 addresses, the client MUST NOT configure the received restricted IPv4 address on-link.

In the event that the DHCPv4 over DHCPv6 configuration mechanism fails for any reason, the client MUST NOT configure an IPv4 link-local address [RFC3927](taken from the 169.254.0.0/16 range).

The mechanism by which a client implements these rules is outside of the scope of this document.

## 5. Additional Changes to [RFC2131] Behaviour

The following change to the behaviour described in [RFC2131] is also necessary in order to implement dynamic shared address allocation.

Section 2.2 The client MUST NOT probe a newly received IPv4 address (e.g. with ARP) to see if it is in use by another host.

## 6. DHCPv4 Port Parameters Option

The Port Parameters Option for DHCPv4 specifies the restricted set of layer 4 source ports that are necessary to dynamically allocate a shared address. The option uses the same fields as the MAP Port Parameters Option described in Section 4.4 of [I-D.ietf-software-map-dhcp], implemented as a DHCPv4 option. This is to maintain compatibility with existing implementations.

The construction and usage of OPTION\_PORTPARAMSV4 is

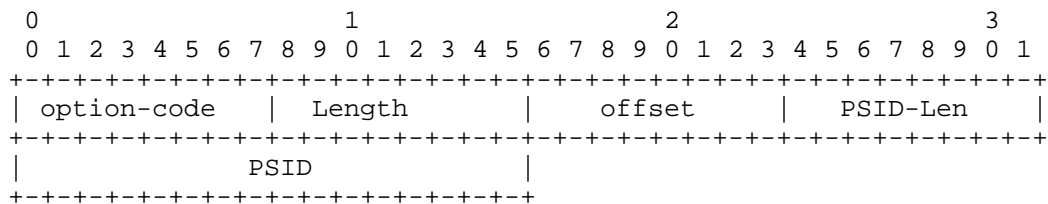


Figure 1: DHCPv4 Port Parameters Option

- o option-code: OPTION\_PORTPARAMSV4 (TBA)
- o option-length: 3
- o offset: (PSID offset) 8 bits long field that specifies the numeric value for the MAP algorithm's excluded port range/offset bits (A-bits), as per section 5.1.1 in [I-D.ietf-software-map].

Allowed values are between 0 and 16, with the default value being 6 for a MAP client. This parameter is unused by a Lightweight 4over6 client and should be set to 0.

- o PSID-len: Bit length value of the number of significant bits in the PSID field. (also known as 'k'). When set to 0, the PSID field is to be ignored. After the first 'a' bits, there are k bits in the port number representing valid of PSID. Subsequently, the address sharing ratio would be  $2^k$ .
- o PSID: Explicit 16-bit (unsigned word) PSID value. The PSID value algorithmically identifies a set of ports assigned to a CE. The first k-bits on the left of this 2-octets field is the PSID value. The remaining (16-k) bits on the right are padding zeros.

[I-D.ietf-softwire-map] (Section 5.1) provides a full description of how the PSID is interpreted by the client.

When receiveing the Port Parameters option with an explicit PSID, the client MUST use apply this PSID to the interface being configured by DHCPv4 over DHCPv6.

## 7. Specific Behaviour for Softwire Clients

[DISCUSSION NOTE: Should the following section be moved into a separate draft as it is more softwire specific?]

Current mechanisms suitable for extending to incorporate dynamic, shared IPv4 addressing include [I-D.ietf-softwire-map] and [I-D.ietf-softwire-lw4over6]. For these mechanisms to function, the operator needs information about the clients allocated IPv4 address, PSID and also the /128 IPv6 prefix which the client will use as the IPv4 in IPv6 tunnel endpoint. This binding information is used by other functional elements in the operator's network (e.g. a softwire tunnel concentrator) for correctly routing traffic to and from clients.

For the shared, dynamic address allocation model, a two-way communication model is necessary so that the Unified Server can indicate to the client the preferred prefix to use for binding the received IPv4 configuration and sourcing tunnel traffic.

As the Unified server is managing the leasing of DHCPv4 to clients, it holds the most accurate IPv4 lease information available in the network. It follows that the unified server should also hold information about the /128 IPv6 prefixes in use by active clients as tunnel endpoints, so that the unified server contains a single comprehensive dynamic IPv4/IPv6 binding table.

To achieve this, the client also needs to inform the Unified server of the /128 prefix that it has bound the IPv4 configuration to.

To provide this function, the DHCPv4oDHCPv6 client MAY implement `OPTION_DHCPV4_O_DHCPV6_SADDR` (defined below). This option is included by the client within `OPTION_BOOTP_MSG` messages and is used alongside the DHCPv4 request process.

The option comprises of two IPv6 prefixes (with associated prefix length fields):

- `cipv6-prefix-hint` Sent by the server to indicate to the client the preferred prefix to bind IPv4 configuration to. If this field contains a prefix, the client MUST perform a longest prefix match between `cipv6-prefix-hint` and all prefixes configured on the device. The selected prefix MUST then be used to bind the received IPv4 configuration to. If this field is left blank, then the client MAY select any valid IPv6 prefix.
- `cipv6-bound-prefix` Used by the client to inform the Unified Server of the IPv6 prefix that it has bound the IPv4 configuration to. This SHOULD be a /128 prefix configured on the client.

If used, this option MUST be present within all future `OPTION_BOOTP_MSG` transactions between the client and the server.

The message flow for this process (aligned to the DHCPv4 address allocation process) is as follows:

DHCPv4 Message	<code>cipv6-prefix-hint</code>	<code>cipv6-hintlen</code>	<code>cipv6-bound-prefix</code>	<code>cipv6-boundlen</code>
DHCPDISCOVER	blank	blank	blank	blank
DHCPOFFER	Preferred prefix	Preferred prefix Length	blank	blank
DHCPREQUEST	Preferred prefix	Preferred prefix length	Bound prefix	Bound prefix Length
DHCPACK	Preferred prefix	Preferred prefix	Bound prefix	Bound prefix

		length		Length	
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

Table 1: IPv6/IPv4 Binding Message Flow

## 8. DHCPv4 over DHCPv6 Source Address Option

The DHCPv4 over DHCPv6 Source address option is used by the Unified server to indicate an IPv6 prefix to use for DHCPv4 over DHCPv6 functions. It is also used by the client to inform the unified server of the prefix it has selected for binding DHCPv4 over DHCPv6 functions to.

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			
+-----+	+-----+	+-----+	+-----+
OPTION_DHCPV4_O_DHCPV6_SADDR	Length		
+-----+	+-----+	+-----+	+-----+
.	cipv6-prefix-hint (variable length)		.
+-----+	+-----+	+-----+	+-----+
cipv6-hintlen	cipv6-bound-prefix (variable length)		.
+-----+	+-----+	+-----+	+-----+
.		cipv6-boundlen	
+-----+	+-----+	+-----+	+-----+

- o option-code: OPTION\_DHCPV4\_O\_DHCPV6\_SADDR (TBA2)
- o option-length: 2 + length of cipv6-prefix-hint + length of cipv6-bound-prefix, specified in bytes
- o cipv6-prefix-hint:
- o cipv6-hintlen: 8 bit field expressing the bit mask length of the IPv6 prefix specified in cipv6-prefix-hint
- o cipv6-bound-prefix: The IPv6 prefix that the client is using to bind the allocated DHCPv4 configuration to
- o cipv6-boundlen: 8 bit field expressing the bit mask length of the IPv6 prefix specified in cipv6-bound-prefix. Default: 128

## 9. Security Considerations

TBD

## 10. IANA Considerations

IANA is kindly requested to allocate the following DHCPv4 option code: TBD for OPTION\_PORTPARAMSV4 and the DHCPv6 option code: OPTION\_DHCPV4\_O\_DHCPV6\_SADDR.

## 11. Acknowledgements

Thanks to Qi Sun and Olaf Bonness for thier reviews.

## 12. References

### 12.1. Normative References

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

### 12.2. Informative References

- [I-D.ietf-dhc-dhcpv4-over-dhcpv6]  
Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4 over DHCPv6 Transport", draft-ietf-dhc-dhcpv4-over-dhcpv6-00 (work in progress), April 2013.
- [I-D.ietf-softwire-lw4over6]  
Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the DS-Lite Architecture", draft-ietf-softwire-lw4over6-00 (work in progress), April 2013.
- [I-D.ietf-softwire-map-dhcp]  
Mrugalski, T., Troan, O., Dec, W., Bao, C., leaf.yeh.sdo@gmail.com, l., and X. Deng, "DHCPv6 Options for Mapping of Address and Port", draft-ietf-softwire-map-dhcp-03 (work in progress), February 2013.
- [I-D.ietf-softwire-map]  
Troan, O., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, "Mapping of Address and Port with Encapsulation (MAP)", draft-ietf-softwire-map-07 (work in progress), May 2013.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, May 2005.



- [RFC4361] Lemon, T. and B. Sommerfeld, "Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4)", RFC 4361, February 2006.
- [RFC6148] Kurapati, P., Desetti, R., and B. Joshi, "DHCPv4 Lease Query by Relay Agent Remote ID", RFC 6148, February 2011.
- [RFC6346] Bush, R., "The Address plus Port (A+P) Approach to the IPv4 Address Shortage", RFC 6346, August 2011.

Author's Address

Ian Farrer  
Deutsche Telekom AG  
Bonn  
Germany

Email: [ian.farrer@telekom.de](mailto:ian.farrer@telekom.de)

DHC Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: December 13, 2014

Q. Sun  
Y. Cui  
Tsinghua University  
M. Siodelski  
ISC  
S. Krishnan  
Ericsson  
I. Farrer  
Deutsche Telekom AG  
June 11, 2014

DHCPv4 over DHCPv6 Transport  
draft-ietf-dhc-dhcpv4-over-dhcpv6-09

Abstract

IPv4 connectivity is still needed as networks migrate towards IPv6. Users require IPv4 configuration even if the uplink to their service provider supports IPv6 only. This document describes a mechanism for obtaining IPv4 configuration information dynamically in IPv6 networks by carrying DHCPv4 messages over DHCPv6 transport. Two new DHCPv6 messages and two new DHCPv6 options are defined for this purpose.

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 December 13, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1. Introduction . . . . .	2
2. Requirements Language . . . . .	3
3. Terminology . . . . .	3
4. Applicability . . . . .	4
5. Architecture Overview . . . . .	4
6. New DHCPv6 Messages . . . . .	5
6.1. Message Types . . . . .	6
6.2. Message Formats . . . . .	6
6.3. DHCPv4-query Message Flags . . . . .	7
6.4. DHCPv4-response Message Flags . . . . .	7
7. New DHCPv6 Options . . . . .	7
7.1. DHCPv4 Message Option Format . . . . .	7
7.2. 4o6 Server Address Option Format . . . . .	8
8. Use of the DHCPv4-query Unicast Flag . . . . .	9
9. DHCP 4o6 Client Behavior . . . . .	10
10. Relay Agent Behavior . . . . .	12
11. DHCP 4o6 Server Behavior . . . . .	12
12. Security Considerations . . . . .	13
13. IANA Considerations . . . . .	14
14. Contributors List . . . . .	14
15. References . . . . .	14
15.1. Normative References . . . . .	14
15.2. Informative References . . . . .	15
Authors' Addresses . . . . .	15

## 1. Introduction

As the migration towards IPv6 continues, IPv6-only networks will become more prevalent. In such networks, IPv4 connectivity will continue to be provided as a service over IPv6-only networks. In addition to provisioning IPv4 addresses for clients of this service, other IPv4 configuration parameters may also be needed (e.g. addresses of IPv4-only services).

This document describes a transport mechanism to carry DHCPv4 messages using the DHCPv6 protocol for the dynamic provisioning of IPv4 addresses and other DHCPv4 specific configuration parameters across IPv6-only networks. It leverages the existing DHCPv4

infrastructure, e.g. failover, DNS updates, DHCP Leasequery, etc.

When IPv6 multicast is used to transport 4o6 messages, another benefit is that the operator can gain information about the underlying IPv6 network the 4o6 client is connected to from the the DHCPv6 relay agents the request has passed through.

## 2. 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 [RFC2119].

## 3. Terminology

This document makes use of the following terms:

CPE:	Customer Premises Equipment (also known as Customer Provided Equipment), which provides access for devices connected to a Local Area Network (typically at the customer's site/home) to the Internet Service Provider's network.
DHCP 4o6 client (or client):	A DHCP client supporting both the DHCPv6 protocol [RFC3315] as well as the DHCPv4 over DHCPv6 protocol described in this document. Such a client is capable of requesting IPv6 configuration using DHCPv6 and IPv4 configuration using DHCPv4 over DHCPv6.
DHCP 4o6 server (or server):	A DHCP server that is capable of processing DHCPv4 packets encapsulated in the DHCPv4 Message option (defined below).
DHCPv4 over DHCPv6:	A protocol described in this document, used to carry DHCPv4 messages in the payload of DHCPv6 messages.

#### 4. Applicability

The mechanism described in this document is not universally applicable. This is intended as a special-purpose mechanism that will be implemented on nodes that must obtain IPv4 configuration information using DHCPv4 in specific environments where native DHCPv4 is not available. Such nodes are expected to follow the advice in the "client behavior" section; nodes that do not require this functionality are expected not to implement it, or not to enable it by default. This mechanism may be enabled using an administrative control, or may be enabled automatically in accordance with the needs of some dual-stack transition mechanism such as [I-D.ietf-softwire-lw4over6]. Such mechanisms are beyond the scope of this document.

#### 5. Architecture Overview

The architecture described here addresses a typical use case, where a DHCP client's uplink supports IPv6 only and the Service Provider's network supports IPv6 and limited IPv4 services. In this scenario, the client can only use the IPv6 network to access IPv4 services, so IPv4 services must be configured using IPv6 as the underlying network protocol.

Although the purpose of this document is to address the problem of communication between the DHCPv4 client and the DHCPv4 server, the mechanism that it describes does not restrict the transported messages types to DHCPv4 only. As the DHCPv4 message is a special type of BOOTP message, BOOTP messages [RFC0951] MAY also be transported using the same mechanism.

DHCP clients may be running on CPE devices, end hosts or any other device that supports the DHCP client function. This document uses the CPE as an example for describing the mechanism. This does not preclude any end-host, or other device requiring IPv4 configuration, from implementing DHCPv4 over DHCPv6 in the future.

This mechanism works by carrying DHCPv4 messages encapsulated within the newly defined DHCPv6 messages. The DHCPv6 relay encapsulation is used solely to deliver DHCPv4 packets to a DHCPv4-capable server, and do not allocate any IPv6 addresses nor provide IPv6 configuration information to the client. Figure 1, below, illustrates one possible deployment architecture of this mechanism.

The DHCP 4o6 client implements a new DHCPv6 message called DHCPv4-query, which contains a new option called the DHCPv4 Message option encapsulating a DHCPv4 message sent by the client. The format of this option is described in Section 7.1.

The DHCPv6 message can be transmitted either via DHCPv6 Relay Agents or directly to the DHCP 4o6 server. The server replies with a DHCPv4-response message, which is a new DHCPv6 message carrying the DHCPv4 response encapsulated in the DHCPv4 Message option.

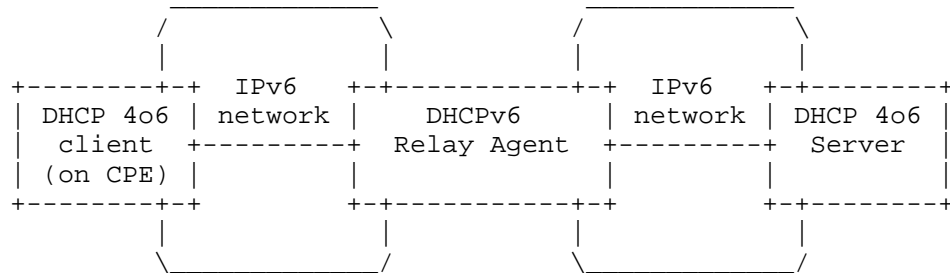


Figure 1: Architecture Overview

Before the client can use DHCPv4 over DHCPv6, it MUST obtain the necessary IPv6 configuration. The client requests the 4o6 Server Address option from the server by sending the option code in Option Request option as described in [RFC3315]. If the server responds with the 4o6 Server Address option, it is an indication to the client to attempt using DHCPv4 over DHCPv6 to obtain IPv4 configuration. Otherwise, the client MUST NOT use DHCPv4 over DHCPv6 to request IPv4 configuration.

The client obtains the address(es) of the DHCP 4o6 server(s) from the 4o6 Server Address option and uses them to communicate with the DHCP 4o6 servers as described in Section 9. If the 4o6 Server Address option contains no addresses (is empty), the client uses the well-known All\_DHCP\_Relay\_Agents\_and\_Servers multicast address to communicate with the DHCP 4o6 server(s).

Before applying for an IPv4 address via a DHCPv4-query message, the client must identify a suitable network interface for the address. Once the request is acknowledged by the server, the client can configure the address and other relevant parameters on this interface. The mechanism for determining a suitable interface is out of the scope of the document.

## 6. New DHCPv6 Messages

Two new DHCPv6 messages carry DHCPv4 messages between the client and the server using the DHCPv6 protocol: DHCPv4-query and DHCPv4-response. This section describes the structures of these messages.

### 6.1. Message Types

- DHCPV4-QUERY (TBD):** The DHCP 4o6 client sends a DHCPv4-query message to a DHCP 4o6 server. The DHCPv4 Message option carried by this message contains a DHCPv4 message that the DHCP 4o6 client uses to request IPv4 configuration parameters from the server.
- DHCPv4-RESPONSE (TBD):** A DHCP 4o6 server sends a DHCPv4-response message to a DHCP 4o6 client. It contains a DHCPv4 Message option carrying a DHCPv4 message received by the server in the DHCPv4 Message option of the DHCPv4-query message.

### 6.2. Message Formats

Both DHCPv6 messages defined in this document share the following format:

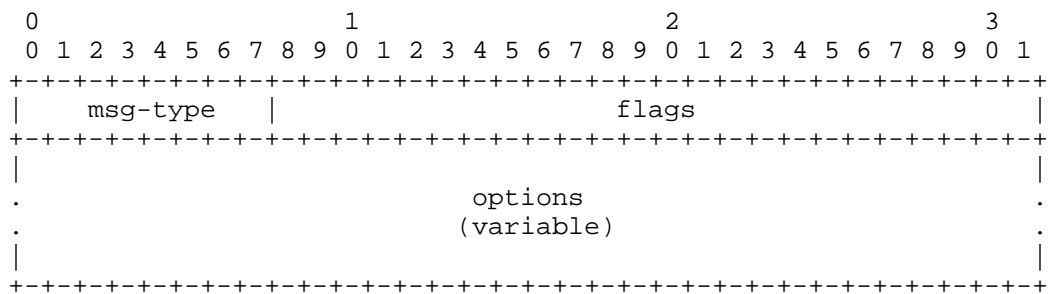


Figure 2: The format of DHCPv4-query and DHCPv4-response messages

- msg-type** Identifies the message type. It can be either DHCPV4-QUERY (TBD) or DHCPV4-RESPONSE (TBD) corresponding to the contained DHCPv4-query or DHCPv4-response, respectively.
- flags** Specifies flags providing additional information required by the server to process the DHCPv4 message encapsulated in the DHCPv4-query message, or required by the client to process a DHCPv4 message encapsulated in the DHCPv4-response message.
- options** Options carried by the message. The DHCPv4 Message Option (described in Section 7.1) MUST be carried by the message. Only DHCPv6 options for IPv4

configuration may be included in this field. It MUST NOT contain DHCPv6 options related solely to IPv6, or IPv6-only service configuration.

### 6.3. DHCPv4-query Message Flags

The "flags" field of the DHCPv4-query is used to carry additional information that may be used by the server to process the encapsulated DHCPv4 message. Currently only one bit of this field is used. Remaining bits are reserved for the future use. The "flags" field has the following format:

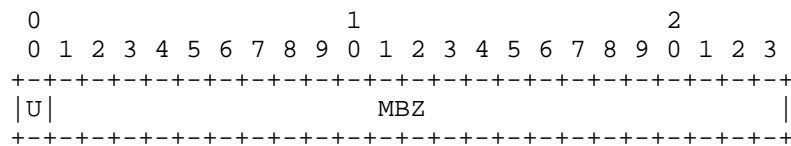


Figure 3: DHCPv4-query flags format

- U                      Unicast Flag. If set to 1, it indicates that the DHCPv4 message encapsulated within the DHCPv4-query message would be sent to a unicast address if it was sent using IPv4. If this flag is set to 0, it indicates that the DHCPv4 message would be sent to the broadcast address if it was sent using IPv4. The usage of the flag is described in detail in Section 8.
- MBZ                    Bits MUST be set to zero when sending and MUST be ignored when receiving.

### 6.4. DHCPv4-response Message Flags

This document introduces no flags to be carried in the "flags" field of the DHCPv4-response message. They are all reserved for the future use. The DHCP 4o6 server MUST set all bits of this field to 0 and the DHCP 4o6 client MUST ignore the content in this field.

## 7. New DHCPv6 Options

### 7.1. DHCPv4 Message Option Format

The DHCPv4 Message option carries a DHCPv4 message that is sent by the client or the server. Such messages exclude any IP or UDP headers.

The format of the DHCPv4 Message option is:



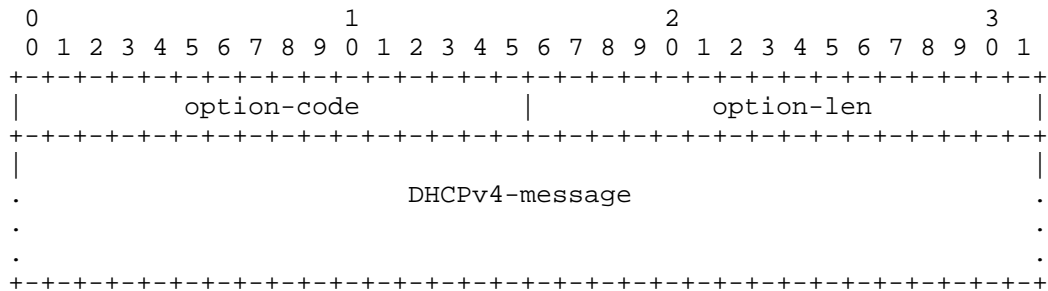


Figure 4: DHCPv4 Message option Format

option-code	OPTION_DHCPV4_MSG (TBD).
option-len	Length of the DHCPv4 message.
DHCPv4-message	The DHCPv4 message sent by the client or the server. In a DHCPv4-query message it contains a DHCPv4 message sent by a client. In a DHCPv4-response message it contains a DHCPv4 message sent by a server in response to a client.

## 7.2. 4o6 Server Address Option Format

The 4o6 Server Address option is sent by a server to a client requesting IPv6 configuration using DHCPv6 [RFC3315]. It carries a list of DHCP 4o6 servers' IPv6 addresses that the client should contact to obtain IPv4 configuration. This list may include multicast and unicast addresses. The client sends its requests to all unique addresses carried in this option.

This option may also carry no IPv6 addresses, which instructs the client to use the All\_DHCP\_Relay\_Agents\_and\_Servers multicast address as the destination address.

The presence of this option in the server's response indicates to the client that it should use DHCPv4 over DHCPv6 to obtain IPv4 configuration. If the option is absent, the client MUST NOT enable DHCPv4-over-DHCPv6 function.

The format of the 4o6 Server Address option is:

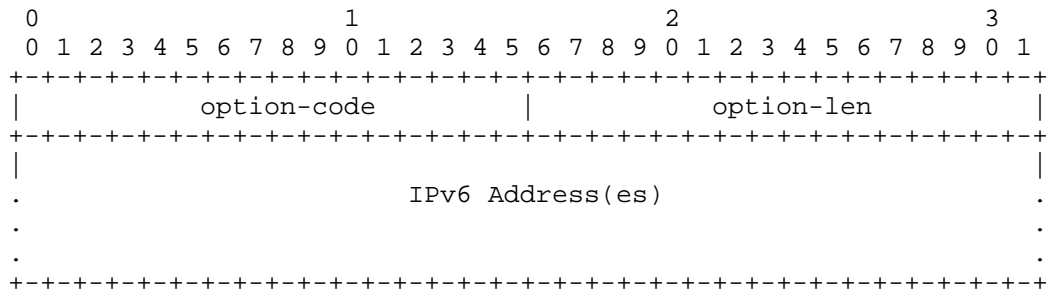


Figure 5: 4o6 Servers Address Option Format

option-code	OPTION_DHCP4_O_DHCP6_SERVER (TBD).
option-len	Length of the IPv6 address(es) carried by the option, i.e. multiple of 16 octets. Minimal length of this option is 0.
IPv6 Address	Zero or more IPv6 addresses of the DHCP 4o6 Server(s).

#### 8. Use of the DHCPv4-query Unicast Flag

A DHCPv4 client conforming to [RFC2131] may send its DHCPREQUEST message to either a broadcast or unicast address depending on its state. For example, a client in the RENEWING state uses a unicast address to contact the DHCPv4 server to renew its lease. A client in the REBINDING state uses a broadcast address.

In DHCPv4 over DHCPv6, IPv6 is used to deliver DHCPv4 messages to the DHCP 4o6 server. There is no relation between the outer IPv6 address and the inner DHCPv4 message. As a result, the server is unable to determine whether the received DHCPv4 messages should have been sent using broadcast or unicast in IPv4 by checking the IPv6 address.

In order to allow the server to determine the client's state, the "Unicast" flag is carried in the DHCPv4-query message. The client MUST set this flag to 1 when the DHCPv4 message would have been sent to the unicast address if using DHCPv4 over IPv4. This flag MUST be set to 0 if the DHCPv4 client would have sent the message to the broadcast address in IPv4. The choice whether a given message should be sent to a broadcast or unicast address is made based on the [RFC2131] and its extensions.

Note: The "Unicast" flag reflects how the DHCPv4 packet would have been sent; not how the DHCPv6 packet itself is sent.

## 9. DHCP 4o6 Client Behavior

The client MUST obtain necessary IPv6 configuration from a DHCPv6 server before using DHCPv4 over DHCPv6. The client requests the 4o6 Server Address option using Option Request option (ORO) in every Solicit, Request, Renew, Rebind and Information-request message. If the DHCPv6 server includes the 4o6 Server Address option in its response, it is an indication that the client can use DHCPv4 over DHCPv6 to obtain the IPv4 configuration (by sending DHCPv4 messages encapsulated in DHCPv4-query messages).

The client MUST NOT use DHCPv4 over DHCPv6 to request IPv4 configuration if the DHCPv6 server does not include the 4o6 Server Address option. If the IPv6 configuration that contained the 4o6 Server Address option subsequently expires, or if the renewed IPv6 configuration does not contain the 4o6 Server Address option, the client MUST stop using DHCPv4 over DHCPv6 to request or renew IPv4 configuration. However, the client continues to request 4o6 Server Address option in the messages sent to the DHCPv6 server as long as it desires to use DHCPv4 over DHCPv6.

It is possible in a multi-homed configuration for there to be more than one DHCPv6 configuration active at the same time that contains a 4o6 Server Address option. In this case, the configurations are treated as being independent, so that when any such configuration is active, a DHCPv4-over-DHCPv6 function may be enabled for that configuration.

An implementation may also treat such configurations as being exclusive, such that only one is kept active at a time. In this case, the client keeps the same configuration active continuously as long as it is valid. If that configuration becomes invalid but one or more other configurations remain valid, the client activates one of the remaining valid configurations.

Which strategy to follow is dependent on the implementation: keeping multiple configurations active at the same time may provide useful redundancy in some applications, but may be needlessly complex in other cases.

If the client receives the 4o6 Server Address option and DHCPv4 [RFC2131] is used on the interface over which the DHCPv6 option was received, the client MUST stop using the IPv4 configuration received using DHCPv4 on this interface. The client MAY send a DHCPRELEASE to the DHCPv4 server to relinquish an existing lease as described in [RFC2131] in section 4.4.6. The client MUST NOT use DHCPv4 on this interface as long as it receives 4o6 Server Address option in the messages received from the DHCPv6 server.

If the client receives a 4o6 Server Address option that contains no IP addresses, i.e. the option is empty, the client MUST send its requests to the All\_DHCP\_Relay\_Agents\_and\_Servers multicast address. If there is a list of IP addresses in the option, the client SHOULD send requests to each unique address carried by the option.

If the client obtained stateless IPv6 configuration by sending Information-request message to the server, the client MUST follow the rules in [RFC4242] to periodically refresh the DHCPv4-over-DHCPv6 configuration (i.e. list of DHCP 4o6 servers) as well as other configuration data. The client which obtained stateful IPv6 configuration will refresh the status of DHCPv4-over-DHCPv6 function when extending a lifetime of acquired IPv6 address (Renew and Rebind messages).

The client MUST employ an IPv6 address of an appropriate scope to source the DHCPv4-query message from. When the client sends a DHCPv4-query message to the multicast address, it MUST use a link-local address as the source address as described in [RFC3315]. When the client sends a DHCPv4-query message using unicast, the source address MUST be an address of appropriate scope, acquired in advance.

The client generates a DHCPv4 message and stores it verbatim in the DHCPv4 Message option carried by the DHCPv4-query message. The client MUST put exactly one DHCPv4 Message option into a single DHCPv4-query message. The client MUST NOT request the 4o6 Server Address option in the DHCPv4-query message.

The client MUST follow rules defined in Section 8 when setting the Unicast flag based on the DHCPv4 destination.

On receiving a DHCPv4-response message, the client MUST look for the DHCPv4 Message option within this message. If this option is not found, the DHCPv4-response message is discarded. If the DHCPv4 Message option is present, the client extracts the DHCPv4 message it contains and processes it as described in section 4.4 of [RFC2131].

When dealing with IPv4 configuration, the client MUST follow the normal DHCPv4 retransmission requirements and strategy as specified in section 4.1 of [RFC2131]. There are no explicit transmission parameters associated with a DHCPv4-query message, as this is governed by the DHCPv4 [RFC2131] "state machine".

The client MUST implement [RFC4361] to ensure that the device correctly identifies itself. It MUST send a 'client identifier' option when using DHCPv4 over DHCPv6.

## 10. Relay Agent Behavior

When a DHCPv6 relay agent receives a DHCPv4-query message, it may not recognize this message. The unknown message MUST be forwarded as described in [I-D.ietf-dhc-dhcpv6-unknown-msgl].

If it recognises the message, the DHCPv6 relay agent MAY allow the configuration of a dedicated DHCPv4 over DHCPv6 specific destination address(es), differing from the address(es) of the DHCPv6-only server(s). To implement this function, the relay checks the received DHCPv6 message type and forwards according to the following logic:

1. If the message type is DHCPV4-QUERY, the packet is relayed to the configured DHCP 4o6 Server's address(es) in the form of normal DHCPv6 packet (i.e. DHCPv6/UDP/IPv6).
2. For any other DHCPv6 message type, forward according to section 20 of [RFC3315].

The above logic only allows for separate relay destinations configured on the relay agent closest to the client (single relay hop). Multiple relaying hops are not considered in the case of separate relay destinations.

## 11. DHCP 4o6 Server Behavior

When the server receives a DHCPv4-query message from a client, it searches for the DHCPv4 Message option. The server discards a packet without this option. In addition, the server MAY notify an administrator about the receipt of this malformed packet. The mechanism for this notification is out of scope for this document.

If the server finds a valid DHCPv4 Message option, it extracts the original DHCPv4 message. Since the DHCPv4 message is encapsulated in the DHCPv6 message, it lacks the information which is typically used by the DHCPv4 server, implementing [RFC2131], to make address allocation decisions, e.g. giaddr for relayed messages and IPv4 address of the interface which the server is using to communicate with directly connected client. Therefore, the DHCP 4o6 server allocates addresses according to the local address assignment policies determined by the server administrator. For example, if the DHCPv4-query message has been sent via a relay, the server MAY use the link-address field of the Relay-forward message as a lookup for the IPv4 subnet to assign DHCPv4 address from. If the DHCPv4-query message has been sent from a directly connected client, the server MAY use IPv6 source address of the message to determine the appropriate IPv4 subnet to use for DHCPv4 address assignment.

Alternatively, the server may act as a DHCPv4 relay agent and forward the DHCPv4 packet to a "normal" DHCPv4 server. The details of such a solution have not been considered by the working group; describing that solution is out of scope of this document and is left as future work should the need for it arise.

The server SHOULD use the "flags" field of the DHCPv4-query message to create a response (server to client DHCPv4 message). The use of this field is described in detail in Section 8.

When an appropriate DHCPv4 response is created, the server places it in the payload of a DHCPv4 Message option, which it puts into the DHCPv4-response message.

If the DHCPv4-query message was received directly by the server, the DHCPv4-response message MUST be unicast from the interface on which the original message was received.

If the DHCPv4-query message was received in a Relay-forward message, the server creates a Relay-reply message with the DHCPv4-response message in the payload of a Relay Message option, and responds as described in section 20.3 of [RFC3315].

## 12. Security Considerations

In this specification, DHCPv4 messages are encapsulated in the newly defined option and messages. This is similar to the handling of the current relay agent messages. In order to bypass firewalls or network authentication gateways, a malicious attacker may leverage this feature to convey other messages using DHCPv6, i.e. use DHCPv6 as a form of encapsulation. However, the potential risk from this is no more severe than that with the current DHCPv4 and DHCPv6 practice.

It is possible for a rogue server to reply with a 4o6 Server Address Option containing duplicated IPv6 addresses, which could cause an amplification attack. To avoid this, the client MUST check if there are duplicate IPv6 addresses in a 4o6 Server Address Option when receiving one. The client MUST ignore any but the first instance of each address.

When considering whether to enable DHCPv4-over-DHCPv6, one important consideration is that when it is enabled, this gives the DHCPv6 server the ability to shut off DHCPv4 traffic, and, consequently, IPv4 traffic, on the interface that is configured to do DHCPv4-over-DHCPv6. For this reason, DHCPv4-over-DHCPv6 should only be enabled in situations where there is a clear trust relationship that eliminates this concern. For instance, a CPE device can safely enable this on its WAN interface, because it is reasonable to assume

that an ISP will not accidentally configure DHCPv4 over DHCPv6 service on that link, and that it will be impractical for an attacker to set up a rogue DHCPv6 server in the ISP's network.

### 13. IANA Considerations

IANA is requested to allocate two DHCPv6 option codes for use by `OPTION_DHCPV4_MSG` and `OPTION_DHCP4_O_DHCP6_SERVER` from the "Option Codes" table, and two DHCPv6 message type codes for the `DHCPV4-QUERY` and `DHCPV4-RESPONSE` from the "Message Types" table of the Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Registry. Both tables can be found at <http://www.iana.org/assignments/dhcpv6-parameters/>.

### 14. Contributors List

Many thanks to Ted Lemon, Bernie Volz, Tomek Mrugalski, Cong Liu and Yuchi Chen, for their great contributions to the specification.

### 15. References

#### 15.1. Normative References

- [I-D.ietf-dhc-dhcpv6-unknown-msg]  
Cui, Y., Sun, Q., and T. Lemon, "Handling Unknown DHCPv6 Messages", draft-ietf-dhc-dhcpv6-unknown-msg-08 (work in progress), March 2014.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC4242] Venaas, S., Chown, T., and B. Volz, "Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 4242, November 2005.
- [RFC4361] Lemon, T. and B. Sommerfeld, "Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4)", RFC 4361, February 2006.

## 15.2. Informative References

- [I-D.ietf-softwire-lw4over6]  
Cui, Y., Qiong, Q., Boucadair, M., Tsou, T., Lee, Y., and  
I. Farrer, "Lightweight 4over6: An Extension to the DS-  
Lite Architecture", draft-ietf-softwire-lw4over6-10 (work  
in progress), June 2014.
- [RFC0951] Croft, B. and J. Gilmore, "Bootstrap Protocol", RFC 951,  
September 1985.

## Authors' Addresses

Qi Sun  
Tsinghua University  
Beijing 100084  
P.R.China

Phone: +86-10-6278-5822  
Email: sunqi@csnet1.cs.tsinghua.edu.cn

Yong Cui  
Tsinghua University  
Beijing 100084  
P.R.China

Phone: +86-10-6260-3059  
Email: yong@csnet1.cs.tsinghua.edu.cn

Marcin Siodelski  
950 Charter Street  
Redwood City, CA 94063  
USA

Phone: +1 650 423 1431  
Email: msiodelski@gmail.com

Suresh Krishnan  
Ericsson

Email: suresh.krishnan@ericsson.com



Ian Farrer  
Deutsche Telekom AG  
GTN-FM4, Landgrabenweg 151  
Bonn, NRW 53227  
Germany

Email: [ian.farrer@telekom.de](mailto:ian.farrer@telekom.de)

Network Working Group  
Internet-Draft  
Updates: 3315,3633 (if approved)  
Intended status: Standards Track  
Expires: September 21, 2015

O. Troan  
B. Volz  
Cisco Systems, Inc.  
M. Siodelski  
ISC  
March 20, 2015

Issues and Recommendations with Multiple Stateful DHCPv6 Options  
draft-ietf-dhc-dhcpv6-stateful-issues-12.txt

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) specification defined two stateful options, IA\_NA and IA\_TA, but did not anticipate the development of additional stateful options. DHCPv6 Prefix Delegation added the IA\_PD option, which is stateful. Applications that use IA\_NA and IA\_PD together have revealed issues that need to be addressed. This document updates RFC 3315 and RFC 3633 to address these issues.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 21, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

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

## Table of Contents

1. Introduction . . . . .	3
2. Conventions . . . . .	3
3. Terminology . . . . .	3
4. Handling of Multiple IA Option Types . . . . .	4
4.1. Placement of Status Codes in an Advertise Message . . . . .	5
4.2. Advertise Message Processing by a Client . . . . .	7
4.3. T1/T2 Timers . . . . .	8
4.4. Renew and Rebind Messages . . . . .	9
4.4.1. Renew Message . . . . .	9
4.4.2. Rebind Message . . . . .	10
4.4.3. Updates to section 18.1.3 of RFC 3315 . . . . .	10
4.4.4. Updates to Section 18.1.4 of RFC 3315 . . . . .	12
4.4.5. Updates to Section 18.1.8 of RFC 3315 . . . . .	13
4.4.6. Updates to Section 18.2.3 of RFC 3315 . . . . .	15
4.4.7. Updates to Section 18.2.4 of RFC 3315 . . . . .	17
4.4.8. Updates to RFC 3633 . . . . .	18
4.5. Confirm Message . . . . .	19
4.6. Decline Should Not Necessarily Trigger a Release . . . . .	20
4.7. Multiple Provisioning Domains . . . . .	21
5. IANA Considerations . . . . .	21
6. Security Considerations . . . . .	21
7. Acknowledgements . . . . .	21
8. References . . . . .	21
8.1. Normative References . . . . .	21
8.2. Informative References . . . . .	22
Authors' Addresses . . . . .	22

## 1. Introduction

DHCPv6 [RFC3315] was written without the expectation that additional stateful DHCPv6 options would be developed. DHCPv6 Prefix Delegation [RFC3633] since added a new stateful option for Prefix Delegation to DHCPv6. Implementation experience of the Customer Edge Router (CER) model described in [RFC7084] has shown issues with the DHCPv6 protocol in supporting multiple stateful option types, in particular IA\_NA (non-temporary addresses) and IA\_PD (delegated prefixes).

This document describes a number of problems encountered with coexistence of the IA\_NA and IA\_PD option types and specifies changes to the DHCPv6 protocol to address these problems.

The intention of this work is to clarify and, where needed, modify the DHCPv6 protocol specification to support IA\_NA and IA\_PD option types within a single DHCPv6 session.

Note that while IA\_TA (temporary addresses) options may be included with other IA option type requests, these generally are not renewed (there are no T1/T2 times) and have a separate life cycle from IA\_NA and IA\_PD option types. Therefore, the IA\_TA option type is mostly out of scope for this document.

The changes described in this document are intended to be incorporated in a new revision of the DHCPv6 protocol specification ([I-D.dhcgw-dhc-rfc3315bis]).

## 2. Conventions

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. Terminology

In addition to the terminology defined in [RFC3315], [RFC3633], and [RFC7227], the following terminology is used in this document:

Identity association (IA):           Throughout this document, "IA" is used to refer to the Identity Association containing addresses or prefixes assigned to a client and carried in the IA\_NA or IA\_PD options respectively.

IA option types:                   This is used to generally mean an IA\_NA and/or IA\_PD option.

Stateful options:	Options that require dynamic binding state per client on the server.
Top-level options:	Top-level options are DHCPv6 options that are not encapsulated within other options, excluding the Relay-Message option. Options encapsulated by Relay-message options, but not by any other option, are still top-level options, whether they appear in a relay agent message or a server message. See [RFC7227].

#### 4. Handling of Multiple IA Option Types

The DHCPv6 specification [RFC3315] was written with the assumption that the only stateful options were for assigning addresses. DHCPv6 Prefix Delegation [RFC3633] describes how to extend the DHCPv6 protocol to handle prefix delegation, but does not clearly specify how the DHCP address assignment and prefix delegation co-exist.

If a client requests multiple IA option types, but the server is configured to only offer a subset of them, the client could react in several ways:

1. Reset the state machine and continue to send Solicit messages,
2. Create separate DHCP sessions for each IA option type and continue to Solicit for the unfulfilled IA options, or
3. The client could continue with the single session, and include the unfulfilled IA options in subsequent messages to the server.

Resetting the state machine and continuing to send Solicit messages may result in the client never completing DHCP and is generally not considered a good solution. It can also result in a packet storm if the client does not appropriately rate limit its sending of Solicit messages or there are many clients on the network. Client implementors that follow this approach, SHOULD implement the updates to RFC-3315 specified in [RFC7083].

Creating a separate DHCP session (separate instances of the client state machine) per IA option type, while conceptually simple, causes a number of issues: additional host resources required to create and maintain multiple instances of the state machine in clients, additional DHCP protocol traffic, unnecessary duplication of other configuration options and the potential for conflict, divergence in

that each IA option type specification specifies its 'own' version of the DHCP protocol.

The single session and state machine allows the client to use the best configuration it is able to obtain from a single DHCP server during the configuration exchange. Note, however, that the server may not be configured to deliver the entire configuration requested by the client. In that case the client could continue to operate only using the configuration received, even if other servers can provide the missing configuration. In practice, especially in the case of handling IA\_NA and IA\_PD, this situation should be rare or a temporary operational error. So, it is more likely for the client to get all configuration if it continues, in each subsequent configuration exchange, to request all the configuration information it is programmed to try to obtain, including any stateful configuration options for which no results were returned in previous exchanges.

One major issue of this last approach is that it is difficult to allow it with the current DHCPv6 specifications; in some cases they are not clear enough, and in other cases existing restrictions can make it impossible. This document introduces some clarifications and small modifications to the current specifications to address these concerns.

While all approaches have their own pros and cons, approach 3 SHOULD be used and is the focus of this document because it is deemed to work best for common cases of the mixed use of IA\_NA and IA\_PD. But this document does not exclude other approaches. Also, in some corner cases it may not be feasible to maintain a single DHCPv6 session for both IA\_NA and IA\_PD. These corner cases are beyond the scope of this document and may depend on the network in which the client (CER) is designed to operate and on the functions the client is required to perform.

The sections which follow update RFC 3315 and RFC 3633 to accommodate the recommendation, though many of the changes are also applicable even if other approaches are used.

#### 4.1. Placement of Status Codes in an Advertise Message

In Reply messages IA specific status codes (i.e., NoAddrsAvail, NotOnLink, NoBinding, NoPrefixAvail) are encapsulated in the IA option. In Advertise messages though, the NoAddrsAvail code is returned at in the top level. This makes sense if the client is only interested in the assignment of the addresses and the failure case is fatal. However, if the client sends both IA\_NA and IA\_PD options in a Solicit message, it is possible that the server offers no addresses

but it offers some prefixes, and the client may choose to send a Request message to obtain the offered prefixes. In this case, it is better if the Status Code option for IA specific status codes is encapsulated in the IA option to indicate that the failure occurred for the specific IA. This also makes the NoAddrsAvail and NoPrefixAvail Status Code option placement for Advertise messages identical to Reply messages.

In addition, how a server formats the Advertise message when addresses are not available has been a point of some confusion and implementations seem to vary (some strictly follow RFC 3315 while others assumed it was encapsulated in the IA option as for Reply messages).

We have chosen the following solution:

Clients MUST handle each of the following Advertise messages formats when there are no addresses available (even when no other IA option types were in the Solicit):

1. Advertise containing the IA\_NAs and/or IA\_TAs with encapsulated Status Code option of NoAddrsAvail and no top-level Status Code option.
2. Advertise containing just a top-level Status Code option of NoAddrsAvail and no IA\_NAs/IA\_TAs.
3. Advertise containing a top-level Status Code option of NoAddrsAvail and IA\_NAs and/or IA\_TAs with a Status Code option of NoAddrsAvail.

Note: Clients MUST handle the last two formats listed above to facilitate backward compatibility with the servers which have not been updated to this specification.

See Section 4.2 for updated text for Section 17.1.3 of RFC 3315 and Section 11.1 of RFC 3633.

Servers MUST return the Status Code option of NoAddrsAvail encapsulated in IA\_NA/IA\_TA options and MUST NOT return a top-level Status Code option of NoAddrsAvail when no addresses will be assigned (1 in the above list). This means that the Advertise response matches the Reply response with respect to the handling of the NoAddrsAvail status.

Replace the following paragraph in RFC 3315, section 17.2.2:

If the server will not assign any addresses to any IAs in a subsequent Request from the client, the server MUST send an Advertise message to the client that includes only a Status Code option with code NoAddrsAvail and a status message for the user, a Server Identifier option with the server's DUID, and a Client Identifier option with the client's DUID.

With:

If the server will not assign any addresses to an IA in a subsequent Request from the client, the server MUST include the IA in the Advertise message with no addresses in the IA and a Status Code option encapsulated in the IA containing status code NoAddrsAvail.

#### 4.2. Advertise Message Processing by a Client

[RFC3315] specifies that a client must ignore an Advertise message if a server will not assign any addresses to a client, and [RFC3633] specifies that a client must ignore an Advertise message if a server returns the NoPrefixAvail status to a requesting router. Thus, a client requesting both IA\_NA and IA\_PD, with a server that only offers either addresses or delegated prefixes, is not supported by the current protocol specifications.

Solution: a client SHOULD accept Advertise messages, even when not all IA option types are being offered. And, in this case, the client SHOULD include the not offered IA option types in its Request. A client SHOULD only ignore an Advertise message when none of the requested IA options include offered addresses or delegated prefixes. Note that ignored messages MUST still be processed for SOL\_MAX\_RT and INF\_MAX\_RT options as specified in [RFC7083].

Replace Section 17.1.3 of RFC 3315: (existing errata)

The client MUST ignore any Advertise message that includes a Status Code option containing the value NoAddrsAvail, with the exception that the client MAY display the associated status message(s) to the user.

With (this includes the changes made by [RFC7083]):



The client MUST ignore any Advertise message that contains no addresses (IAADDR options encapsulated in IA\_NA or IA\_TA options) and no delegated prefixes (IAPREFIX options encapsulated in IA\_PD options, see RFC 3633) with the exception that the client:

- MUST process an included SOL\_MAX\_RT option (RFC 7083) and
- MUST process an included INF\_MAX\_RT option (RFC 7083).

A client can display any associated status message(s) to the user or activity log.

The client ignoring this Advertise message MUST NOT restart the Solicit retransmission timer.

And, replace:

- The client MAY choose a less-preferred server if that server has a better set of advertised parameters, such as the available addresses advertised in IAs.

With:

- The client MAY choose a less-preferred server if that server has a better set of advertised parameters, such as the available set of IAs, as well as the set of other configuration options advertised.

And, replace the last paragraph of Section 11.1 of RFC 3633 with:

The requesting router MUST ignore any Advertise message that contains no addresses (IAADDR options encapsulated in IA\_NA or IA\_TA options) and no delegated prefixes (IAPREFIX options encapsulated in IA\_PD options, see RFC 3633) with the exception that the requesting router:

- MUST process an included SOL\_MAX\_RT option (RFC 7083) and
- MUST process an included INF\_MAX\_RT option (RFC 7083).

A client can display any associated status message(s) to the user or activity log.

The requesting router ignoring this Advertise message MUST NOT restart the Solicit retransmission timer.

#### 4.3. T1/T2 Timers

The T1 and T2 times determine when the client will contact the server to extend lifetimes of information received in an IA. How should a client handle the case where multiple IA options have different T1 and T2 times?

In a multiple IA option type model, the T1/T2 times are protocol timers, that should be independent of the IA options themselves. If we were to redo the DHCP protocol from scratch the T1/T2 times should be carried in a separate DHCP option.

Solution: The server MUST set the T1/T2 times in all IA options in a Reply or Advertise message to the same value. To deal with the case where servers have not yet been updated to do that, the client MUST select a T1 and T2 time from all IA options which will guarantee that the client will send Renew/Rebind messages not later than at the T1/T2 times associated with any of the client's bindings.

As an example, if the client receives a Reply with T1\_NA of 3600 / T2\_NA of 5760 and T1\_PD of 0 / T2\_PD of 1800, the client SHOULD use the T1\_PD of 0 / T2\_PD of 1800. The reason for this is that a T1 of 0 means that the Renew time is at the client's discretion, but this value cannot be greater than the T2 value (1800).

The following paragraph should be added to Sections 18.2.1, 18.2.3, and 18.2.4 of RFC 3315:

The T1/T2 times set in each applicable IA option for a Reply MUST be the same values across all IAs. The server MUST determine the T1/T2 times across all of the applicable client's bindings in the Reply. This facilitates the client being able to renew all of the bindings at the same time.

Note: This additional paragraph has also been included in the revised text later for Sections 18.2.3 and 18.2.4 of RFC 3315.

Changes for client T1/T2 handling are included in Section 4.4.3 and Section 4.4.4.

#### 4.4. Renew and Rebind Messages

This section presents issues with handling multiple IA option types in the context of creation and processing the Renew and Rebind messages. It also introduces relevant updates to the [RFC3315] and [RFC3633].

##### 4.4.1. Renew Message

In multiple IA option type model, the client may include multiple IA options in the Request message, and the server may create bindings only for a subset of the IA options included by the client. For the IA options in the Request message for which the server does not create the bindings, the server sends the IA options in the Reply message with the NoAddrsAvail or NoPrefixAvail status codes.

The client may accept the bindings created by the server, but may desire the other bindings to be created once they become available, e.g. when the server configuration is changed. The client which accepted the bindings created by the server will periodically send a Renew message to extend their lifetimes. However, the Renew message, as described in the [RFC3315], does not support the ability for the client to extend the lifetimes of the bindings for some IAs, while requesting bindings for other IAs.

Solution: The client, which sends a Renew message to extend the lifetimes of the bindings assigned to the client, SHOULD include IA options for these bindings as well as IA options for all other bindings that the client desires but has been unable to obtain. The client and server processing need to be modified. Note that this change makes the server's IA processing of Renew similar to the Request processing.

#### 4.4.2. Rebind Message

According to the Section 4.4.1, the client includes IA options in a Renew message for the bindings it desires but has been unable to obtain by sending a Request message, apart from the IA options for the existing bindings.

At time T2, the client stops sending Renew messages to the server and initiates the Rebind/Reply message exchange with any available server. In this case, it should be possible to continue trying to obtain new bindings using the Rebind message if the client failed to get the response from the server to the Renew message.

Solution: The client SHOULD continue to include the IA options received from the server and it MAY include additional IA options to request creation of the additional bindings.

#### 4.4.3. Updates to section 18.1.3 of RFC 3315

Replace Section 18.1.3 of RFC 3315 with the following text:

To extend the valid and preferred lifetimes for the addresses assigned to an IA, the client sends a Renew message to the server from which the addresses were obtained, which includes an IA option for the IA whose address lifetimes are to be extended. The client includes IA Address options within the IA option for the addresses assigned to the IA. The server determines new lifetimes for these addresses according to the administrative configuration of the server. The server may also add new addresses to the IA. The server can remove addresses from the IA by returning IA Address

options for such addresses with preferred and valid lifetimes set to zero.

The server controls the time at which the client contacts the server to extend the lifetimes on assigned addresses through the T1 and T2 parameters assigned to an IA. However, as the client Renews/Rebinds all IAs from the server at the same time, the client MUST select a T1 and T2 time from all IA options which will guarantee that the client will send Renew/Rebind messages not later than at the T1/T2 times associated with any of the client's bindings.

At time T1, the client initiates a Renew/Reply message exchange to extend the lifetimes on any addresses in the IA.

If T1 or T2 had been set to 0 by the server (for an IA\_NA) or there are no T1 or T2 times (for an IA\_TA) in a previous Reply, the client may send a Renew or Rebind message, respectively, at the client's discretion.

The client sets the "msg-type" field to RENEW. The client generates a transaction ID and inserts this value in the "transaction-id" field.

The client places the identifier of the destination server in a Server Identifier option.

The client MUST include a Client Identifier option to identify itself to the server. The client adds any appropriate options, including one or more IA options.

For IAs to which addresses have been assigned, the client includes a corresponding IA option containing an IA Address option for each address assigned to the IA. The client MUST NOT include addresses in any IA option that the client did not obtain from the server or that are no longer valid (that have a zero valid lifetime).

The client MAY include an IA option for each binding it desires but has been unable to obtain. This IA option MUST NOT contain any addresses. However, it MAY contain the IA Address option with IPv6 address field set to 0 to indicate the client's preference for the preferred and valid lifetimes for any newly assigned addresses.

The client MUST include an Option Request option (see section 22.7) to indicate the options the client is interested in receiving. The client MAY include options with data values as hints to the server about parameter values the client would like to have returned.

The client transmits the message according to section 14, using the following parameters:

IRT	REN_TIMEOUT
MRT	REN_MAX_RT
MRC	0
MRD	Remaining time until T2

The message exchange is terminated when time T2 is reached (see section 18.1.4), at which time the client begins a Rebind message exchange.

#### 4.4.4. Updates to Section 18.1.4 of RFC 3315

Replace Section 18.1.4 of RFC 3315 with the following text:

At time T2 (which will only be reached if the server to which the Renew message was sent at time T1 has not responded), the client initiates a Rebind/Reply message exchange with any available server.

The client constructs the Rebind message as described in 18.1.3 with the following differences:

- The client sets the "msg-type" field to REBIND.
- The client does not include the Server Identifier option in the Rebind message.

The client transmits the message according to section 14, using the following parameters:

IRT	REB_TIMEOUT
MRT	REB_MAX_RT
MRC	0
MRD	Remaining time until valid lifetimes of all addresses in all IAs have expired

If all addresses for an IA have expired the client may choose to include this IA without any addresses (or with only a hint for lifetimes) in subsequent Rebind messages to indicate that the client is interested in assignment of the addresses to this IA.

The message exchange is terminated when the valid lifetimes of all addresses across all IAs have expired, at which time the client uses Solicit message to locate a new DHCP server and sends a Request for the expired IAs to the new server.

#### 4.4.5. Updates to Section 18.1.8 of RFC 3315

Replace Section 18.1.8 of RFC 3315 with the following text:

Upon the receipt of a valid Reply message in response to a Solicit (with a Rapid Commit option), Request, Confirm, Renew, Rebind or Information-request message, the client extracts the configuration information contained in the Reply. The client MAY choose to report any status code or message from the status code option in the Reply message.

If the client receives a Reply message with a Status Code containing UnspecFail, the server is indicating that it was unable to process the message due to an unspecified failure condition. If the client retransmits the original message to the same server to retry the desired operation, the client MUST limit the rate at which it retransmits the message and limit the duration of the time during which it retransmits the message.

When the client receives a Reply message with a Status Code option with the value UseMulticast, the client records the receipt of the message and sends subsequent messages to the server through the interface on which the message was received using multicast. The client resends the original message using multicast.

When the client receives a NotOnLink status from the server in response to a Confirm message, the client performs DHCP server solicitation, as described in section 17, and client-initiated configuration as described in section 18. If the client receives any Reply messages that do not indicate a NotOnLink status, the client can use the addresses in the IA and ignore any messages that indicate a NotOnLink status.

When the client receives a NotOnLink status from the server in response to a Request, the client can either re-issue the Request without specifying any addresses or restart the DHCP server discovery process (see section 17).

The client SHOULD perform duplicate address detection [17] on each of the received addresses in any IAs, on which it has not performed duplicate address detection during processing of any of the previous Reply messages from the server. The client performs the duplicate address detection before using the received addresses for

the traffic. If any of the addresses are found to be in use on the link, the client sends a Decline message to the server for those addresses as described in section 18.1.7.

If the Reply was received in response to a Solicit (with a Rapid Commit option), Request, Renew or Rebind message, the client updates the information it has recorded about IAs from the IA options contained in the Reply message:

- Record T1 and T2 times.
- Add any new addresses in the IA option to the IA as recorded by the client.
- Update lifetimes for any addresses in the IA option that the client already has recorded in the IA.
- Discard any addresses from the IA, as recorded by the client, that have a valid lifetime of 0 in the IA Address option.
- Leave unchanged any information about addresses the client has recorded in the IA but that were not included in the IA from the server.

Management of the specific configuration information is detailed in the definition of each option in section 22.

The client examines the status code in each IA individually. If the client receives a NoAddrsAvail status code, the client has received no usable addresses in the IA.

If the client can operate with the addresses obtained from the server the client uses addresses and other information from any IAs that do not contain a Status Code option with the NoAddrsAvail status code. The client MAY include the IAs for which it received the NoAddrsAvail status code, with no addresses, in subsequent Renew and Rebind messages sent to the server, to retry obtaining the addresses for these IAs.

If the client cannot operate without the addresses for the IAs for which it received the NoAddrsAvail status code, the client may try another server (perhaps by restarting the DHCP server discovery process).

If the client finds no usable addresses in any of the IAs, it may either try another server (perhaps restarting the DHCP server discovery process) or use the Information-request message to obtain other configuration information only.

When the client receives a Reply message in response to a Renew or Rebind message, the client:

- sends a Request message if any of the IAs in the Reply message contains the NoBinding status code. The client places IA options in this message for only those IAs for which the server returned the NoBinding status code in the Reply message. The client continues to use other bindings for which the server did not return an error
- sends a Renew/Rebind if any of the IAs is not in the Reply message, but in this case the client MUST limit the rate at which it sends these messages, to avoid the Renew/Rebind storm
- otherwise accepts the information in the IA.

When the client receives a valid Reply message in response to a Release message, the client considers the Release event completed, regardless of the Status Code option(s) returned by the server.

When the client receives a valid Reply message in response to a Decline message, the client considers the Decline event completed, regardless of the Status Code option(s) returned by the server.

#### 4.4.6. Updates to Section 18.2.3 of RFC 3315

Replace Section 18.2.3 of RFC 3315 with the following text:

When the server receives a Renew message via unicast from a client to which the server has not sent a unicast option, the server discards the Renew message and responds with a Reply message containing a Status Code option with the value UseMulticast, a Server Identifier option containing the server's DUID, the Client Identifier option from the client message, and no other options.

For each IA in the Renew message from a client, the server locates the client's binding and verifies that the information in the IA from the client matches the information stored for that client.

If the server finds the client entry for the IA the server sends back the IA to the client with new lifetimes and, if applicable, T1/T2 times. If the server is unable to extend the lifetimes of an address in the IA, the server MAY choose not to include the IA Address option for this address.

The server may choose to change the list of addresses and the lifetimes of addresses in IAs that are returned to the client.



If the server finds that any of the addresses in the IA are not appropriate for the link to which the client is attached, the server returns the address to the client with lifetimes of 0.

For each IA for which the server cannot find a client entry, the server has the following choices depending on the server's policy and configuration information:

- If the server is configured to create new bindings as a result of processing Renew messages, the server SHOULD create a binding and return the IA with allocated addresses with lifetimes and, if applicable, T1/T2 times and other information requested by the client. The server MAY use values in the IA Address option (if included) as a hint.
- If the server is configured to create new bindings as a result of processing Renew messages, but the server will not assign any addresses to an IA, the server returns the IA option containing a Status Code option with the NoAddrsAvail status code and a status message for a user.
- If the server does not support creation of new bindings for the client sending a Renew message, or if this behavior is disabled according to the server's policy or configuration information, the server returns the IA option containing a Status code option with the NoBinding status code and a status message for a user.

The server constructs a Reply message by setting the "msg-type" field to REPLY, and copying the transaction ID from the Renew message into the transaction-id field.

The server MUST include a Server Identifier option containing the server's DUID and the Client Identifier option from the Renew message in the Reply message.

The server includes other options containing configuration information to be returned to the client as described in section 18.2.

The T1/T2 times set in each applicable IA option for a Reply MUST be the same values across all IAs. The server MUST determine the T1/T2 times across all of the applicable client's bindings in the Reply. This facilitates the client being able to renew all of the bindings at the same time.

## 4.4.7. Updates to Section 18.2.4 of RFC 3315

Replace Section 18.2.4 of RFC 3315 with the following text:

When the server receives a Rebind message that contains an IA option from a client, it locates the client's binding and verifies that the information in the IA from the client matches the information stored for that client.

If the server finds the client entry for the IA and the server determines that the addresses in the IA are appropriate for the link to which the client's interface is attached according to the server's explicit configuration information, the server SHOULD send back the IA to the client with new lifetimes and, if applicable, T1/T2 times. If the server is unable to extend the lifetimes of an address in the IA, the server MAY choose not to include the IA Address option for this address.

If the server finds the client entry for the IA and any of the addresses are no longer appropriate for the link to which the client's interface is attached according to the server's explicit configuration information, the server returns the address to the client with lifetimes of 0.

If the server cannot find a client entry for the IA, the IA contains addresses and the server determines that the addresses in the IA are not appropriate for the link to which the client's interface is attached according to the server's explicit configuration information, the server MAY send a Reply message to the client containing the client's IA, with the lifetimes for the addresses in the IA set to 0. This Reply constitutes an explicit notification to the client that the addresses in the IA are no longer valid. In this situation, if the server does not send a Reply message it silently discards the Rebind message.

Otherwise, for each IA for which the server cannot find a client entry, the server has the following choices depending on the server's policy and configuration information:

- If the server is configured to create new bindings as a result of processing Rebind messages (also see the note about the Rapid Commit option below), the server SHOULD create a binding and return the IA with allocated addresses with lifetimes and, if applicable, T1/T2 times and other information requested by the client. The server MAY use values in the IA Address option (if included) as a hint.

- If the server is configured to create new bindings as a result of processing Rebind messages, but the server will not assign any addresses to an IA, the server returns the IA option containing a Status Code option with the NoAddrsAvail status code and a status message for a user.
- If the server does not support creation of new bindings for the client sending a Rebind message, or if this behavior is disabled according to the server's policy or configuration information, the server returns the IA option containing a Status Code option with the NoBinding status code and a status message for a user.

When the server creates new bindings for the IA it is possible that other servers also create bindings as a result of receiving the same Rebind message. This is the same issue as in the Discussion under the Rapid Commit option, see section 22.14. Therefore, the server SHOULD only create new bindings during processing of a Rebind message if the server is configured to respond with a Reply message to a Solicit message containing the Rapid Commit option.

The server constructs a Reply message by setting the "msg-type" field to REPLY, and copying the transaction ID from the Rebind message into the transaction-id field.

The server MUST include a Server Identifier option containing the server's DUID and the Client Identifier option from the Rebind message in the Reply message.

The server includes other options containing configuration information to be returned to the client as described in section 18.2.

The T1/T2 times set in each applicable IA option for a Reply MUST be the same values across all IAs. The server MUST determine the T1/T2 times across all of the applicable client's bindings in the Reply. This facilitates the client being able to renew all of the bindings at the same time.

#### 4.4.8. Updates to RFC 3633

Replace the following text in Section 12.1 of RFC 3633:

Each prefix has valid and preferred lifetimes whose durations are specified in the IA\_PD Prefix option for that prefix. The requesting router uses Renew and Rebind messages to request the extension of the lifetimes of a delegated prefix.

With:

Each prefix has valid and preferred lifetimes whose durations are specified in the IA\_PD Prefix option for that prefix. The requesting router uses Renew and Rebind messages to request the extension of the lifetimes of a delegated prefix.

The requesting router MAY include IA\_PD options without any prefixes, i.e. without IA Prefix option or with IPv6 prefix field of IA Prefix option set to 0, in a Renew or Rebind message to obtain bindings it desires but has been unable to obtain. The requesting router MAY set the prefix-length field of the IA Prefix option as a hint to the server. As in [RFC3315], the requesting router MAY also provide lifetime hints in the IA Prefix option.

Replace the following text in Section 12.2 of RFC 3633:

The delegating router behaves as follows when it cannot find a binding for the requesting router's IA\_PD:

With:

For the Renew or Rebind, if the IA\_PD contains no IA Prefix option or it contains an IA Prefix option with the IPv6 prefix field set to 0, the delegating router SHOULD assign prefixes to the IA\_PD according to the delegating router's explicit configuration information. In this case, if the IA\_PD contains an IA Prefix option with the IPv6 prefix field set to 0, the delegating router MAY use the value in the prefix-length field of the IA Prefix option as a hint for the length of the prefixes to be assigned. The delegating router MAY also respect lifetime hints provided by the requesting router in the IA Prefix option.

The delegating router behaves as follows when it cannot find a binding for the requesting router's IA\_PD containing prefixes:

#### 4.5. Confirm Message

The Confirm message, as described in [RFC3315], is specific to address assignment. It allows a server without a binding to reply to the message, under the assumption that the server only needs knowledge about the prefix(es) on the link, to inform the client that the address is likely valid or not. This message is sent when e.g. the client has moved and needs to validate its addresses. Not all bindings can be validated by servers and the Confirm message provides for this by specifying that a server that is unable to determine the on-link status MUST NOT send a Reply.

Note: Confirm has a specific meaning and does not overload Renew/Rebind. It also is lower processing cost as the server does NOT need to extend lease times or otherwise send back other configuration options.

The Confirm message is used by the client to verify that it has not moved to a different link. For IAs with addresses, the mechanism used to verify if a client has moved or not, is by matching the link's on-link prefix(es) (typically a /64) against the prefix-length first bits of the addresses provided by the client in the IA\_NA or IA\_TA IA-types. As a consequence Confirm can only be used when the client has an IA with address(es) (IA\_NA or IA\_TA).

A client MUST have a binding including an IA with addresses to use the Confirm message. A client with IAs with addresses as well as other IA-types MAY, depending on the IA-type, use the Confirm message to detect if the client has moved to a different link. A client that does not have a binding with an IA with addresses MUST use the Rebind message instead.

IA\_PD requires verification that the delegating router (server) has the binding for the IAs. In that case a requesting router (client) MUST use the Rebind message in place of the Confirm message and it MUST include all of its bindings, even address IAs.

Note that Section 18.1.2 of RFC 3315 states that a client MUST initiate a Confirm when it may have moved to a new link. This is relaxed to a SHOULD as a client may have determined whether it has or has not moved using other techniques, such as described in [RFC6059]. And, as stated above, a client with delegated prefixes, MUST send a Rebind instead of a Confirm.

#### 4.6. Decline Should Not Necessarily Trigger a Release

Some client implementations have been found to send a Release message for other bindings they may have received after they determine a conflict and have correctly sent a Decline message for the conflicting address(es).

A client SHOULD NOT send a Release message for other bindings it may have received just because it sent a Decline message. The client SHOULD retain the non-conflicting bindings. The client SHOULD treat the failure to acquire a binding as a result of the conflict, to be equivalent to not having received the binding, insofar as it behaves when sending Renew and Rebind messages.

#### 4.7. Multiple Provisioning Domains

This document has assumed that all DHCP servers on a network are in a single provisioning domain and thus should be "equal" in the service that they offer. This was also assumed by [RFC3315] and [RFC3633].

One could envision a network where the DHCP servers are in multiple provisioning domains, and it may be desirable to have the DHCP client obtain different IA types from different provisioning domains. How a client detects the multiple provisioning domains and how it would interact with the multiple servers in these different domains is outside the scope of this document (see [I-D.ietf-mif-mpvd-arch] and [I-D.ietf-mif-mpvd-dhcp-support]).

#### 5. IANA Considerations

This specification does not require any IANA actions.

#### 6. Security Considerations

There are no new security considerations pertaining to this document.

#### 7. Acknowledgements

Thanks to many people that contributed to identify the stateful issues addressed by this document and for reviewing drafts of the document, including Ralph Droms, John Brzozowski, Ted Lemon, Hemant Singh, Wes Beebe, Gaurau Halwasia, Bud Millword, Tim Winters, Rob Shakir, Jinmei Tatuya, Andrew Yourtchenko, Fred Templin, Tomek Mrugalski, Suresh Krishnan, and Ian Farrer.

#### 8. References

##### 8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.
- [RFC7083] Droms, R., "Modification to Default Values of SOL\_MAX\_RT and INF\_MAX\_RT", RFC 7083, November 2013.

## 8.2. Informative References

- [I-D.dhcgw-dhc-rfc3315bis]  
Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A.,  
Richardson, M., Jiang, S., and T. Lemon, "Dynamic Host  
Configuration Protocol for IPv6 (DHCPv6) bis", draft-  
dhcgw-dhc-rfc3315bis-04 (work in progress), February 2015.
- [I-D.ietf-mif-mpvd-arch]  
Anipko, D., "Multiple Provisioning Domain Architecture",  
draft-ietf-mif-mpvd-arch-11 (work in progress), March  
2015.
- [I-D.ietf-mif-mpvd-dhcp-support]  
Krishnan, S., Korhonen, J., and S. Bhandari, "Support for  
multiple provisioning domains in DHCPv6", draft-ietf-mif-  
mpvd-dhcp-support-01 (work in progress), March 2015.
- [RFC6059] Krishnan, S. and G. Daley, "Simple Procedures for  
Detecting Network Attachment in IPv6", RFC 6059, November  
2010.
- [RFC7084] Singh, H., Beebe, W., Donley, C., and B. Stark, "Basic  
Requirements for IPv6 Customer Edge Routers", RFC 7084,  
November 2013.
- [RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and  
S. Krishnan, "Guidelines for Creating New DHCPv6 Options",  
BCP 187, RFC 7227, May 2014.

## Authors' Addresses

Ole Troan  
Cisco Systems, Inc.  
Philip Pedersens vei 20  
N-1324 Lysaker  
Norway

Email: ot@cisco.com

Bernie Volz  
Cisco Systems, Inc.  
1414 Massachusetts Ave  
Boxborough, MA 01719  
USA

Email: volz@cisco.com

Marcin Siodelski  
ISC  
950 Charter Street  
Redwood City, CA 94063  
USA

Email: [msiodelski@gmail.com](mailto:msiodelski@gmail.com)



DHC Working Group  
Internet-Draft  
Updates: 3315 (if approved)  
Intended status: Standards Track  
Expires: September 28, 2014

Y. Cui  
Q. Sun  
Tsinghua University  
T. Lemon  
Nominum, Inc.  
March 27, 2014

Handling Unknown DHCPv6 Messages  
draft-ietf-dhc-dhcpv6-unknown-msg-08

Abstract

DHCPv6 is not specific about handling messages with unknown types. This memo describes the problems and defines how a DHCPv6 server, client or relay agent should behave when receiving unknown DHCPv6 messages. This document also provides advice for authors of future documents defining new messages sent from DHCP servers to DHCP relay agents, and should be read by potential authors of such documents. This document updates RFC3315.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 28, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

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

## Table of Contents

1. Introduction . . . . .	2
2. Requirements Language . . . . .	2
3. Problem Statement . . . . .	3
4. Relay Agent Behavior Update . . . . .	3
4.1. A Valid Message for Constructing a New Relay-forward Message . . . . .	3
4.2. Relaying a Message toward Server . . . . .	4
4.3. Relaying a Message toward Client . . . . .	4
5. Client and Server Behavior Update . . . . .	5
6. Security Considerations . . . . .	5
7. IANA Considerations . . . . .	6
8. Contributors List . . . . .	6
9. Normative References . . . . .	6
Authors' Addresses . . . . .	6

## 1. Introduction

DHCPv6 [RFC3315] provides a framework for conveying IPv6 configuration information to hosts on a TCP/IP network. But [RFC3315] is not specific about how to deal with messages with unrecognized types. This document describes the problems and defines the behavior of a DHCPv6 server, client or relay agent when handling unknown DHCPv6 messages.

## 2. 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 [RFC2119].

### 3. Problem Statement

When a relay agent receives a message, it decides to send the message either toward the server or toward the client. It decides on the direction to forward based on the message type. Since RFC3315 was published new message types have been defined. More such messages may be defined in the future. RFC3315 does not specify the what to do when a DHCP agent does not recognize the type of message it has received. This may lead to relay agents inappropriately dropping such messages, and to other DHCP agents inappropriately processing such messages.

In addition, there is no specific requirement for dealing with unknown messages by the client or server in RFC3315.

Note it is expected that most future DHCPv6 messages will not be used to communicate directly with relay agents (though they may need to be relayed by relay agents).

### 4. Relay Agent Behavior Update

Relay agents relay messages toward servers and clients according to the message type. The Relay-reply message is sent toward the client. The Relay-forward message and other types of messages are sent toward the server.

We say "toward the client" and "toward the server" because relay agents may be chained together, so a relay message may be sent through multiple relay agents along the path to its destination. Relay-reply messages specify a destination address; the relay agent extracts the encapsulated message and sends it to the specified destination address. Any message other than a Relay-reply does not have such a specified destination, so it follows the default forwarding path configured on the relay agent, which is always toward the server.

The sole purpose of requiring relay agents to relay unknown messages is to ensure that when legitimate new messages are defined in the protocol, relay agents, even if they were manufactured prior to the definition of these new messages, will, by default, succeed in relaying such messages.

#### 4.1. A Valid Message for Constructing a New Relay-forward Message

Section 20.1 of [RFC3315] states that:

"When a relay agent receives a valid message to be relayed, it constructs a new Relay-forward message."

It does not define which types of messages are valid for constructing Relay-Forward messages. In this document, we specify the definition as follows.

The message is valid for constructing a new Relay-forward message:

- (a) if the message is a Relay-forward message, or
- (b) if the relay agent recognizes the message type and is not the intended target, or
- (c) if the relay agent does not recognize the message type.

New DHCP message types may be defined in future that are sent, unsolicited, to relay agents. Relay agents that do not implement these messages will not recognize such messages as being intended for them. A relay agent that implements this specification will therefore forward such messages to the DHCP servers to which it is configured to relay client messages.

At this time, no such message types have been specified. If such a message is specified in the future, it is possible that this would result in needless load on DHCP servers. If such a message type is defined in a future specification, authors may need to consider some strategy for identifying non-conforming relays and not sending such messages to them.

However, since DHCP servers do not respond to unknown messages, this is unlikely to create significant load, and therefore is likely to be unnecessary.

#### 4.2. Relaying a Message toward Server

If the relay agent receives a Relay-forward message, Section 20.1.2 of [RFC3315] defines the required behavior. If the relay agent receives messages other than Relay-forward and Relay-reply and the relay agent does not recognize its message type, it MUST forward them as is described in Section 20.1.1 of [RFC3315].

#### 4.3. Relaying a Message toward Client

If the relay agent receives a Relay-reply message, it MUST process the message as is defined in Section 20.2 of [RFC3315], regardless of the type of the message encapsulated in the Relay Message Option.

## 5. Client and Server Behavior Update

A client or server MUST silently discard any received DHCPv6 message with an unknown message type.

## 6. Security Considerations

This document creates no new security issues that are not already present in RFC3315. By explicitly documenting the correct handling of unknown messages, this document, if implemented, reduces any security exposure that might result from incorrect handling of unknown messages. The following issues are issues that could already be present with section 23 of [RFC3315], but we discuss them in detail here as guidance for implementors.

As the relay agent will forward all unknown types of DHCPv6 messages, a malicious attacker can interfere with the relaying function by constructing fake DHCPv6 messages with arbitrary type code. The same problem may happen in current DHCPv4 and DHCPv6 practice where the attacker constructs the fake DHCP message with a known type code.

Clients and servers that implement this specification will discard unknown DHCPv6 messages. Since RFC3315 did not specify either relay agent, client or server behavior in the presence of unknown messages, it is possible that some servers or clients that have not been updated to conform to this specification might be made vulnerable to client attacks through the relay agent.

For this reason, we recommend that relay agents, clients and servers be updated to follow this new specification. However, in most deployment scenarios, it will be much easier to attack clients directly than through a relay agent; furthermore, attacks using unknown message types are already possible on the local wire.

So in most cases, if clients are not upgraded there should be minimal additional risk; at sites where only servers and relay agents can be upgraded, the incremental benefit of doing so most likely exceeds any risk due to vulnerable clients.

Nothing in this update should be construed to mean that relay agents may not be administratively configurable to drop messages on the basis of the message type, for security reasons (e.g., in a firewall).

## 7. IANA Considerations

This document does not include an IANA request.

## 8. Contributors List

Many thanks to Bernie Volz, Tomek Mrugalski, Sheng Jiang, Cong Liu and Yuchi Chen for their contributions to the document.

## 9. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

## Authors' Addresses

Yong Cui  
Tsinghua University  
Beijing 100084  
P.R.China

Phone: +86-10-6260-3059  
Email: yong@csnet1.cs.tsinghua.edu.cn

Qi Sun  
Tsinghua University  
Beijing 100084  
P.R.China

Phone: +86-10-6278-5822  
Email: sunqi@csnet1.cs.tsinghua.edu.cn

Ted Lemon  
Nominum, Inc.  
2000 Seaport Blvd  
Redwood City, CA 94063  
USA

Phone: +1-650-381-6000  
Email: Ted.Lemon@nominum.com

Dynamic Host Configuration Working Group  
Internet-Draft  
Updates: 3315 (if approved)  
Intended status: Best Current Practice  
Expires: July 11, 2014

D. Hankins  
Google  
T. Mrugalski  
M. Siodelski  
ISC  
S. Jiang  
Huawei Technologies Co., Ltd  
S. Krishnan  
Ericsson  
January 7, 2014

Guidelines for Creating New DHCPv6 Options  
draft-ietf-dhc-option-guidelines-17

Abstract

This document provides guidance to prospective DHCPv6 Option developers to help them creating option formats that are easily adoptable by existing DHCPv6 software. It also provides guidelines for expert reviewers to evaluate new registrations. This document updates RFC3315.

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 July 11, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Requirements Language . . . . .	3
2. Introduction . . . . .	3
3. When to Use DHCPv6 . . . . .	4
4. General Principles . . . . .	4
5. Reusing Other Options Formats . . . . .	5
5.1. Option with IPv6 addresses . . . . .	6
5.2. Option with a single flag (boolean) . . . . .	7
5.3. Option with IPv6 prefix . . . . .	7
5.4. Option with 32-bit integer value . . . . .	8
5.5. Option with 16-bit integer value . . . . .	9
5.6. Option with 8-bit integer value . . . . .	9
5.7. Option with URI . . . . .	9
5.8. Option with Text String . . . . .	11
5.9. Option with variable length data . . . . .	12
5.10. Option with DNS Wire Format Domain Name List . . . . .	12
6. Avoid Conditional Formatting . . . . .	13
7. Avoid Aliasing . . . . .	13
8. Choosing between FQDN and address . . . . .	14
9. Encapsulated options in DHCPv6 . . . . .	17
10. Additional States Considered Harmful . . . . .	18
11. Configuration changes occur at fixed times . . . . .	19
12. Multiple provisioning domains . . . . .	20
13. Chartering Requirements and Advice for Responsible Area Directors . . . . .	20
14. Considerations for Creating New Formats . . . . .	21
15. Option Size . . . . .	22
16. Singleton options . . . . .	22
17. Option Order . . . . .	23
18. Relay Options . . . . .	24
19. Clients Request their Options . . . . .	24
20. Transition Technologies . . . . .	25
21. Recommended sections in the new document . . . . .	25
21.1. DHCPv6 Client Behavior Text . . . . .	26
21.2. DHCPv6 Server Behavior Text . . . . .	27
21.3. DHCPv6 Relay Agent Behavior Text . . . . .	27
22. Should the new document update existing RFCs? . . . . .	27
23. Security Considerations . . . . .	28
24. Privacy considerations . . . . .	29
25. IANA Considerations . . . . .	29



26. Acknowledgements . . . . .	30
27. References . . . . .	30
27.1. Normative References . . . . .	30
27.2. Informative References . . . . .	30
Authors' Addresses . . . . .	33

## 1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

## 2. Introduction

Most protocol developers ask themselves if a protocol will work, or work efficiently. These are important questions, but another less frequently considered question is whether the proposed protocol presents itself needless barriers to adoption by deployed software.

DHCPv6 [RFC3315] software implementors are not merely faced with the task of handling a given option's format on the wire. The option must fit into every stage of the system's process, starting with the user interface used to enter the configuration up to the machine interfaces where configuration is ultimately consumed.

Another frequently overlooked aspect of rapid adoption is whether the option requires operators to be intimately familiar with the option's internal format in order to use it? Most DHCPv6 software provides a facility for handling unknown options at the time of publication. The handling of such options usually needs to be manually configured by the operator. But if doing so requires extensive reading (more than can be covered in a simple FAQ for example), it inhibits adoption.

So although a given solution would work, and might even be space, time, or aesthetically optimal, a given option is presented with a series of ever-worsening challenges to be adopted:

- o If it doesn't fit neatly into existing config files.
- o If it requires source code changes to be adopted, and hence upgrades of deployed software.
- o If it does not share its deployment fate in a general manner with other options, standing alone in requiring code changes or reworking configuration file syntaxes.

- o If the option would work well in the particular deployment environment the proponents currently envision, but has equally valid uses in some other environment where the proposed option format would fail or would produce inconsistent results.

There are many things DHCPv6 option creators can do to avoid the pitfalls in this list entirely, or failing that, to make software implementors lives easier and improve its chances for widespread adoption.

This document is envisaged as a help for protocol developers that define new options and for expert reviewers that review submitted proposals.

### 3. When to Use DHCPv6

Principally, DHCPv6 carries configuration parameters for its clients. Any knob, dial, slider, or checkbox on the client system, such as "my domain name servers", "my hostname", or even "my shutdown temperature" are candidates for being configured by DHCPv6.

The presence of such a knob isn't enough, because DHCPv6 also presents the extension of an administrative domain - the operator of the network to which the client is currently attached. Someone runs not only the local switching network infrastructure that the client is directly (or wirelessly) attached to, but the various methods of accessing the external Internet via local assist services that the network must also provide (such as domain name servers, or routers). This means that, even if a configuration parameter can potentially be delivered by DHCPv6, it is necessary to evaluate whether it is reasonable for this parameter to be under the control of the administrator of whatever network a client is attached to at any given time.

Note that the client is not required to configure any of these values received via DHCPv6 (e.g., due to having these values locally configured by its own administrator). But it needs to be noted that overriding DHCPv6-provided values may cause the client to be denied certain services in the network to which it has attached. The possibility of having higher level of control over client node configuration is one of the reasons that DHCPv6 is preferred in enterprise networks.

### 4. General Principles

The primary guiding principle to follow in order to enhance an option's adoptability is reuse. The option should be created in such a way that does not require any new or special case software to

support. If old software currently deployed and in the field can adopt the option through supplied configuration facilities then it's fairly certain that new software can easily formally adopt it.

There are at least two classes of DHCPv6 options: simple options which are provided explicitly to carry data from one side of the DHCPv6 exchange to the other (such as nameservers, domain names, or time servers), and a protocol class of options which require special processing on the part of the DHCPv6 software or are used during special processing (such as the Fully Qualified Domain Name (FQDN) option [RFC4704]), and so forth; these options carry data that is the result of a routine in some DHCPv6 software.

The guidelines laid out here should be applied in a relaxed manner for the protocol class of options. Wherever special case code is already required to adopt the DHCPv6 option, it is substantially more reasonable to format the option in a less generic fashion, if there are measurable benefits to doing so.

## 5. Reusing Other Options Formats

The easiest approach to manufacturing trivially deployable DHCPv6 Options is to assemble the option out of whatever common fragments fit - possibly allowing a group of data elements to repeat to fill the remaining space (if present) and so provide multiple values. Place all fixed size values at the start of the option, and any variable/indeterminate sized value at the tail end of the option.

This means that implementations will likely be able to reuse code paths designed to support the other options.

There is a tradeoff between the adoptability of previously defined option formats, and the advantages that new or specialized formats can provide. In general, it is usually preferable to reuse previously used option formats.

However, it isn't very practical to consider the bulk of DHCPv6 options already allocated, and consider which of those solve a similar problem. So, the following list of common option format data elements is provided as a shorthand. Please note that it is not complete in terms of exemplifying every option format ever devised.

If more complex options are needed, those basic formats mentioned here may be considered as primitives (or 'fragment types') that can be used to build more complex formats. It should be noted that it is often easier to implement two options with trivial formats than one option with more complex format. That is not unconditional requirement though. In some cases splitting one complex option into

two or more simple options introduces inter-option dependencies that should be avoided. In such a case, it is usually better to keep one complex option.

#### 5.1. Option with IPv6 addresses

This option format is used to carry one or many IPv6 addresses. In some cases the number of allowed address is limited (e.g. to one):

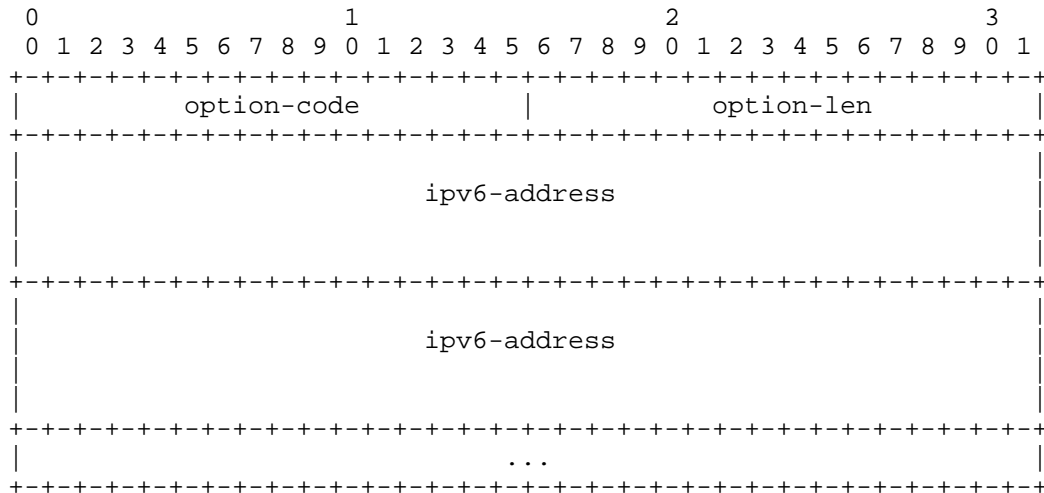


Figure 1: Option with IPv6 address

Examples of use:

- o DHCPv6 server unicast address (a single address only) [RFC3315]
- o SIP Servers IPv6 Address List [RFC3319]
- o DNS Recursive Name Server [RFC3646]
- o NIS Servers [RFC3898]
- o SNTP Servers [RFC4075]
- o Broadcast and Multicast Service Controller IPv6 Address Option for DHCPv6 [RFC4280]
- o MIPv6 Home Agent Address [RFC6610] (a single address only)
- o NTP server [RFC5908] (a single address only)

- o NTP Multicast address [RFC5908] (a single address only)

## 5.2. Option with a single flag (boolean)

Sometimes it is useful to convey a single flag that can take either on or off values. Instead of specifying an option with one bit of usable data and 7 bits of padding, it is better to define an option without any content. It is the presence or absence of the option that conveys the value. This approach has the additional benefit of absent option designating the default, i.e. administrator has to take explicit actions to deploy the opposite of the default value.

The absence of the option represents the default value and the presence of the option represents the other value, but that does not necessarily mean that absence is "off" (or "false") and presence is "on" (or "true"). That is, if it's desired that the default value for a bistable option is "true"/"on", then the presence of that option would turn it off (make it false). If the option presence signifies off/false state, that should be reflected in the option name, e.g. OPTION\_DISABLE\_FOO.

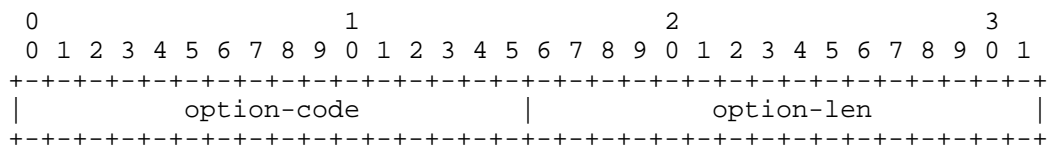


Figure 2: Option for conveying boolean

Examples of use:

- o DHCPv6 rapid-commit [RFC3315]

## 5.3. Option with IPv6 prefix

Sometimes there is a need to convey an IPv6 prefix. The information to be carried by such an option includes the 128-bit IPv6 prefix together with a length of this prefix taking values from 0 to 128. Using the simplest approach, the option could convey this data in two fixed length fields: one carrying prefix length, another carrying the prefix. However, in many cases /64 or shorter prefixes are used. This implies that the large part of the prefix data carried by the option would have its bits set to zero and would be unused. In order to avoid carrying unused data, it is recommended to store prefix in the variable length data field. The appropriate option format is defined as follows:

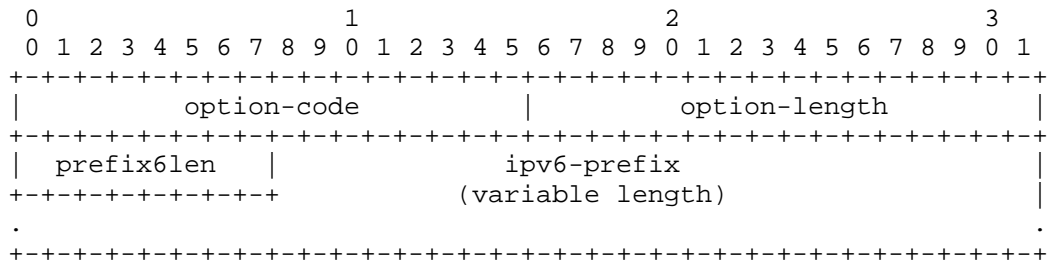


Figure 3: Option with IPv6 Prefix

option-length is set to 1 + length of the IPv6 prefix.

prefix6len is one octet long and specifies the length in bits of the IPv6 prefix. Typically allowed values are 0 to 128.

ipv6-prefix field is a variable length field that specifies the IPv6 prefix. The length is  $(\text{prefix6len} + 7) / 8$ . This field is padded with zero bits up to the nearest octet boundary when prefix6len is not divisible by 8.

Examples of use:

- o Default Mapping Rule [I-D.ietf-software-map-dhcp]

For example, the prefix 2001:db8::/60 would be encoded with an option-length of 9, prefix6-len would be set to 60, the ipv6-prefix would be 8 octets and would contain octets 20 01 0d b8 00 00 00 00.

It should be noted that the IAPREFIX option defined by [RFC3633] uses a full length 16-octet prefix field. The concern about option length was not well understood at the time of its publication.

#### 5.4. Option with 32-bit integer value

This option format can be used to carry 32 bit-signed or unsigned integer value:

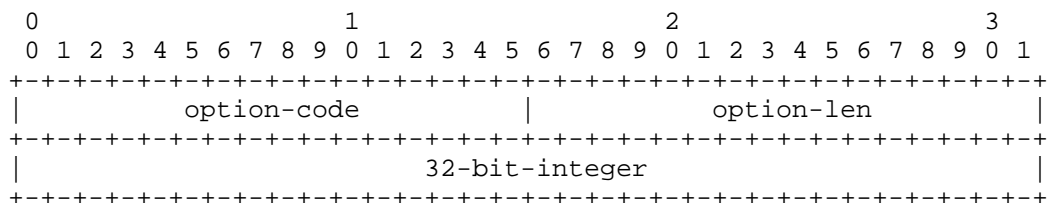


Figure 4: Option with 32-bit-integer value

Examples of use:

- o Information Refresh Time [RFC4242]

#### 5.5. Option with 16-bit integer value

This option format can be used to carry 16-bit signed or unsigned integer values:

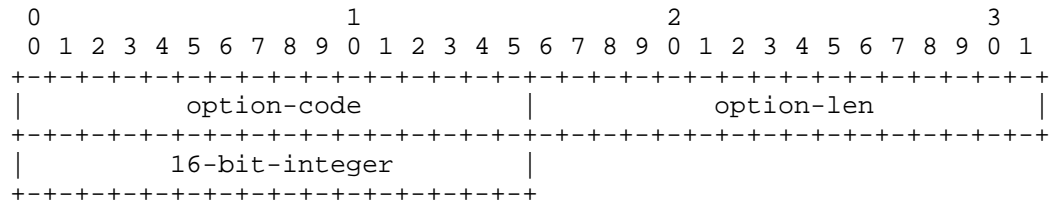


Figure 5: Option with 16-bit integer value

Examples of use:

- o Elapsed Time [RFC3315]

#### 5.6. Option with 8-bit integer value

This option format can be used to carry 8-bit integer values:

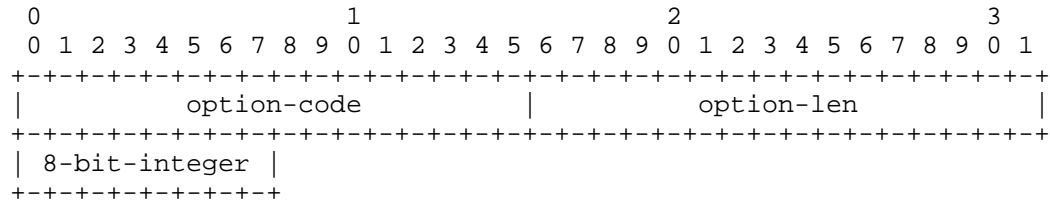


Figure 6: Option with 8-bit integer value

Examples of use:

- o DHCPv6 Preference [RFC3315]

#### 5.7. Option with URI

A Uniform Resource Identifier (URI) [RFC3986] is a compact sequence of characters that identifies an abstract or physical resource. The term "Uniform Resource Locator" (URL) refers to the subset of URIs that, in addition to identifying a resource, provide a means of locating the resource by describing its primary access mechanism

(e.g., its network "location"). This option format can be used to carry a single URI:

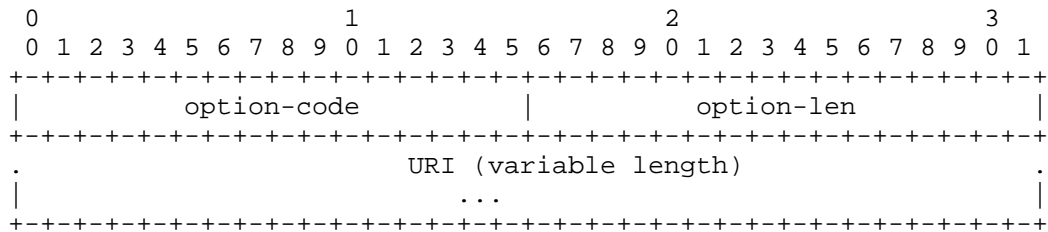


Figure 7: Option with URI

Examples of use:

- o Boot File URL [RFC5970]

An alternate encoding to support multiple URIs is available. An option must be defined to use either the single URI format above or the multiple URI format below depending on whether a single is always sufficient or if multiple URIs are possible.

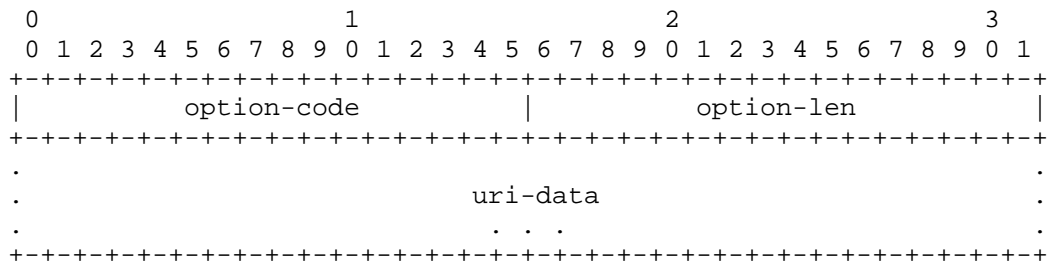
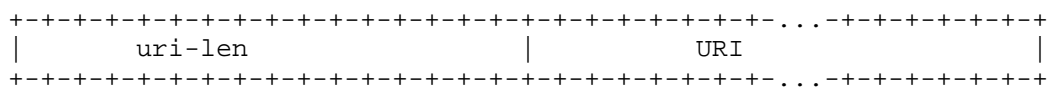


Figure 8: Option with multiple URIs

Each instance of the uri-data is formatted as follows:



The uri-len is two octets long and specifies the length of the uri data. Although URI format in theory supports up to 64k of data, in practice large chunks of data may be problematic. See Section 15 for details.



### 5.8. Option with Text String

A text string is a sequence of characters that have no semantics. The encoding of the text string **MUST** be specified. Unless otherwise specified, all text strings in newly defined options are expected to be Unicode strings that are encoded using UTF-8 [RFC3629] in Net-Unicode form [RFC5198]. Please note that all strings containing only 7 bit ASCII characters are also valid UTF-8 Net-Unicode strings.

If a data format has semantics other than just being text, it is not a string. E.g., a FQDN is not a string, and a URI is also not a string, because they have different semantics. A string must not include any terminator (such as a null byte). The null byte is treated as any other character and does not have any special meaning. This option format can be used to carry a text string:

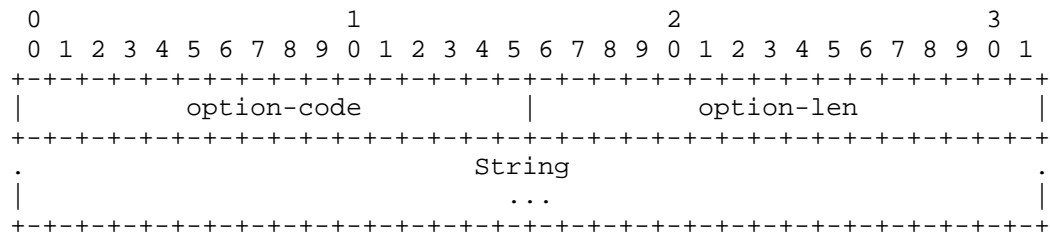


Figure 9: Option with text string

Examples of use:

- o Timezone Options for DHCPv6 [RFC4833]

An alternate encoding to support multiple text strings is available. An option must be defined to use either the single text string format above or the multiple text string format below depending on whether a single is always sufficient or if multiple text strings are possible.

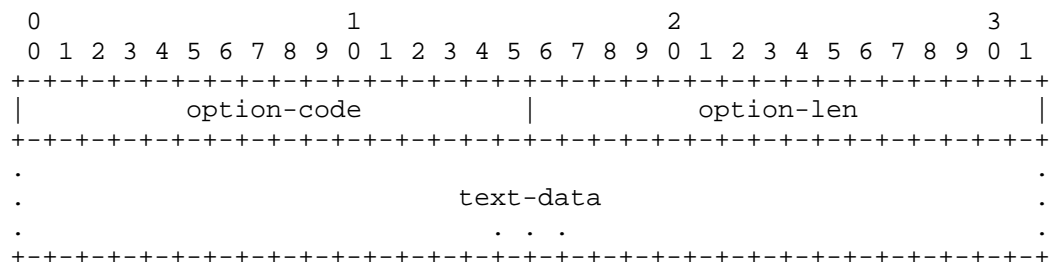


Figure 10: Option with multiple text strings

Each instance of the text-data is formatted as follows:

```
+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+
|           text-len           |           String           |
+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+
```

The text-len is two octets long and specifies the length of the string.

#### 5.9. Option with variable length data

This option can be used to carry variable length data of any kind. Internal representation of carried data is option specific. Whenever this format is used by the new option being defined, the data encoding should be documented.

This option format provides a lot of flexibility to pass data of almost any kind. Though, whenever possible it is highly recommended to use more specialized options, with field types better matching carried data types.

```

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
+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+
|           option-code           |           option-len           |
+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+
.
.           variable length data           .
.
+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+
```

Figure 11: Option with variable length data

Examples of use:

- o Client Identifier [RFC3315]
- o Server Identifier [RFC3315]

#### 5.10. Option with DNS Wire Format Domain Name List

This option is used to carry 'domain search' lists or any host or domain name. It uses the same format as described in Section 5.9, but with the special data encoding, described in section 8 of [RFC3315]. This data encoding supports carrying multiple instances of hosts or domain names in a single option, by terminating each instance with the byte value of 0.

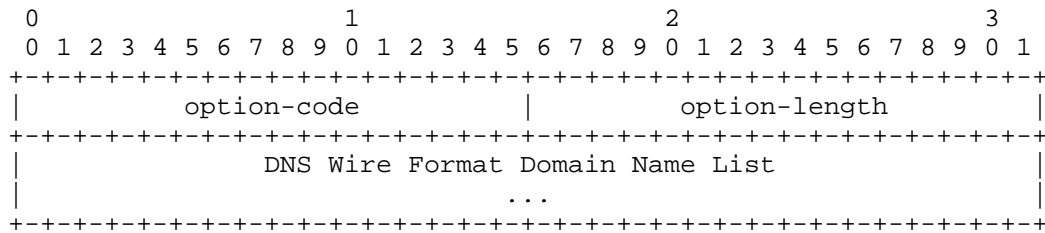


Figure 12: Option with DNS Wire Format Domain Name List

Examples of use:

- o SIP Servers Domain Name List [RFC3319] (many domains)
- o NIS Domain Name (many domains) [RFC3898] (many domains)
- o LoST Server Domain name [RFC5223]
- o LIS Domain name [RFC5986]
- o DS-Lite AFTR location [RFC6334] (a single FQDN)
- o Home Network Identifier [RFC6610] (a single FQDN)
- o Home Agent FQDN [RFC6610] (a single FQDN)

## 6. Avoid Conditional Formatting

Placing an octet at the start of the option which informs the software how to process the remaining octets of the option may appear simple to the casual observer. But the only conditional formatting methods that are in widespread use today are 'protocol' class options. Therefore conditional formatting requires new code to be written and complicates future interoperability should new conditional formats be added; and existing code has to ignore conditional format that it does not support.

## 7. Avoid Aliasing

Options are said to be aliases of each other if they provide input to the same configuration parameter. A commonly proposed example is to configure the location of some new service ("my foo server") using a binary IP address, a domain name field, and an URL. This kind of aliasing is undesirable, and is not recommended.

In this case, where three different formats are supposed, it more than triples the work of the software involved, requiring support for

not merely one format, but support to produce and digest all three. Furthermore, code development and testing must cover all possible combinations of defined formats. Since clients cannot predict what values the server will provide, they must request all formats. So in the case where the server is configured with all formats, DHCPv6 message bandwidth is wasted on option contents that are redundant. Also, the DHCPv6 option number space is wasted, as three new option codes are required, rather than one.

It also becomes unclear which types of values are mandatory, and how configuring some of the options may influence the others. For example, if an operator configures the URL only, should the server synthesize a domain name and IP address?

A single configuration value on a host is probably presented to the operator (or other software on the machine) in a single field or channel. If that channel has a natural format, then any alternative formats merely make more work for intervening software in providing conversions.

So the best advice is to choose the one method that best fulfills the requirements, be that for simplicity (such as with an IP address and port pair), late binding (such as with DNS), or completeness (such as with a URL).

#### 8. Choosing between FQDN and address

Some parameters may be specified as FQDN or an address. In most cases one or the other should be used. This section discusses pros and cons of each approach and is intended to help make an informed decision in that regard. It is strongly discouraged to define both option types at the same time (see Section 7), unless there is sufficient motivation to do so.

There is no single recommendation that works for every case. It very much depends on the nature of the parameter being configured. For parameters that are network specific or represent certain aspects of network infrastructure, like available mobility services, in most cases addresses are a more usable choice. For parameters that can be considered application specific configuration, like SIP servers, it is usually better to use FQDN.

Applications are often better suited to deal with FQDN failures than with address failures. Most operating systems provide a way to retry FQDN resolution if the previous attempt fails. That type of error recovery is supported by a great number of applications. On the other hand, there is typically no API available for applications to reconfigure over DHCP to get a new address value if the one received

is no longer appropriate. This problem may be usually addressed by providing a list of addresses, rather than just a single one. That, on the other hand, requires defined procedure how multiple addresses should be used (all at once, round robin, try first and fail over to the next if it fails etc.).

FQDN provide a higher level of indirection and ambiguity. In many cases that may be considered a benefit, but can be considered a flaw in others. For example, one operator suggested to have the same name being resolved to different addresses depending on the point of attachment of the host doing resolution. This is one way to provide localized addressing. However, in order to do this, it is necessary to violate the DNS convention that a query on a particular name should always return the same answer (aside from ordering of IP addresses in the response, which is supposed to be varied by the name server). This same locality of reference for configuration information can be achieved directly using DHCP, since the DHCP server must know the network topology in order to provide IP address or prefix configuration.

The other type of ambiguity is related to multiple provisioning domains (see Section 12). The stub resolver on the DHCP client cannot at present be assumed to make the DNS query for a DHCP-supplied FQDN on the same interface on which it received its DHCP configuration, and may therefore get a different answer from the DNS than was intended.

This is particularly a problem when the normal expected use of the option makes sense with private DNS zone(s), as might be the case on an enterprise network. It may also be the case that the client has an explicit DNS server configured, and may therefore never query the enterprise network's internal DNS server.

FQDN does require a resolution into an actual address. This implies the question when the FQDN resolution should be conducted. There are a couple of possible answers: a) by the server, when it is started, b) by the server, when it is about to send an option, c) by the client, immediately after receiving an option, d) by the client, when the content of the option is actually consumed. For a), b) and possibly c), the option should really convey an address, not FQDN. The only real incentive to use FQDN is case d). It is the only case that allows possible changes in the DNS to be picked up by clients.

If the parameter is expected to be used by constrained devices (low power, battery operated, low capabilities) or in very lossy networks, it may be appealing to drop the requirement of having DNS resolution being performed and use addresses. Another example of a constrained device is a network booted device, where despite the fact that the

node itself is very capable once it's booted, the boot prom is quite constrained.

Another aspect that should be considered is time required for the clients to notice any configuration changes. Consider a case where a server configures a service A using address and service B using FQDN. When an administrator decides to update the configuration, he or she can update the DHCP server configuration to change both services. If the clients do not support reconfigure (which is an optional feature of RFC3315, but in some environments, e.g. cable modems, is mandatory), the configuration will be updated on clients after T1 timer elapses. Depending on the nature of the change (is it a new server added to a cluster of already operating servers or a new server that replaces the only available server that crashed?), this may be an issue. On the other hand, updating service B may be achieved with DNS record update. That information may be cached by caching DNS servers for up to TTL. Depending on the values of T1 and TTL, one update may be faster than another. Furthermore, depending on the nature of the change (planned modification or unexpected failure), T1 or TTL may be lowered before the change to speed up new configuration adoption.

Simply speaking protocol designers don't know what the TTL or the T1 time will be, so they can't make assumptions about whether a DHCP option will be refreshed more quickly based on T1 or TTL.

Addresses have a benefit of being easier to implement and handle by the DHCP software. An address option is simpler to use, its validation is trivial (multiple of 16 constitutes a valid option), is explicit and does not allow any ambiguity. It is faster (does not require extra round trip time), so it is more efficient, which can be especially important for energy restricted devices. It also does not require that the client implements DNS resolution.

FQDN imposes a number of additional failure modes and issues that should be dealt with:

1. The client must have a knowledge about available DNS servers. That typically means that option DNS\_SERVERS [RFC3646] is mandatory. This should be mentioned in the draft that defines new option. It is possible that the server will return FQDN option, but not the DNS Servers option. There should be a brief discussion about it;
2. The DNS may not be reachable;
3. DNS may be available, but may not have appropriate information (e.g. no AAAA records for specified FQDN);

4. Address family must be specified (A, AAAA or any); the information being configured may require specific address family (e.g. IPv6), but there may be a DNS record only of another type (e.g. A only with IPv4 address).
5. What should the client do if there are multiple records available (use only the first one, use all, use one and switch to the second if the first fails for whatever reason, etc.); This may be an issue if there is an expectation that the parameter being configured will need exactly one address;
6. Multi-homed devices may be connected to different administrative domains with each domain providing different information in DNS (e.g. an enterprise network exposing private domains). Client may send DNS queries to a different DNS server;
7. It should be mentioned if Internationalized Domain Names are allowed. If they are, DNS option encoding should be specified.

Address options that are used with overly long T1 (renew timer) values have some characteristics of hardcoded values. That is strongly discouraged. See [RFC4085] for an in depth discussion. If the option may appear in Information-Request, its lifetime should be controlled using information refresh time option [RFC4242].

One specific case that makes the choice between address and FQDN not obvious is a DNSSEC bootstrap scenario. DNSSEC validation imposes a requirement for clock sync (to the accuracy reasonably required to consider signature inception and expiry times). This often implies usage of NTP configuration. However, if the NTP is provided as FQDN, there is no way to validate its DNSSEC signature. This is somewhat weak argument though, as providing NTP server as an address is also not verifiable using DNSSEC. If the trustworthiness of the configuration provided by DHCP server is in question, DHCPv6 offers authentication mechanisms that allow server authentication.

#### 9. Encapsulated options in DHCPv6

Most options are conveyed in a DHCPv6 message directly. Although there is no codified normative language for such options, they are often referred to as top-level options. Many options may include other options. Such inner options are often referred to as encapsulated or nested options. Those options are sometimes called sub-options, but this term actually means something else, and therefore should never be used to describe encapsulated options. It is recommended to use term "encapsulated" as this terminology is used in [RFC3315]. The difference between encapsulated and sub-options are that the former uses normal DHCPv6 option numbers, while the

latter uses option number space specific to a given parent option. It should be noted that, contrary to DHCPv4, there is no shortage of option numbers. Therefore almost all options share a common option space. For example option type 1 meant different things in DHCPv4, depending if it was located in top-level or inside of Relay Agent Information option. There is no such ambiguity in DHCPv6 (with the exception of [RFC5908], which SHOULD NOT be used as a template for future DHCP option definitions).

From the implementation perspective, it is easier to implement encapsulated options rather than sub-options, as the implementers do not have to deal with separate option spaces and can use the same buffer parser in several places throughout the code.

Such encapsulation is not limited to one level. There is at least one defined option that is encapsulated twice: Identity Association for Prefix Delegation (IA\_PD, defined in [RFC3633], section 9) conveys IA Prefix (IAPREFIX, defined in [RFC3633], section 10). Such delegated prefix may contain an excluded prefix range that is represented by PD\_EXCLUDE option that is conveyed as encapsulated inside IAPREFIX (PD\_EXCLUDE, defined in [RFC6603]). It seems awkward to refer to such options as sub-sub-option or doubly encapsulated option, therefore "encapsulated option" term is typically used, regardless of the nesting level.

When defining a DHCP-based configuration mechanism for a protocol that requires something more complex than a single option, it may be tempting to group configuration values using sub-options. That should preferably be avoided, as it increases complexity of the parser. It is much easier, faster and less error prone to parse a large number of options on a single (top-level) scope, than parse options on several scopes. The use of sub-options should be avoided as much as possible, but it is better to use sub-options rather than conditional formatting.

It should be noted that currently there is no clear way defined for requesting sub-options. Most known implementations are simply using top-level ORO for requesting both top-level options and encapsulated options.

#### 10. Additional States Considered Harmful

DHCP is a protocol designed for provisioning clients. Less experienced protocol designers often assume that it is easy to define an option that will convey a different parameter for each client in a network. Such problems arose during designs of MAP [I-D.ietf-software-map-dhcp] and 4rd [I-D.ietf-software-4rd]. While it would be easier for provisioned clients to get ready to use per-



client option values, such requirement puts exceedingly large loads on the server side. The new extensions may introduce new implementation complexity and additional database state on the server. Alternatives should be considered, if possible. As an example, [I-D.ietf-softwire-map-dhcp] was designed in a way that all clients are provisioned with the same set of MAP options and each provisioned client uses its unique address and delegated prefix to generate client-specific information. Such a solution does not introduce any additional state for the server and therefore scales better.

It also should be noted that contrary to DHCPv4, DHCPv6 keeps several timers for renewals. Each IA\_NA (addresses) and IA\_PD (prefixes) contains T1 and T2 timers that designate time after which client will initiate renewal. Those timers apply only to its own IA containers. Refreshing other parameters should be initiated after a time specified in the Information Refresh Time Option (defined in [RFC4242]), carried in the Reply message and returned in response to Information-Request message. Introducing additional timers make deployment unnecessarily complex and SHOULD be avoided.

#### 11. Configuration changes occur at fixed times

In general, DHCPv6 clients only refresh configuration data from the DHCP server when the T1 timer expires. Although there is a RECONFIGURE mechanism that allows a DHCP server to request that clients initiate reconfiguration, support for this mechanism is optional and cannot be relied upon.

Even when DHCP clients refresh their configuration information, not all consumers of DHCP-sourced configuration data notice these changes. For instance, if a server is started using parameters received in an early DHCP transaction, but does not check for updates from DHCP, it may well continue to use the same parameter indefinitely. There are a few operating systems that take care of reconfiguring services when the client moves to a new network (e.g. based on mechanisms like [RFC4436], [RFC4957] or [RFC6059]), but it's worth bearing in mind that a renew may not always result in the client taking up new configuration information that it receives.

In light of the above, when designing an option you should take into consideration the fact that your option may hold stale data that will only be updated at an arbitrary time in the future.

## 12. Multiple provisioning domains

In some cases there could be more than one DHCPv6 server on a link, with each providing a different set of parameters. One notable example of such a case is a home network with a connection to two independent ISPs.

The DHCPv6 protocol specification does not provide clear advice on how to handle multiple provisioning sources. Although [RFC3315] states that a client that receives more than one ADVERTISE message, may respond to one or more of them, such capability has not been observed in existing implementations. Existing clients will pick one server and will continue configuration process with that server, ignoring all other servers.

In addition, a node that acts as a DHCPv6 client may be connected to more than one physical network. In this case, it will in most cases operate a separate DHCP client state machine on each interface, acquiring different, possibly conflicting information through each. This information will not be acquired in any synchronized way.

Existing nodes cannot be assumed to systematically segregate configuration information on the basis of its source; as a result, it is quite possible that a node may receive an FQDN on one network interface, but do the DNS resolution on a different network interface, using different DNS servers. As a consequence, DNS resolution done by the DHCP server is more likely to behave predictably than DNS resolution done on a multi-interface or multi-homed client.

This is a generic DHCP protocol issue and should not be dealt within each option separately. This issue is better dealt with using a protocol-level solution and fixing this problem should not be attempted on a per option basis. Work is ongoing in the IETF to provide a systematic solution to this problem.

## 13. Chartering Requirements and Advice for Responsible Area Directors

Adding a simple DHCP option is straightforward, and generally something that any working group can do, perhaps with some help from designated DHCP experts. However, when new fragment types need to be devised, this requires the attention of DHCP experts, and should not be done in a working group that doesn't have a quorum of such experts. This is true whether the new fragment type has the same structure as an existing fragment type but has different semantics, or the new format has a new structure.

Responsible Area Directors for working groups that wish to add a work item to a working group charter to define a new DHCP option should get clarity from the working group as to whether the new option will require a new fragment type or new semantics, or whether it is a simple DHCP option that fits existing definitions.

If a working group needs a new fragment type, it is preferable to see if another working group exists whose members already have sufficient expertise to evaluate the new work. If such a working group is available, the work should be chartered in that working group instead. If there is no other working group with DHCP expertise that can define the new fragment type, the responsible AD should seek help from known DHCP experts within the IETF to provide advice and frequent early review as the original working group defines the new fragment type.

In either case, the new option should be defined in a separate document, and the work should focus on defining a new format that generalizes well and can be reused, rather than a single-use fragment type. The working group that needs the new fragment type can define their new option referencing the new fragment type document, and the work can generally be done in parallel, avoiding unnecessary delays. Having the definition in its own document will foster reuse of the new fragment type.

The responsible AD should work with all relevant working group chairs and DHCP experts to ensure that the new fragment type document has in fact been carefully reviewed by the experts and appears satisfactory.

Responsible area directors for working groups that are considering defining options that actually update the DHCP protocol, as opposed to simple options, should go through a process similar to that described above when trying to determine where to do the work. Under no circumstances should a working group be given a charter deliverable to define a new DHCP option, and then on the basis of that charter item actually make updates to the DHCP protocol.

#### 14. Considerations for Creating New Formats

When defining new options, one specific consideration to evaluate is whether or not options of a similar format would need to have multiple or single values encoded (whatever differs from the current option), and how that might be accomplished in a similar format.

When defining a new option, it is best to synthesize the option format using fragment types already in use. However, in some cases there may be no fragment type that accomplishes the intended purpose.

The matter of size considerations and option order are further discussed in Section 15 and Section 17.

## 15. Option Size

DHCPv6 [RFC3315] allows for packet sizes up to 64KB. First, through its use of link-local addresses, it avoids many of the deployment problems that plague DHCPv4, and is actually an UDP over IPv6 based protocol (compared to DHCPv4, which is mostly UDP over IPv4 protocol, but with layer 2 hacks). Second, RFC 3315 explicitly refers readers to RFC 2460 Section 5, which describes an MTU of 1280 octets and a minimum fragment reassembly of 1500 octets. It's feasible to suggest that DHCPv6 is capable of having larger options deployed over it, and at least no common upper limit is yet known to have been encoded by its implementors. It is not really possible to describe a fixed limit that cleanly divides workable option sizes from those that are too big.

It is advantageous to prefer option formats which contain the desired information in the smallest form factor that satisfies the requirements. Common sense still applies here. It is better to split distinct values into separate octets rather than propose overly complex bit shifting operations to save several bits (or even an octet or two) that would be padded to the next octet boundary anyway.

DHCPv6 does allow for multiple instances of a given option, and they are treated as distinct values following the defined format, however this feature is generally preferred to be restricted to protocol class features (such as the IA\_\* series of options). In such cases, it is better to define an option as an array if it is possible. It is recommended to clarify (with normative language) whether a given DHCPv6 option may appear once or multiple times. The default assumption is only once.

In general, if a lot of data needs to be configured (for example, some option lengths are quite large), DHCPv6 may not be the best choice to deliver such configuration information and SHOULD simply be used to deliver a URI that specifies where to obtain the actual configuration information.

## 16. Singleton options

Although [RFC3315] states that each option type MAY appear more than once, the original idea was that multiple instances are reserved for stateful options, like IA\_NA or IA\_PD. For most other options it is usually expected that they will appear at most once. Such options are called singleton options. Sadly, RFCs have often failed to

clearly specify whether a given option can appear more than once or not.

Documents that define new options SHOULD state whether these options are singletons or not. Unless otherwise specified, newly defined options are considered to be singletons. If multiple instances are allowed, the document MUST explain how to use them. Care should be taken to not assume they will be processed in the order they appear in the message. See Section 17 for more details.

When deciding whether a single or multiple option instances are allowed in a message, take into consideration how the content of the option will be used. Depending on the service being configured it may or may not make sense to have multiple values configured. If multiple values make sense, it is better to explicitly allow that by using option format that allows multiple values within one option instance.

Allowing multiple option instances often leads to confusion. Consider the following example. Basic DS-Lite architecture assumes that the B4 element (DHCPv6 client) will receive AFTR option and establish a single tunnel to configured tunnel termination point (AFTR). During standardization process of [RFC6334] there was a discussion whether multiple instances of DS-Lite tunnel option should be allowed. This created an unfounded expectation that the clients receiving multiple instances of the option will somehow know when one tunnel endpoint goes off-line and do some sort of failover between other values provided in other instances of the AFTR option. Others assumed that if there are multiple options, the client will somehow do a load balancing between provided tunnel endpoints. Neither failover nor load balancing was defined for DS-Lite architecture, so it caused confusion. It was eventually decided to allow only one instance of the AFTR option.

## 17. Option Order

Option order, either the order among many DHCPv6 options or the order of multiple instances of the same option, SHOULD NOT be significant. New documents MUST NOT assume any specific option processing order.

As there is no explicit order for multiple instances of the same option, an option definition SHOULD instead restrict ordering by using a single option that contains ordered fields.

As [RFC3315] does not impose option order, some implementations use hash tables to store received options (which is a conformant behavior). Depending on the hash implementation, the processing

order is almost always different then the order in which options appeared in the packet on wire.

#### 18. Relay Options

In DHCPv4, all relay options are organized as sub-options within DHCP Relay Agent Information Option[RFC3046]. And an independent number space called "DHCP Relay Agent Sub-options" is maintained by IANA. Different from DHCPv4, in DHCPv6, Relay options are defined in the same way as client/server options, and they too use the same option number space as client/server options. Future DHCPv6 Relay options MUST be allocated from this single DHCPv6 Option number space.

E.g. the Relay-Supplied Options Option [RFC6422] may also contain some DHCPv6 options as permitted, such as the EAP Re-authentication Protocol (ERP) Local Domain Name DHCPv6 Option [RFC6440].

#### 19. Clients Request their Options

The DHCPv6 Option Request Option (OPTION\_ORO) [RFC3315], is an option that serves two purposes - to inform the server what options the client supports and to inform what options the client is willing to consume.

For some options, such as the options required for the functioning of the DHCPv6 protocol itself, it doesn't make sense to require that they be explicitly requested using the Option Request Option. In all other cases, it is prudent to assume that any new option must be present on the relevant option request list if the client desires to receive it.

It is tempting to add text that requires the client to include a new option in Option Request Option list, similar to this text: "Clients MUST place the foo option code on the Option Request Option list, clients MAY include option foo in their packets as hints for the server as values the desire, and servers MUST include option foo when the client requested it (and the server has been so configured)". Such text is discouraged as there are several issues with it. First, it assumes that client implementation that supports a given option will always want to use it. This is not true. The second and more important reason is that such text essentially duplicates mechanism already defined in [RFC3315]. It is better to simply refer to the existing mechanism rather than define it again. See Section 21 for proposed examples on how to do that.

Creators of DHCPv6 options cannot not assume special ordering of options either as they appear in the option request option, or as they appear within the packet. Although it is reasonable to expect

that options will be processed in the order they appear in ORO, server software is not required to sort DHCPv6 options into the same order in reply messages.

It should also be noted that options values are never aligned within the DHCP packet, even the option code and option length may appear on odd byte boundaries.

## 20. Transition Technologies

Transition from IPv4 to IPv6 is progressing. Many transition technologies are proposed to speed it up. As a natural consequence there are also DHCP options proposed to provision those proposals. The inevitable question is whether the required parameters should be delivered over DHCPv4 or DHCPv6. Authors often don't give much thought about it and simply pick DHCPv6 without realizing the consequences. IPv6 is expected to stay with us for many decades, and so is DHCPv6. There is no mechanism available to deprecate an option in DHCPv6, so any options defined will stay with us as long as DHCPv6 protocol itself. It seems likely that such options defined to transition from IPv4 will outlive IPv4 by many decades. From that perspective it is better to implement provisioning of the transition technologies in DHCPv4, which will be obsoleted together with IPv4.

When the network infrastructure becomes IPv6-only, the support for IPv4-only nodes may still be needed. In such a scenario, a mechanism for providing IPv4 configuration information over IPv6-only networks such as [I-D.ietf-dhc-v4configuration] may be needed.

## 21. Recommended sections in the new document

There are three major entities in DHCPv6 protocol: server, relay agent, and client. It is very helpful for implementers to include separate sections that describe operation for those three major entities. Even when a given entity does not participate, it is useful to have a very short section stating that it must not send a given option and must ignore it when received.

There is also a separate entity called requestor, which is a special client-like type that participates in leasequery protocol [RFC5007] and [RFC5460]. A similar section for the requestor is not required, unless the new option has anything to do with requestor (or it is likely that the reader may think that is has). It should be noted that while in the majority of deployments, requestor is co-located with relay agent, those are two separate entities from the protocol perspective and they may be used separately. There are stand-alone requestor implementations available.

The following sections include proposed text for such sections. That text is not required to appear, but it is appropriate in most cases. Additional or modified text specific to a given option is often required.

Although requestor is somewhat uncommon functionality, its existence should be noted, especially when allowing or disallowing options to appear in certain message or being sent by certain entities. Additional message types may appear in the future, besides types defined in [RFC3315]. Therefore authors are encouraged to familiarize themselves with a list of currently defined DHCPv6 messages available on IANA website [iana].

Typically new options are requested by clients and assigned by the server, so there is no specific relay behavior. Nevertheless it is good to include a section for relay agent behavior and simply state that there are no additional requirements for relays. The same applies for client behavior if the options are to be exchanged between relay and server.

Sections that contain option definitions MUST include formal verification procedure. Often it is very simple, e.g. option that conveys IPv6 address must be exactly 16 bytes long, but sometimes the rules are more complex. It is recommended to refer to existing documents (e.g. section 8 of RFC3315 for domain name encoding) rather than trying to repeat such rules.

#### 21.1. DHCPv6 Client Behavior Text

Clients MAY request option foo, as defined in [RFC3315], sections 17.1.1, 18.1.1, 18.1.3, 18.1.4, 18.1.5 and 22.7. As a convenience to the reader, we mention here that the client includes requested option codes in Option Request Option.

Optional text (if client's hints make sense): Client also MAY include option foo in its SOLICIT, REQUEST, RENEW, REBIND and INFORMATION-REQUEST messages as a hint for the server regarding preferred option values.

Optional text (if the option contains FQDN): If the client requests an option that conveys an FQDN, it is expected that the contents of that option will be resolved using DNS. Hence the following text may be useful: Clients that request option foo SHOULD also request option OPTION\_DNS\_SERVERS specified in [RFC3646].

Clients MUST discard option foo if it is invalid (i.e. did not pass validation steps defined in Section X.Y).



Optional text (if option foo is expected to be exchanged between relays and servers): Option foo is exchanged between relays and servers only. Clients are not aware of the usage of option foo. Clients MUST ignore received option foo.

#### 21.2. DHCPv6 Server Behavior Text

Sections 17.2.2 and 18.2 of [RFC3315] govern server operation in regards to option assignment. As a convenience to the reader, we mention here that the server will send option foo only if configured with specific values for foo and the client requested it.

Optional text: Option foo is a singleton. Servers MUST NOT send more than one instance of foo option.

Optional text (if server is never supposed to receive option foo): Servers MUST ignore incoming foo option.

#### 21.3. DHCPv6 Relay Agent Behavior Text

It's never appropriate for a relay agent to add options to a message heading toward the client, and relay agents don't actually construct Relay-Reply messages anyway.

Optional text (if foo option is exchanged between clients and server or between requestors and servers): There are no additional requirements for relays.

Optional text (if relays are expected to insert or consume option foo): Relay agents MAY include option foo in a Relay-Forw when forwarding packets from clients to the servers.

#### 22. Should the new document update existing RFCs?

Authors often ask themselves a question whether their proposal updates exist RFCs, especially 3315. In April 2013 there were about 80 options defined. Had all documents that defined them also updated RFC3315, comprehension of such a document set would be extremely difficult. It should be noted that "extends" and "updates" are two very different verbs. If a new draft defines a new option that clients request and servers provide, it merely extends current standards, so "updates 3315" is not required in the new document header. On the other hand, if a new document replaces or modifies existing behavior, includes clarifications or other corrections, it should be noted that it updates the other document. For example, [RFC6644] clearly updates [RFC3315] as it replaces existing with new text.

If in doubt, authors should try to answer a question whether implementor reading the base RFC alone (without reading the new draft) would be able to properly implement the software. If the base RFC is sufficient, that the new draft most probably does not update the base RFC. On the other hand, if reading your draft is necessary to properly implement the base RFC, then the new draft most likely updates the base RFC.

## 23. Security Considerations

DHCPv6 does have an Authentication mechanism ([RFC3315]) that makes it possible for DHCPv6 software to discriminate between authentic endpoints and man-in-the-middle. Other authentication mechanisms may optionally be deployed. Sadly, as of late 2013, the authentication in DHCPv6 is rarely used and support for it is not common in existing implementations. Some specific deployment types make it mandatory (or parts of thereof, e.g. DOCSIS3.0 compatible cable modems require reconfigure-key support), so in certain cases specific authentication aspects can be relied upon. That is not true in the generic case, though.

So, while creating a new option, it is prudent to assume that the DHCPv6 packet contents are always transmitted in the clear, and actual production use of the software will probably be vulnerable at least to man-in-the-middle attacks from within the network, even where the network itself is protected from external attacks by firewalls. In particular, some DHCPv6 message exchanges are transmitted to multicast addresses that are likely broadcast anyway.

If an option is of a specific fixed length, it is useful to remind the implementer of the option data's full length. This is easily done by declaring the specific value of the 'length' tag of the option. This helps to gently remind implementers to validate option length before digesting them into likewise fixed length regions of memory or stack.

If an option may be of variable size (such as having indeterminate length fields, such as domain names or text strings), it is advisable to explicitly remind the implementor to be aware of the potential for long options. Either define a reasonable upper limit (and suggest validating it), or explicitly remind the implementor that an option may be exceptionally long (to be prepared to handle errors rather than truncate values).

For some option contents, out of bound values may be used to breach security. An IP address field might be made to carry a loopback address, or local multicast address, and depending on the protocol this may lead to undesirable results. A domain name field may be

filled with contrived contents that exceed the limitations placed upon domain name formatting - as this value is possibly delivered to "internal configuration" records of the system, it may be implicitly trusted without being validated.

Authors of drafts defining new DHCP options are therefore strongly advised to explicitly define validation measures that recipients of such options are required to do before processing such options. However, validation measures already defined by RFC3315 or other specifications referenced by the new option document are redundant, and can introduce errors, so authors are equally strongly advised to refer to the base specification for any such validation language rather than copying it into the new specification.

Also see Section 24.

#### 24. Privacy considerations

As discussed in Section 23 the DHCPv6 packets are typically transmitted in the clear, so they are susceptible to eavesdropping. This should be considered when defining options that may convey personally identifying information (PII) or any other type of sensitive data.

If the transmission of sensitive or confidential content is required, it is still possible to secure communication between relay agents and servers. Relay agents and servers communicating with relay agents must support the use of IPsec Encapsulating Security Payload (ESP) with encryption in transport mode, according to Section 3.1.1 of [RFC4303] and Section 21.1 of [RFC3315]. Sadly, this requirement is almost universally ignored in real deployments. Even if the communication path between relay agents and server is secured, the path between clients and relay agents or server is not.

Unless underlying transmission technology provides a secure transmission channel, the DHCPv6 options SHOULD NOT include PII or other sensitive information. If there are special circumstances that warrant sending such information over unsecured DHCPv6, the dangers MUST be clearly discussed in security considerations.

#### 25. IANA Considerations

This document has no actions for IANA.

## 26. Acknowledgements

Authors would like to thank Simon Perreault, Bernie Volz, Ted Lemon, Bud Millwood, Ralph Droms, Barry Leiba, Benoit Claise, Brian Haberman, Richard Barnes, Stephen Farrell and Steward Bryant for their comments.

## 27. References

### 27.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

### 27.2. Informative References

- [I-D.ietf-dhc-v4configuration]  
Rajtar, B. and I. Farrer, "Provisioning IPv4 Configuration Over IPv6 Only Networks", draft-ietf-dhc-v4configuration-03 (work in progress), December 2013.
- [I-D.ietf-softwire-4rd]  
Despres, R., Jiang, S., Penno, R., Lee, Y., Chen, G., and M. Chen, "IPv4 Residual Deployment via IPv6 - a Stateless Solution (4rd)", draft-ietf-softwire-4rd-07 (work in progress), October 2013.
- [I-D.ietf-softwire-map-dhcp]  
Mrugalski, T., Troan, O., Dec, W., Bao, C., leaf.yeh.sdo@gmail.com, l., and X. Deng, "DHCPv6 Options for configuration of Softwire Address and Port Mapped Clients", draft-ietf-softwire-map-dhcp-06 (work in progress), November 2013.
- [RFC3046] Patrick, M., "DHCP Relay Agent Information Option", RFC 3046, January 2001.
- [RFC3319] Schulzrinne, H. and B. Volz, "Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers", RFC 3319, July 2003.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.
- [RFC3646] Droms, R., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, December 2003.
- [RFC3898] Kalusivalingam, V., "Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3898, October 2004.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.
- [RFC4075] Kalusivalingam, V., "Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6", RFC 4075, May 2005.
- [RFC4085] Plonka, D., "Embedding Globally-Routable Internet Addresses Considered Harmful", BCP 105, RFC 4085, June 2005.
- [RFC4242] Venaas, S., Chown, T., and B. Volz, "Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 4242, November 2005.
- [RFC4280] Chowdhury, K., Yegani, P., and L. Madour, "Dynamic Host Configuration Protocol (DHCP) Options for Broadcast and Multicast Control Servers", RFC 4280, November 2005.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.
- [RFC4436] Aboba, B., Carlson, J., and S. Cheshire, "Detecting Network Attachment in IPv4 (DNav4)", RFC 4436, March 2006.
- [RFC4704] Volz, B., "The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option", RFC 4704, October 2006.
- [RFC4833] Lear, E. and P. Eggert, "Timezone Options for DHCP", RFC 4833, April 2007.
- [RFC4957] Krishnan, S., Montavont, N., Njedjou, E., Veerepalli, S., and A. Yegin, "Link-Layer Event Notifications for Detecting Network Attachments", RFC 4957, August 2007.

- [RFC5007] Brzozowski, J., Kinnear, K., Volz, B., and S. Zeng, "DHCPv6 Leasequery", RFC 5007, September 2007.
- [RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, March 2008.
- [RFC5223] Schulzrinne, H., Polk, J., and H. Tschofenig, "Discovering Location-to-Service Translation (LoST) Servers Using the Dynamic Host Configuration Protocol (DHCP)", RFC 5223, August 2008.
- [RFC5460] Stapp, M., "DHCPv6 Bulk Leasequery", RFC 5460, February 2009.
- [RFC5908] Gayraud, R. and B. Lourdelet, "Network Time Protocol (NTP) Server Option for DHCPv6", RFC 5908, June 2010.
- [RFC5970] Huth, T., Freimann, J., Zimmer, V., and D. Thaler, "DHCPv6 Options for Network Boot", RFC 5970, September 2010.
- [RFC5986] Thomson, M. and J. Winterbottom, "Discovering the Local Location Information Server (LIS)", RFC 5986, September 2010.
- [RFC6059] Krishnan, S. and G. Daley, "Simple Procedures for Detecting Network Attachment in IPv6", RFC 6059, November 2010.
- [RFC6334] Hankins, D. and T. Mrugalski, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Option for Dual-Stack Lite", RFC 6334, August 2011.
- [RFC6422] Lemon, T. and Q. Wu, "Relay-Supplied DHCP Options", RFC 6422, December 2011.
- [RFC6440] Zorn, G., Wu, Q., and Y. Wang, "The EAP Re-authentication Protocol (ERP) Local Domain Name DHCPv6 Option", RFC 6440, December 2011.
- [RFC6603] Korhonen, J., Savolainen, T., Krishnan, S., and O. Troan, "Prefix Exclude Option for DHCPv6-based Prefix Delegation", RFC 6603, May 2012.
- [RFC6610] Jang, H., Yegin, A., Chowdhury, K., Choi, J., and T. Lemon, "DHCP Options for Home Information Discovery in Mobile IPv6 (MIPv6)", RFC 6610, May 2012.

[RFC6644] Evans, D., Droms, R., and S. Jiang, "Rebind Capability in DHCPv6 Reconfigure Messages", RFC 6644, July 2012.

[iana] IANA, , "DHCPv6 parameters (IANA webpage)", November 2003, <<http://www.iana.org/assignments/dhcpv6-parameters/>>.

#### Authors' Addresses

David W. Hankins  
Google, Inc.  
1600 Amphitheatre Parkway  
Mountain View, CA 94043  
USA

Email: [dhankins@google.com](mailto:dhankins@google.com)

Tomek Mrugalski  
Internet Systems Consortium, Inc.  
950 Charter Street  
Redwood City, CA 94063  
USA

Phone: +1 650 423 1345  
Email: [tomasz.mrugalski@gmail.com](mailto:tomasz.mrugalski@gmail.com)

Marcin Siodelski  
950 Charter Street  
Redwood City, CA 94063  
USA

Phone: +1 650 423 1431  
Email: [msiodelski@gmail.com](mailto:msiodelski@gmail.com)

Sheng Jiang  
Huawei Technologies Co., Ltd  
Q14, Huawei Campus, No.156 Beiqing Road  
Hai-Dian District, Beijing, 100095  
P.R. China

Email: [jiangsheng@huawei.com](mailto:jiangsheng@huawei.com)

Suresh Krishnan  
Ericsson  
8400 Blvd Decarie  
Town of Mount Royal, Quebec  
Canada

Email: suresh.krishnan@ericsson.com



DHC WG  
Internet-Draft  
Intended status: Informational  
Expires: January 2, 2015

B. Rajtar  
Hrvatski Telekom  
I. Farrer  
Deutsche Telekom AG  
July 01, 2014

Provisioning IPv4 Configuration Over IPv6 Only Networks  
draft-ietf-dhc-v4configuration-06

Abstract

As IPv6 becomes more widely adopted, some service providers are choosing to deploy IPv6 only networks without dual-stack functionality for IPv4. However, as access to IPv4 based services will continue to be a requirement for the foreseeable future, IPv4 over IPv6 mechanisms, such as softwire tunnels are being developed.

In order to provision end-user's hosts with the IPv4 configuration necessary for such mechanisms, a number of different approaches have been proposed. This memo discusses each of the proposals, identifies the benefits and drawbacks and recommends approaches to be used as the basis for future deployment and development.

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 2, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

## Table of Contents

1. Introduction . . . . .	3
1.1. Overview of IPv4 Parameter Configuration Approaches . . . .	4
1.2. DHCPv4o6 Based Provisioning - Functional Overview . . . .	4
1.3. DHCPv6 Based Provisioning - Functional Overview . . . . .	6
1.4. DHCPv6 + Stateless DHCPv4oSW Based Provisioning - Functional Overview . . . . .	6
1.5. DHCPv4oDHCPv6 Based Provisioning - Functional Overview .	7
2. Requirements for the Solution Evaluation . . . . .	8
3. Comparison of the Four Approaches . . . . .	9
3.1. DHCPv4o6 Based Provisioning . . . . .	9
3.1.1. Pros . . . . .	9
3.1.2. Cons . . . . .	10
3.2. DHCPv6 Based Provisioning . . . . .	10
3.2.1. Pros . . . . .	10
3.2.2. Cons . . . . .	10
3.3. DHCPv6 + Stateless DHCPv4oSW Based Provisioning . . . . .	11
3.3.1. Pros . . . . .	11
3.3.2. Cons . . . . .	11
3.4. DHCPv4oDHCPv6 Based Provisioning . . . . .	12
3.4.1. Pros . . . . .	12
3.4.2. Cons . . . . .	12
4. Conclusion . . . . .	13
5. Transporting Unmodified DHCPv4 Messages over an IPv6 Link Layer . . . . .	13
5.1. Combined Hub and DHCPv4 Relay Required Functionality . .	14
6. IANA Considerations . . . . .	14
7. Security Considerations . . . . .	15
7.1. DHCPv4oIPv6 . . . . .	15
7.2. DHCPv6 . . . . .	15
7.3. DHCPv6+DHCPv4oSW . . . . .	15
7.4. DHCPv4oDHCPv6 . . . . .	15
8. Acknowledgements . . . . .	15
9. Informative References . . . . .	15
Authors' Addresses . . . . .	17

## 1. Introduction

A service provider with an IPv6-only network must also be able to provide customers with access to the IPv4 Internet and other IPv4-only services. IPv4 over IPv6 tunneling / translation mechanisms are an obvious example of this, such as the ones described in:

- o [I-D.ietf-softwire-lw4over6]
- o [I-D.ietf-softwire-map]
- o [I-D.ietf-softwire-map-t]

In today's home networks, each residential user is allocated a single global IPv4 address which is used for NAT44. Decentralizing NAT44 allows for much better scaling and, when combined with stateless network functions, can simplify redundancy and logging when compared to centralized Carrier Grade NAT architectures. This results in the need to provision a number of configuration parameters to the CPE, such as the external public IPv4 address and a restricted port-range to use for NAT. Other parameters may also be necessary, depending on the underlying transport technology that is in use. In IPv4 only networks, DHCPv4 has often been used to provide IPv4 configuration, but in an IPv6 only network, DHCPv4 messages cannot be transported natively without either IPv6 encapsulation or translation.

DHCPv4 messages can be transported, unmodified, over a broadcast capable link-layer, depending on the underlying IPv4 in IPv6 technology, network topology and DHCPv4 client capabilities. A functional description of how unmodified DHCPv4 can be used is provided in Section 5. This approach is recommended for service providers whose network and clients can support this DHCPv4 architecture.

For the most simple IPv4 provisioning case, where the client only needs to receive a static IPv4 address assignment (with no dynamic address leasing or additional IPv4 configuration), a DHCPv6 based approach (e.g. [I-D.ietf-softwire-map-dhcp]) may provide a suitable solution.

This document is concerned with more complex IPv4 configuration scenarios, to bring IPv4 configuration over IPv6-only networks in line with the functionality offered by DHCPv4 in IPv4 native networks. DHCPv4 options may also need to be conveyed to clients for configuring IPv4 based services, e.g., SIP server addresses.

Although IPv4-in-IPv6 software tunnel and translation clients are currently the only use-case for DHCP based configuration of IPv4 parameters in IPv6 only networks, a suitable IPv4 provisioning solution should not be limited to only supporting the configuration of softwires, or be bound to specific IPv4 over IPv6 architectures or mechanisms. The solution needs to be flexible enough to support new IPv4 over IPv6 technologies as they are developed.

This document describes and compares four different methods which have been proposed as solutions to this problem.

### 1.1. Overview of IPv4 Parameter Configuration Approaches

The following approaches for transporting IPv4 configuration parameters over IPv6 only networks have been suggested:

1. Adapt DHCPv4 format messages to be transported over IPv6 as described in [I-D.ietf-dhc-dhcpv4-over-ipv6]. For brevity, this is referred to as DHCPv4o6.
2. Extend DHCPv6 to support IPv4 address leasing and other DHCPv4 options.
3. Use DHCPv6 for external IPv4 address and source port configuration (e.g. [I-D.ietf-softwire-map-dhcp]). Use DHCPv4 over IPv4 messages within an IPv6 software for configuring additional parameters. This is referred to as DHCPv6 + Stateless DHCPv4oSW.
4. Use DHCPv4 format messages, transporting them within a new DHCPv6 message type as described in [I-D.ietf-dhc-dhcpv4-over-dhcpv6]. This is referred to as DHCPv4oDHCPv6.

At the time of writing, working examples of all but the third method have been developed and successfully tested in several different operators networks.

The following sections describe each of the approaches in more detail.

### 1.2. DHCPv4o6 Based Provisioning - Functional Overview

In order to receive IPv4 configuration parameters, IPv4-only clients initiate and exchange DHCPv4 messages with the DHCPv4 server. To adapt this for an IPv6-only network, an existing DHCPv4 client implements a Host Client Relay Agent (HCRA) function, which takes DHCPv4 messages and puts them into UDP and IPv6.

As the mechanism involves unicast IPv6 based communications, the IPv6 address of the server must be provisioned to the client. A DHCPv6 option for provisioning clients with this address is described in [I-D.mrugalski-softwire-dhcpv4-over-v6-option].

The IPv6 Transport Server (TSV) provides an IPv6 interface to the client. This interface may be implemented directly on the server and/or via an intermediary 'Transport Relay Agent' (TRA) device which acts as the gateway between the IPv4 and IPv6 domains.

For the dynamic allocation of IPv4 addresses, the DHCPv4 server function needs to be extended to add DHCPv4o6 TSV capabilities, such as the storing the IPv6 address of DHCPv4o6 clients and implementing the CRA6ADDR option.

This approach currently uses functional elements for ingress and egress of the IPv6-only transport domain - the HCRA on the host and the TRA or TSV on the server. As a result, this has sometimes been referred to as a tunneling approach. However, relay agent encapsulation is not a tunnel, since it carries only DHCP traffic; it would be more accurate to describe it as an encapsulation based transport.

[I-D.ietf-dhc-dhcpv4-over-ipv6] also defines an On-Link Client Relay Agent (LCRA), which is a Client Relay Agent located on the same link as an unmodified DHCPv4 client. It is worth noting that there is no technical reason for using relay encapsulation for DHCPv4o6; this approach was taken because the authors of the draft originally imagined that it might be used to provide configuration information for an unmodified DHCPv4 client. However, this turns out not to be a viable approach: in order for this to work, there would have to be IPv4 routing on the local link to which the client is connected. In that case, there's no need for DHCPv4o6.

Given that this is the case, there is no technical reason why DHCPv4o6 can't simply use the IPv6 transport directly, without any relay encapsulation. This would greatly simplify the specification and the implementation, and would still address the requirements stated in this document.

[I-D.ietf-dhc-dhcpv4-over-ipv6] describes this solution in detail.

The protocol stack for provisioning IPv4/IPv6 tunneling and translation mechanisms is as follows:

DHCPv4/UDP/IPv6

### 1.3. DHCPv6 Based Provisioning - Functional Overview

In this approach, DHCPv6 [RFC3315] would be extended with new DHCPv6 options for configuring all IPv4 based services and functions (i.e. IPv4 address assignment and any necessary DHCPv4 options). DHCPv4 options needed by IPv4 clients connected to the IPv6 network are updated as new DHCPv6 native options carrying IPv4 configuration parameters. IPv4 address leasing would also need to be managed by the DHCPv6 server.

At the time of writing, it is not known which or how many such options would need to be ported from DHCPv4 to DHCPv6.

The protocol stack for provisioning IPv4/IPv6 tunneling and translation mechanisms is as follows:

DHCPv6/UDP/IPv6

### 1.4. DHCPv6 + Stateless DHCPv4oSW Based Provisioning - Functional Overview

In this approach, configuration of the IPv4 address and source ports (if required) is carried out using DHCPv6, e.g. using [I-D.ietf-software-map-dhcp]. Any additional IPv4 configuration parameters that are required are then provisioned using DHCPv4 messages transported, within IPv6, through the configured software in the same manner as any other IPv4 based traffic. Broadcast based DHCPv4 DHCPDISCOVER messages (necessary for IPv4 address assignment) can not be transported as some software mechanisms implement NBMA links, where broadcast isn't supported. Additionally, there is a more general issue with the use of fixed L4 ports in A+P [RFC6346] based approaches. Here, a single IPv4 address is shared among multiple users, each using a unique set of ports for differentiation meaning that it is not possible for every client to be allocated a fixed L4 within its unique port set.

On receipt by the tunnel concentrator (e.g. MAP Border Router or a Lightweight 4over6 lwAFTR), the DHCPv4 message is extracted from the IPv6 packet and forwarded to the DHCPv4 server in the same way as any other IPv4 forwarding plane packet is handled.

As the client is already configured with its external IPv4 address and source ports (using DHCPv6 or a well-known IPv4 address for DS-Lite clients), the messages exchanged between the DHCPv4 client and server would be strictly DHCPINFORM/DHCPACK messages. These can be used for conveying additional DHCPv4 based options.

For this approach to function, a mechanism for the DHCPv4 client to learn the IPv4 address of the DHCPv4 server is also required. This could be via a well-known IPv4 address for the DHCPv4 server, a DHCPv4 relay function within the tunnel concentrator or other methods.

From a transport perspective, the key difference between this method and DHCPv4o6 (described above) is the protocol stack. Here the DHCPv4 message is first put into UDP and IPv4 and then into the IPv6 software, instead of placing the DHCPv4 message directly into UDP and IPv6.

Currently, this approach is only theoretical and does not have a corresponding Internet Draft providing more detail.

For IPv4/IPv6 tunneling and translation mechanism, the protocol stack used for obtaining an IPv4 address and source ports (if required) is as follows:

DHCPv6/UDP/IPv6

For provisioning IPv4/IPv6 tunneling mechanisms, the protocol stack for obtaining additional IPv4 configuration is:

DHCPv4/UDP/IPv4

NB: The encapsulating IPv6 tunneling header is not shown as it is functionally a layer 2 header.

And for provisioning IPv4/IPv6 translation mechanisms:

DHCPv4/UDP/IPv6

#### 1.5. DHCPv4oDHCPv6 Based Provisioning - Functional Overview

[I-D.ietf-dhc-dhcpv4-over-dhcpv6] describes transporting DHCPv4 messages within two new DHCPv6 messages types: DHCPV4-QUERY and DHCPV4-RESPONSE. These new messages types must be implemented in both the DHCPv4oDHCPv6 client and server.

In this approach, dynamic IPv4 addressing, and/or any additional IPv4 configuration, is provided using DHCPv4 messages carried (without IPv4/UDP headers) within a new OPTION\_DHCPV4\_MSG DHCPv6 option.

OPTION\_DHCPV4\_MSG enables the client and server to send BOOTP/DHCPv4 messages verbatim across the IPv6 network. When a DHCPv4oDHCPv6 server receives a DHCPv6 request containing OPTION\_DHCPV4\_MSG within a DHCPV4-QUERY message, it passes it to the DHCPv4 server engine.

Likewise, the DHCPv4 server place its DHCPv4 response in the payload of OPTION\_DHCPV4\_MSG and puts this into a DHCPV4-RESPONSE message.

DHCPv4 messages can be carried within DHCPv6 multicast messages, using the All\_DHCP\_Relay\_Agents\_and\_Servers multicast address. These can be relayed in exactly the same way as any other DHCPv6 multicasted messages.

Optionally, DHCPv6 relays could be updated so that they forward the DHCPV4-QUERY message to a different destination address, allowing for the separation of DHCPv4 and DHCPv6 provisioning infrastructure.

If the DHCPv4/DHCPv6 client is provisioned with a unicast IPv6 address(es) for the server(s), then an entirely unicast message flow between the client and server is also possible without the need for relaying.

For provisioning IPv4/IPv6 tunneling and translation mechanisms, the protocol stack used for obtaining dynamic v4 addressing and/or additional IPv4 configuration is as follows:

DHCPv4/DHCPv6/UDP/IPv6

## 2. Requirements for the Solution Evaluation

The following requirements have been defined to evaluate the different approaches:

1. Minimize the amount of work necessary to implement the solution through re-use of existing standards and implementations as much as possible.
2. Provide a method of supporting all DHCPv4 options so that they can be utilized without the need for further standardization.
3. Allow for the dynamic leasing of IPv4 addresses to clients. This allows for more efficient use of limited IPv4 resources.
4. Enable the separation of IPv4 and IPv6 host configuration infrastructure, i.e. independent DHCPv4 and DHCPv6 server functions to restrict provisioning domains to the relevant protocol and allow the removal of IPv4 infrastructure in the future.
5. Avoid leaving legacy IPv4 options in DHCPv6.



6. Provide a flexible architecture to give operators the option of only deploying the functional elements necessary for their specific requirements.
7. Not be restricted to specific underlying IPv4 over IPv6 transport mechanisms or architectures. The solution needs to be flexible enough to support new IPv4 over IPv6 technologies as they are developed.

### 3. Comparison of the Four Approaches

The table below provides a comparative evaluation showing how the different approaches meet the solution requirements described above.

Req. No.	DHCPv4o6	DHCPv6	DHCPv6 + Stateless DHCPv4oSW	DHCPv4oDHCPv6
1	No	Yes	No	Yes
2	Yes	No	Yes	Yes
3	Yes	No	No	Yes
4	Yes	No	Yes	Yes
5	Yes	No	Yes	Yes
6	No	No	Yes	Yes
7	Yes	Yes	No	Yes

Table 1: Approach Comparison

The following sections of the document provide more detail on the pros and cons of each of the approaches.

#### 3.1. DHCPv4o6 Based Provisioning

##### 3.1.1. Pros

1. Implementation makes all existing DHCPv4 options available with no further ongoing development work necessary.
2. IPv4 and IPv6 based provisioning can be separated from each other if required, allowing flexibility in network design.
3. Easy to implement through minor adaptation of existing DHCPv4 client, relay and server code.
4. Suitable for dynamic IPv4 address leases where the IPv4 address lifetime is not linked to the lifetime of a DHCPv6 lease.

5. Implementations already exist, proving that the approach works.

#### 3.1.2. Cons

1. More new functional elements required within the architecture (CRA, DHCPv4o6 server and optionally TRA) than are necessary in DHCPv6 based provisioning.
2. A new DHCPv6 option is necessary in order to provision the IPv6 address of the DHCPv4 server to the end device.
3. The DHCPv4 client host needs to be updated to implement the IPv6 encapsulation and decapsulation function (i.e., an HCRA). Otherwise a separate On-Link CRA (LCRA) functional element must be deployed.
4. A DHCPv4 server must be deployed and maintained.
5. The DHCPv4 server needs to be updated to implement new DHCPv4o6 functionality.

### 3.2. DHCPv6 Based Provisioning

#### 3.2.1. Pros

1. No additional functional elements are required except the DHCPv6 client and server.
2. A single protocol is used to deliver configuration information for IPv4 and IPv6.
3. Single provisioning point for all configuration parameters.

#### 3.2.2. Cons

1. Any required DHCPv4 options must be ported to DHCPv6, which will require re-development work for each option.
2. Means that DHCPv4 'legacy' options (which will be of decreasing relevance in the future) will remain in DHCPv6 for the lifetime of the protocol.
3. Each time that a DHCPv4 option is ported to DHCPv6, all clients, servers and possibly relays would need to be updated to implement the new option.
4. Architecture does not allow for the separation of IPv4 and IPv6 domains.

5. Does not provide a mechanism for dynamic IPv4 address leasing. The lifetime of the IPv4 address is linked to the lifetime of a DHCPv6 address lease (i.e. the IPv4 address can only be changed when a DHCPv6 RENEW/REBIND message is sent). To remove this interdependency, a new DHCPv4 lease management mechanism would need to be added to DHCPv6 (e.g. a new Identity Association solely for IPv4 address leasing).

### 3.3. DHCPv6 + Stateless DHCPv4oSW Based Provisioning

#### 3.3.1. Pros

1. Once implemented, all existing DHCPv4 options will be available with no ongoing development work required.
2. Uses existing DHCPv4 and DHCPv6 architectures in order to provide IPv4 configuration in an IPv6 only environment.
3. If required, DHCPv4 and DHCPv6 based provisioning can be separated from each other, allowing flexibility in network design.

#### 3.3.2. Cons

1. More new functional elements required than are necessary with DHCPv6 based provisioning.
2. IPv4 over IPv6 software approaches that distribute the NAT44 function to the CPE and allow for IP address sharing (MAP-E & LW4o6) forbid the use of reserved TCP/UDP ports (e.g. 0-1024). Every DHCPv4 client sharing the same address needs to have a UDP listener running on UDP port 68. To resolve this would require significant rework to either the software mechanisms and/or the DHCPv4 client implementation.
3. From the current specification, DHCPINFORM is not suitable for use over a software. Additional work, such as the development of 'shims' would be necessary.
4. The current DHCPINFORM specification has a number of unclear points, such as those described in [I-D.ietf-dhc-dhcpinform-clarify]. Substantial work would be required to resolve this.
5. Links the deployment of IPv4 configuration over IPv6 to a software implementation (e.g. requiring a software concentrator to act as a DHCPv4 relay). Whilst softwares are the only

application for this functionality at the moment, this may not be the case in the future, meaning another solution may be required.

6. A new mechanism must be defined in order to provide the DHCPv4 client with the IPv4 address of the DHCPv4 server so that unicast DHCPINFORM messages can be sent.
7. As only the DHCPINFORM/DHCPACK DHCPv4 message types are supported, dynamic IPv4 address leasing (using DHCPDISCOVER messages) cannot be used.
8. Restricted to underlying hub-and-spoke IPv4 over IPv6 architectures. The hub is necessary to locate the DHCPv4 relay function, as all traffic must pass through it. An underlying mesh architecture does not have such a location to deploy the relay function.
9. The approach is currently unproven. Although existing implementations may currently exist, the approach has not been demonstrated.

#### 3.4. DHCPv4oDHCPv6 Based Provisioning

##### 3.4.1. Pros

1. Once implemented, all existing DHCPv4 options will be available with no ongoing development work necessary.
2. Uses existing DHCPv4 and DHCPv6 architectures in order to provide IPv4 configuration in an IPv6 only environment.
3. If required, DHCPv4 and DHCPv6 based provisioning can be separated from each other, allowing flexibility in network design.
4. Suitable for the provisioning of dynamic IPv4 configuration as the existing DHCPv4 leasing mechanism can be used.

##### 3.4.2. Cons

1. More new functional elements within the architecture than are necessary in DHCPv6 based provisioning.
2. DHCPv6 clients need to be updated to implement the new DHCPv6 message types (BOOTPREQUESTv6 and BOOTPREPLYv6).
3. The DHCPv6 server needs to be updated to implement the new DHCPv4oDHCPv6 message types and functionality.

4. The approach is currently unproven as no existing implementations exist.

#### 4. Conclusion

Whilst all of the approaches described here will require some development work to realize, it is clear from the above analysis that the most sustainable approach capitalizes on existing DHCPv4 implementations and include them as new DHCPv6 message types. The main rationale for this is that it enables all of DHCPv4's existing options to be migrated for use over IPv6 in a single step.

Porting of all necessary DHCPv4 options to DHCPv6 would require ongoing development work, re-implementing existing DHCPv4 functionality in DHCPv6. This will result in having legacy DHCPv4 options in DHCPv6, which will no longer be useful once IPv4 is completely abandoned.

Therefore, the DHCPv6 approach is not appropriate for delivering IPv4 configuration parameters.

The dynamic leasing of IPv4 addresses is fundamental to the efficient use of remaining IPv4 resources. This will become increasingly important in the future, so a mechanism which supports this is necessary. DHCPv6 + Stateless DHCPv4oSW does not provide this function and so is not recommended.

The DHCPv4o6 approach requires a DHCPv4 server (with DHCPv4o6 functionality) for all deployment scenarios, even when DHCPv4 specific functionality (e.g. sending DHCPv4 options) is not required by the operator.

Therefore, this memo recommends DHCPv4oDHCPv6 [I-D.ietf-dhc-dhcpv4-over-dhcpv6] as the best underlying approach for provisioning IPv4 parameters over an IPv6 only network.

#### 5. Transporting Unmodified DHCPv4 Messages over an IPv6 Link Layer

DHCPv4 can be transported across a broadcast capable link layer, such as a softwire. Functionally, a DHCPv4 client operates on the link layer interface (e.g. the softwire tunnel interface). As the link layer must support broadcasts, DHCPDISCOVER and other broadcast DHCPv4 messages can be transported. The DHCPv4 message flow is then the same as described in section 3.1 of [RFC2131].

For an unmodified DHCPv4 client to function over an IPv6 native network, the underlying IPv4 over IPv6 architecture must be based on a point-to-point link between the client and a central point (i.e. a

hub or tunnel concentrator) which all client DHCPv4 broadcast messages will pass through. This hub must function as either the DHCPv4 server or a DHCPv4 relay. The relay forwards broadcast DHCPv4 DHCPDISCOVER/DHCPREQUEST messages to a separate DHCPv4 server.

#### 5.1. Combined Hub and DHCPv4 Relay Required Functionality

When the DHCPv4 relay function is co-located with the IPv4 in IPv6 hub function, there are some implementation considerations and requirements that must be fulfilled. The following list describes these.

1. Depending on the underlying IPv4 over IPv6 mechanism that the hub is based upon, it may be necessary to modify the encapsulation/decapsulation or IPv6/IPv4 translation packet validation policy so that IPv4 payload packets sourced from the unspecified address (0.0.0.0) are not dropped for broadcast DHCPv4 payload packets.
2. The DHCPv4 relay must use the DHCPv4 Relay Information Option (option 82) Relay-ID sub-option (2) to convey the client's source IPv6 address. This is used by the relay to route DHCPv4 response packets sent by the DHCPv4 server to the correct client.
3. For some IPv4 in IPv6 transition technologies, a client may be configured with an IPv4 address which is shared by other clients. In these cases, clients using a single IPv4 address are differentiated using the combination of the IPv4 address and a range of restricted layer 4 source ports unique to each client (used for NAT). The DHCPv4 client L4 port (68) must not be provisioned to any client for NAT use.
4. The DHCPv4 relay must implement the Server Identifier Override Sub-option described in [RFC5107] to direct all DHCPv4 messages through the DHCPv4 relay. As option 82 is being used to identify the destination IPv6 address for messages from the DHCPv4 server to the client, the L4 destination port is not required for the return path lookup process and is left unchanged as port 68.

#### 6. IANA Considerations

This document does not make any request from IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

## 7. Security Considerations

This document analyzes various solutions and doesn't introduce any new capabilities necessitating additional security considerations. The following sub-sections provide pointers to the documented security considerations associated with each approach.

### 7.1. DHCPv4oIPv6

Security considerations associated with this approach are described in Section 8 of [I-D.ietf-dhc-dhcpv4-over-ipv6].

### 7.2. DHCPv6

Security considerations associated with this approach are described in Section 23 of [RFC3315].

### 7.3. DHCPv6+DHCPv4oSW

There is currently no document describing this mechanism, so no security considerations have been documented.

### 7.4. DHCPv4oDHCPv6

Security considerations associated with this approach are described in [I-D.ietf-dhc-dhcpv4-over-dhcpv6].

## 8. Acknowledgements

Thanks to Ted Lemon, Tomek Mrugalski, Ole Troan, Bernie Volz and Francis Dupont for their input and reviews.

## 9. Informative References

- [I-D.ietf-dhc-dhcpinform-clarify]  
Hankins, D., "Dynamic Host Configuration Protocol DHCPINFORM Message Clarifications", draft-ietf-dhc-dhcpinform-clarify-06 (work in progress), October 2011.
- [I-D.ietf-dhc-dhcpv4-over-dhcpv6]  
Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4 over DHCPv6 Transport", draft-ietf-dhc-dhcpv4-over-dhcpv6-09 (work in progress), June 2014.
- [I-D.ietf-dhc-dhcpv4-over-ipv6]  
Cui, Y., Wu, P., Wu, J., Lemon, T., and Q. Sun, "DHCPv4 over IPv6 Transport", draft-ietf-dhc-dhcpv4-over-ipv6-09 (work in progress), April 2014.

- [I-D.ietf-softwire-lw4over6]  
Cui, Y., Qiong, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the DS-Lite Architecture", draft-ietf-softwire-lw4over6-10 (work in progress), June 2014.
- [I-D.ietf-softwire-map]  
Troan, O., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, "Mapping of Address and Port with Encapsulation (MAP)", draft-ietf-softwire-map-10 (work in progress), January 2014.
- [I-D.ietf-softwire-map-dhcp]  
Mrugalski, T., Troan, O., Farrer, I., Perreault, S., Dec, W., Bao, C., leaf.yeh.sdo@gmail.com, l., and X. Deng, "DHCPv6 Options for configuration of Softwire Address and Port Mapped Clients", draft-ietf-softwire-map-dhcp-07 (work in progress), March 2014.
- [I-D.ietf-softwire-map-t]  
Li, X., Bao, C., Dec, W., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", draft-ietf-softwire-map-t-05 (work in progress), February 2014.
- [I-D.mrugalski-softwire-dhcpv4-over-v6-option]  
Mrugalski, T. and P. Wu, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Option for DHCPv4 over IPv6 Endpoint", draft-mrugalski-softwire-dhcpv4-over-v6-option-01 (work in progress), September 2012.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC5107] Johnson, R., Kumarasamy, J., Kinnear, K., and M. Stapp, "DHCP Server Identifier Override Suboption", RFC 5107, February 2008.
- [RFC6346] Bush, R., "The Address plus Port (A+P) Approach to the IPv4 Address Shortage", RFC 6346, August 2011.



Authors' Addresses

Branimir Rajtar  
Hrvatski Telekom  
Zagreb  
Croatia

Email: [branimir.rajtar@t.ht.hr](mailto:branimir.rajtar@t.ht.hr)

Ian Farrer  
Deutsche Telekom AG  
Bonn  
Germany

Email: [ian.farrer@telekom.de](mailto:ian.farrer@telekom.de)

DHC Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 19, 2014

S. Jiang  
Huawei Technologies Co., Ltd  
S. Shen  
CNNIC  
October 16, 2013

Secure DHCPv6 with Public Key  
draft-jiang-dhc-sedhcpv6-02

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) enables DHCPv6 servers to pass configuration parameters. It offers configuration flexibility. If not secured, DHCPv6 is vulnerable to various attacks, particularly spoofing attacks. This document analyzes the security issues of DHCPv6 and specifies a Secure DHCPv6 mechanism for communication between DHCPv6 client and server. This mechanism is based on public/private key pairs. The authority of the sender may depend on either pre-configuration mechanism or Public Key Infrastructure.

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 April 19, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Requirements Language and Terminology . . . . .	3
3. Security Overview of DHCPv6 . . . . .	3
4. Secure DHCPv6 Overview . . . . .	4
4.1. New Components . . . . .	5
4.2. Support for algorithm agility . . . . .	5
5. Extensions for Secure DHCPv6 . . . . .	5
5.1. Public Key Option . . . . .	6
5.2. Certificate Option . . . . .	6
5.3. Signature Option . . . . .	7
6. Processing Rules and Behaviors . . . . .	8
6.1. Processing Rules of Sender . . . . .	8
6.2. Processing Rules of Recipient . . . . .	9
6.3. Processing Rules of Relay Agent . . . . .	10
6.4. Timestamp Check . . . . .	10
7. Security Considerations . . . . .	12
8. IANA Considerations . . . . .	13
9. Acknowledgements . . . . .	14
10. Change log [RFC Editor: Please remove] . . . . .	14
11. References . . . . .	14
11.1. Normative References . . . . .	14
11.2. Informative References . . . . .	15
Authors' Addresses . . . . .	15

## 1. Introduction

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6, [RFC3315]) enables DHCPv6 servers to pass configuration parameters. It offers configuration flexibility. If not secured, DHCPv6 is vulnerable to various attacks, particularly spoofing attacks.

This document analyzes the security issues of DHCPv6 in details. This document provides mechanisms for improving the security of DHCPv6 between client and server:

- o the identity of a DHCPv6 message sender, which can be a DHCPv6 server or a client, can be verified by a recipient.
- o the integrity of DHCPv6 messages can be checked by the recipient of the message.

Note: this secure mechanism in this document does not protect the relay-relevant options, either added by a relay agent toward a server or added by a server toward a relay agent, are considered less vulnerable, because they are only transported within operator networks. Communication between a server and a relay agent, and communication between relay agents, may be secured through the use of IPsec, as described in section 21.1 in [RFC3315].

The security mechanisms specified in this document is based on self-generated public/private key pairs. It also integrates timestamps for anti-replay. The authentication procedure defined in this document may depend on either deployed Public Key Infrastructure (PKI, [RFC5280]) or pre-configured sender's public key. However, the deployment of PKI or pre-configuration is out of the scope.

Secure DHCPv6 is applicable in environments where physical security on the link is not assured (such as over wireless) and attacks on DHCPv6 are a concern.

## 2. Requirements Language and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] when they appear in ALL CAPS. When these words are not in ALL CAPS (such as "should" or "Should"), they have their usual English meanings, and are not to be interpreted as [RFC2119] key words.

## 3. Security Overview of DHCPv6

DHCPv6 is a client/server protocol that provides managed configuration of devices. It enables DHCPv6 server to automatically configure relevant network parameters on clients. In the basic DHCPv6 specification [RFC3315], security of DHCPv6 message can be improved.

The basic DHCPv6 specifications can optionally authenticate the origin of messages and validate the integrity of messages using an authentication option with a symmetric key pair. [RFC3315] relies on pre-established secret keys. For any kind of meaningful security, each DHCPv6 client would need to be configured with its own secret key; [RFC3315] provides no mechanism for doing this.

For the key of the hash function, there are two key management mechanisms. Firstly, the key management is done out of band, usually through some manual process. For example, operators can set up a key database for both servers and clients which the client obtains a key before running DHCPv6.

Manual key distribution runs counter to the goal of minimizing the configuration data needed at each host. [RFC3315] provides an additional mechanism for preventing off-network timing attacks using the Reconfigure message: the Reconfigure Key authentication method. However, this method provides no message integrity or source integrity check. This key is transmitted in plaintext.

In comparison, the public/private key security mechanism allows the keys to be generated by the sender, and allows the public key database on the recipient to be populated opportunistically or manually, depending on the degree of confidence desired in a specific application. PKI security mechanism is simpler in the local key management respect.

#### 4. Secure DHCPv6 Overview

To solve the above mentioned security issues, this document introduces the use of public/private key pair mechanism into DHCPv6, also with timestamp. The authority of the sender may depend on either pre-configuration mechanism or PKI. By combining with the signatures, sender identity can be verified and messages protected.

This document introduces a Secure DHCPv6 mechanism that uses a public/private key pair to secure the DHCPv6 protocol. It has two modes; in both modes, the sender has a public/private key pair. In the first mode, the public key of the sender is pre-shared with the recipient, either opportunistically or through a manual process. In the second mode, the sender has a certificate for its public key, signed by a Certificate Authority that is trusted by the recipient. It is possible for the same public key to be used with different recipients in both modes.

In this document, we introduce a public key option, a certificate option and a signature options with a corresponding verification mechanism. Timestamp is integrated into signature options. A DHCPv6 message (from a server or a client), with either a public key or certificate option, and carrying a digital signature, can be verified by the recipient for both the timestamp and authentication, then process the payload of the DHCPv6 message only if the validation is successful. Because the sender can be a DHCPv6 server or a client, the end-to-end security protection can be from DHCPv6 servers to or clients, or from clients to DHCPv6 servers.

This improves communication security of DHCPv6 messages. The authentication options [RFC3315] may also be used for replay protection.

#### 4.1. New Components

The components of the solution specified in this document are as follows:

- o The node generates a public/private key pair. A DHCPv6 option is defined that carries the public key.

The node may also obtain a certificate from a Certificate Authority that can be used to establish the trustworthiness of the node. A second option is defined to carry the certificate. Because the certificate contains the public key, there is never a need to send both options at the same time.

- o A signature generated using the private key that protects the integrity of the DHCPv6 messages and authenticates the identity of the sender.
- o A timestamp, to detect and prevent packet replay. The secure DHCPv6 nodes need to meet some accuracy requirements and be synced to global time, while the timestamp checking mechanism allows a configurable time value for clock drift.

#### 4.2. Support for algorithm agility

Hash functions are used to provide message integrity checks. In order to provide a means of addressing problems that may emerge in the future with existing hash algorithms, as recommended in [RFC4270], this document provides a mechanism for negotiating the use of more secure hashes in the future.

In addition to hash algorithm agility, this document also provides a mechanism for signature algorithm agility.

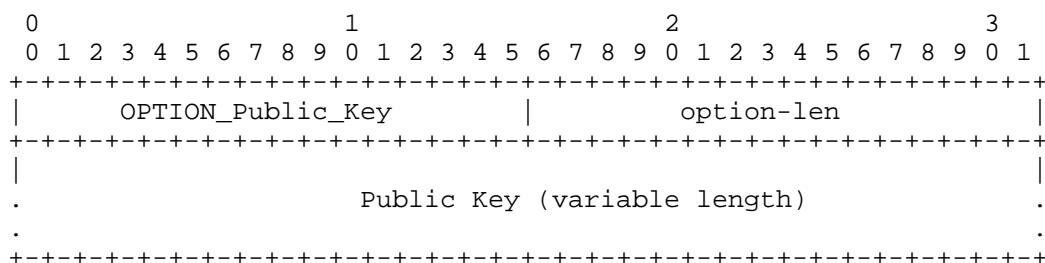
The support for algorithm agility in this document is mainly a unilateral notification mechanism from sender to recipient. If the recipient does not support the algorithm used by the sender, it cannot authenticate the message. Senders in a same administrative domain are not required to upgrade to a new algorithm simultaneously.

### 5. Extensions for Secure DHCPv6

This section extends DHCPv6. Three new options have been defined. The new options MUST be supported in the Secure DHCPv6 message exchange.

### 5.1. Public Key Option

The Public Key option carries the public key of the sender. The format of the Public Key option is described as follows:



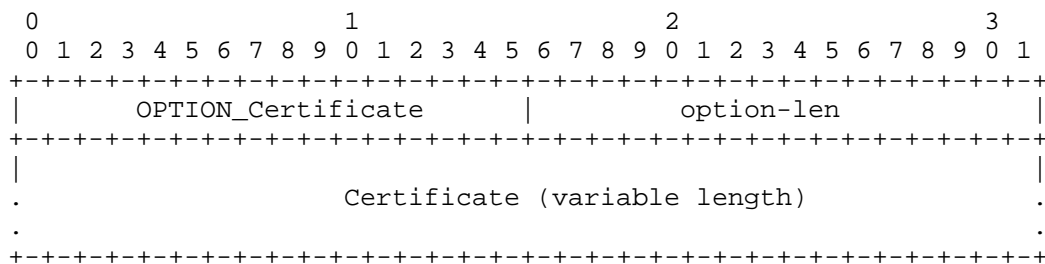
option-code      OPTION\_PK\_PARAMETER (TBA1).

option-len      Length of public key in octets.

Public Key      A variable-length field containing public key. The key MUST be represented as a lower-case hexadecimal string with the most significant octet of the key first.

### 5.2. Certificate Option

The Certificate option carries the certificate of the sender. The format of the Certificate option is described as follows:



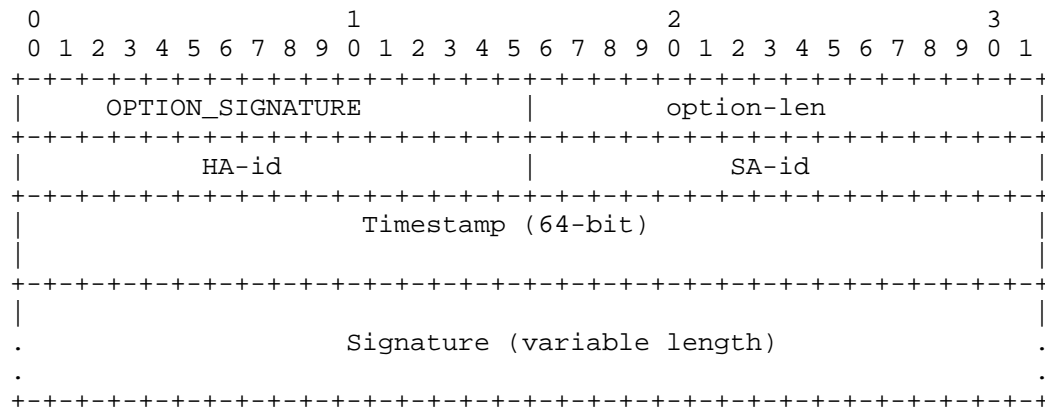
option-code      OPTION\_CERT\_PARAMETER (TBA2).

option-len      Length of certificate in octets.

**Certificate**      A variable-length field containing certificate. The encoding of certificate and certificate data MUST be in format as defined in Section 3.6, [RFC5996].

### 5.3. Signature Option

The Signature option allows public key-based signatures to be attached to a DHCPv6 message. The Signature option could be any place within the DHCPv6 message. It protects the entire DHCPv6 header and options, except for the Authentication Option. The format of the Signature option is described as follows:



**option-code**      OPTION\_SIGNATURE (TBA3).

**option-len**      12 + Length of Signature field in octets.

**HA-id**            Hash Algorithm id. The hash algorithm is used for computing the signature result. This design is adopted in order to provide hash algorithm agility. The value is from the Hash Algorithm for Secure DHCPv6 registry in IANA. The initial values are assigned for SHA-1 is 0x0001.

**SA-id**            Signature Algorithm id. The signature algorithm is used for computing the signature result. This design is adopted in order to provide signature algorithm agility. The value is from the Signature Algorithm for Secure DHCPv6 registry in IANA. The initial values are assigned for RSASSA-PKCS1-v1\_5 is 0x0001.

**Timestamp**      The current time of day (NTP-format timestamp)



[RFC5905] in UTC (Coordinated Universal Time), a 64-bit unsigned fixed-point number, in seconds relative to 0h on 1 January 1900.). It can reduce the danger of replay attacks.

**Signature** A variable-length field containing a digital signature. The signature value is computed with the hash algorithm and the signature algorithm, as described in HA-id and SA-id. The signature constructed by using the sender's private key protects the following sequence of octets:

1. The DHCPv6 message header.
2. All DHCPv6 options including the Signature option (fill the signature field with zeroes) except for the Authentication Option.

The signature field MUST be padded, with all 0, to the next octet boundary if its size is not an even multiple of 8 bits. The padding length depends on the signature algorithm, which is indicated in the SA-id field.

Note: if both signature and authentication option are presented, signature option does not protect authentication option. It is because both needs to apply hash algorithm to whole message, so there must be a clear order and there could be only one last-created option. In order to avoid update RFC3315 because of changing auth option, the authors chose not include authentication option in the signature.

## 6. Processing Rules and Behaviors

### 6.1. Processing Rules of Sender

The sender of a Secure DHCPv6 message could be a DHCPv6 server or a DHCPv6 client.

The node must have a public/private key pair in order to create Secure DHCPv6 messages. The node may have a certificate which is signed by a CA trusted by both sender and recipient.

To support Secure DHCPv6, the Secure DHCPv6 enabled sender MUST construct the DHCPv6 message following the rules defined in [RFC3315].

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain either a the Public Key or Certificate option, which MUST constructed as explained in Section 5.1 or Section 5.2.

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain the Signature option, which MUST be constructed as explained in Section 5.3. It protects the message header and the message payload and all DHCPv6 options except for the Signature option itself and the Authentication Option. Within the Signature option the Timestamp field SHOULD be set to the current time, according to sender's real time clock.

A Relay-forward and relay-reply message MUST NOT contain any Public Key or Certificate option or Signature Option.

## 6.2. Processing Rules of Recipient

When receiving a DHCPv6 message, except for Relay-Forward and Relay-Reply messages, a Secure DHCPv6 enabled recipient SHOULD discard the DHCPv6 message if the Signature option is absent, or both the Public Key and Certificate option is absent, or both the Public Key and Certificate option are presented. If all three options are absent, the recipient MAY fall back the unsecure DHCPv6 model.

The recipient SHOULD first check the authority of this sender. If the sender uses a public key, the recipient SHOULD validate it by finding a match public key from the local trust public key list, which is pre-configured or recorded from previous communications. A local trust public key list is a data table maintained by the recipient. It restores public keys from all trustworthy senders. A fast search index may be created for this data table. If the sender uses certificate, the recipient SHOULD validate the sender's certificate following the rules defined in [RFC5280]. An implementation may then create a local trust certificate record.

The recipient may choose to further process the message from a sender for which no authorization information exists. By recording the key that was used by the sender, when the first time it is seen, the recipient can make a leap of faith that the sender is trustworthy. If no evidence to the contrary surfaces, the recipient can then validate the sender as trustworthy when it subsequently sees the same key used to sign messages from the same server.

At this point, the recipient has either recognized the authorization of the sender, or decided to attempt a leap of faith. The recipient MUST now authenticate the sender by verifying the Signature and checking timestamp. The order of two procedures is left as an implementation decision. It is RECOMMENDED to check timestamp first,

because signature verification is much more computationally expensive.

The signature field verification MUST show that the signature has been calculated as specified in Section 5.3. Only the messages that get through both the signature verifications and timestamp check are accepted as secured DHCPv6 messages and continue to be handled for their contained DHCPv6 options as defined in [RFC3315]. Messages that do not pass the above tests MUST be discarded or treated as unsecure messages.

The recipient MAY record the verified public key or certificate for future authentications.

Furthermore, the node that supports the verification of the Secure DHCPv6 messages MAY record the following information:

**Minbits** The minimum acceptable key length for public keys. An upper limit MAY also be set for the amount of computation needed when verifying packets that use these security associations. The appropriate lengths SHOULD be set according to the signature algorithm and also following prudent cryptographic practice. For example, minimum length 1024 and upper limit 2048 may be used for RSA [RSA].

A Relay-forward or Relay-reply message with any Public Key, Certificate or the Signature option is invalid. The message SHOULD be discarded silently.

### 6.3. Processing Rules of Relay Agent

To support Secure DHCPv6, relay agents just need to follow the same processing rules defined in [RFC3315]. There is nothing more the relay agents have to do, either verify the messages from client or server, or add any secure DHCPv6 options. Actually, by definition in this document, relay agents MUST NOT add any secure DHCPv6 options.

### 6.4. Timestamp Check

Recipients SHOULD be configured with an allowed timestamp Delta value, a "fuzz factor" for comparisons, and an allowed clock drift parameter. The recommended default value for the allowed Delta is 300 seconds (5 minutes); for fuzz factor 1 second; and for clock drift, 0.01 second.

Note: the Timestamp mechanism is based on the assumption that communication peers have rough synchronized clocks, with certain allowed clock drift. So, accurate clock is not necessary. If one

has a clock too far from the current time, the timestamp mechanism would not work.

To facilitate timestamp checking, each recipient SHOULD store the following information for each sender, from which at least one accepted secure DHCPv6 message is successfully verified (for both timestamp check and signature verification):

- o The receive time of the last received and accepted DHCPv6 message. This is called RDlast.
- o The time stamp in the last received and accepted DHCPv6 message. This is called TSlast.

An verified (for both timestamp check and signature verification) secure DHCPv6 message initiates the update of the above variables in the recipient's record.

Recipients MUST check the Timestamp field as follows:

- o When a message is received from a new peer (i.e., one that is not stored in the cache), the received timestamp, TSnew, is checked, and the message is accepted if the timestamp is recent enough to the reception time of the packet, RDnew:

$$-\text{Delta} < (\text{RDnew} - \text{TSnew}) < +\text{Delta}$$

After the signature verification also successes, the RDnew and TSnew values SHOULD be stored in the cache as RDlast and TSlast.

- o When a message is received from a known peer (i.e., one that already has an entry in the cache), the timestamp is checked against the previously received Secure DHCPv6 message:

$$\text{TSnew} + \text{fuzz} > \text{TSlast} + (\text{RDnew} - \text{RDlast}) \times (1 - \text{drift}) - \text{fuzz}$$

If this inequality does not hold, the recipient SHOULD silently discard the message. If, on the other hand, the inequality holds, the recipient SHOULD process the message.

Moreover, if the above inequality holds and TSnew > TSlast, the recipient SHOULD update RDlast and TSlast after the signature verification also successes. Otherwise, the recipient MUST NOT update RDlast or TSlast.

An implementation MAY use some mechanism such as a timestamp cache to strengthen resistance to replay attacks. When there is a very large number of nodes on the same link, or when a cache filling attack is

in progress, it is possible that the cache holding the most recent timestamp per sender will become full. In this case, the node MUST remove some entries from the cache or refuse some new requested entries. The specific policy as to which entries are preferred over others is left as an implementation decision.

## 7. Security Considerations

This document provides new security features to the DHCPv6 protocol.

Using public key based security mechanism and its verification mechanism in DHCPv6 message exchanging provides the authentication and data integrity protection. Timestamp mechanism provides anti-replay function.

The Secure DHCPv6 mechanism is based on the pre-condition that the recipient knows the public key of senders or the sender's certificate can be verified through a trust CA. It prevents DHCPv6 server spoofing. The clients may decline the DHCPv6 messages from unknown/unverified servers, which may be fake servers; or may prefer DHCPv6 messages from known/verified servers over unsigned messages or messages from unknown/unverified servers. The pre-configuration operation also needs to be protected, which is out of scope. The deployment of PKI is also out of scope.

However, when a DHCPv6 client first encounters a new public key or new unverified certificate, it can make a leap of faith. If the DHCPv6 server that used that public key or certificate is in fact legitimate, then all future communication with that DHCPv6 server can be protected by caching the public key. This does not provide complete security, but it limits the opportunity to mount an attack on a specific DHCPv6 client to the first time it communicates with a new DHCPv6 server.

Downgrade attacks cannot be avoided if nodes are configured to accept both secured and unsecured messages. A future specification may provide a mechanism on how to treat unsecured DHCPv6 messages.

[RFC6273] has analyzed possible threats to the hash algorithms used in SEND. Since the Secure DHCPv6 defined in this document uses the same hash algorithms in similar way to SEND, analysis results could be applied as well: current attacks on hash functions do not constitute any practical threat to the digital signatures used in the signature algorithm in the Secure DHCPv6.

A window of vulnerability for replay attacks exists until the timestamp expires. Secure DHCPv6 nodes are protected against replay attacks as long as they cache the state created by the message

containing the timestamp. The cached state allows the node to protect itself against replayed messages. However, once the node flushes the state for whatever reason, an attacker can re-create the state by replaying an old message while the timestamp is still valid.

Attacks against time synchronization protocols such as NTP [RFC5905] may cause Secure DHCPv6 nodes to have an incorrect timestamp value. This can be used to launch replay attacks, even outside the normal window of vulnerability. To protect against these attacks, it is recommended that Secure DHCPv6 nodes keep independently maintained clocks or apply suitable security measures for the time synchronization protocols.

## 8. IANA Considerations

This document defines three new DHCPv6 [RFC3315] options. The IANA is requested to assign values for these three options from the DHCP Option Codes table of the DHCPv6 Parameters registry. The three options are:

The Public Key Option (TBA1), described in Section 5.1.

The Certificate Option (TBA2), described in Section 5.2.

The Signature Option (TBA3), described in Section 5.3.

The IANA is also requested to add two new registry tables to the DHCPv6 Parameters registry. The two tables are the Hash Algorithm for Secure DHCPv6 table and the Signature Algorithm for Secure DHCPv6 table.

Initial values for these registries are given below. Future assignments are to be made through Standards Action [RFC5226]. Assignments for each registry consist of a name, a value and a RFC number where the registry is defined.

Hash Algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for Hash Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
-----	-----	-----
Reserved	0x0000	this document
SHA-1	0x0001	this document
SHA-256	0x0002	this document

Signature Algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for Signature Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
-----+-----+-----		
Reserved	0x0000	this document
RSASSA-PKCS1-v1_5	0x0001	this document

## 9. Acknowledgements

The authors would like to thank Bernie Volz, Ted Lemon, Ralph Droms, Jari Arkko, Sean Turner, Stephen Kent, Thomas Huth, David Schumacher, Dacheng Zhang, Francis Dupont and other members of the IETF DHC working groups for their valuable comments.

This document was produced using the xml2rfc tool [RFC2629].

## 10. Change log [RFC Editor: Please remove]

draft-jiang-dhc-sedhcpv6-01: removed protection between relay agent and server due to complexity, following the comments from Ted Lemon, Bernie Volz. 2013-10-16.

draft-jiang-dhc-sedhcpv6-01: update according to review comments from Ted Lemon, Bernie Volz, Ralph Droms. Separated Public Key/Certificate option into two options. Refined many detailed processes. 2013-10-08.

draft-jiang-dhc-sedhcpv6-00: original version, this draft is a replacement of draft-ietf-dhc-secure-dhcpv6, which reached IESG and dead because of consideration regarding to CGA. The authors followed the suggestion from IESG making a general public key based mechanism. 2013-06-29.

## 11. References

### 11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.
- [RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

#### 11.2. Informative References

- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.
- [RFC4270] Hoffman, P. and B. Schneier, "Attacks on Cryptographic Hashes in Internet Protocols", RFC 4270, November 2005.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6273] Kukec, A., Krishnan, S., and S. Jiang, "The Secure Neighbor Discovery (SEND) Hash Threat Analysis", RFC 6273, June 2011.
- [RSA] RSA Laboratories, "RSA Encryption Standard, Version 2.1, PKCS 1", November 2002.

#### Authors' Addresses

Sheng Jiang  
Huawei Technologies Co., Ltd  
Q14, Huawei Campus, No.156 Beiqing Road  
Hai-Dian District, Beijing, 100095  
P.R. China

Email: [jiangsheng@huawei.com](mailto:jiangsheng@huawei.com)



Sean Shen  
CNNIC  
4, South 4th Street, Zhongguancun  
Beijing 100190  
P.R. China

Email: shenshuo@cnnic.cn

DHC  
Internet-Draft  
Intended status: Standards Track  
Expires: January 06, 2014

D. Migault (Ed)  
Francetelecom - Orange  
W. Cloetens  
SoftAtHome  
C. Griffiths  
Dyn  
R. Weber  
Nominum  
July 05, 2013

DHCP DNS Public Authoritative Server Option  
draft-mglt-dhc-public-authoritative-server-option-00.txt

Abstract

The home network naming architecture as described in [I-D.mglt-homenet-front-end-naming-delegation] requires a complex naming configuration on the CPE. This configuration MAY not be handled easily by the average end user. Furthermore, such misconfiguration MAY result in making home network unreachable.

This document proposes a DHCP option that provides the CPE all necessary parameters to set up the home network naming architecture.

First, this DHCP option provides automatic configuration and avoids most end users' misconfigurations. Most average end users may not require specific configuration, and their ISP default configuration MAY fully address their needs. In that case, the naming homenet architecture configuration will be completely transparent to the end users. Then, saving naming configuration outside the CPE, makes it resilient to change of CPE or CPE upgrades. Such configuration may also be configured by the end user, via the customer area of their ISP.

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 06, 2014.

#### Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

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

#### Table of Contents

1. Requirements notation . . . . .	3
2. Terminology . . . . .	3
3. Introduction . . . . .	4
4. Protocol Overview . . . . .	5
5. Payload Description . . . . .	7
5.1. DNS Public Authoritative Server Option . . . . .	7
5.2. registered-domain-list . . . . .	8
5.3. master-list payload . . . . .	8
5.4. secure-channel-list payload . . . . .	8
6. Exchange Details . . . . .	10
6.1. DHCPv6 Server . . . . .	10
6.2. CPE . . . . .	10
7. IANA Considerations . . . . .	11
8. Security Considerations . . . . .	11
8.1. DNSSEC is recommended to authenticate DNS hosted data . .	11
8.2. Channel between the CPE and ISP DHCP Server MUST be secured . . . . .	12
8.3. CPEs are sensitive to DoS . . . . .	12
9. Acknowledgment . . . . .	12
10. References . . . . .	12
10.1. Normative References . . . . .	12
10.2. Informational References . . . . .	13
Authors' Addresses . . . . .	13

## 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. Terminology

- Customer Premises Equipment: (CPE) is the router providing connectivity to the home network. It is configured and managed by the end user. In this document, the CPE MAY also hosts services such as DHCPv6. This device MAY be provided by the ISP.
- Registered Homenet Domain: is the Domain Name associated to the home network.
- DNS Homenet Zone: is the DNS zone associated to the home network. This zone is set by the CPE and essentially contains the bindings between names and IP addresses of the nodes of the home network. In this document, the CPE does neither perform any DNSSEC management operations such as zone signing nor provide an authoritative service for the zone. Both are delegated to the Public Authoritative Server. The CPE synchronizes the DNS Homenet Zone with the Public Authoritative Server via a hidden master / slave architecture. The Public Authoritative Server MAY use specific servers for the synchronization of the DNS Homenet Zone: the Public Authoritative Name Server Set.
- Public Authoritative Server: performs DNSSEC management operations as well as provides the authoritative service for the zone. In this document, the Public Authoritative Server synchronizes the DNS Homenet Zone with the CPE via a hidden master / slave architecture. The Public Authoritative Server acts as a slave and MAY use specific servers called Public Authoritative Name Server Set. Once the Public Authoritative Server synchronizes the DNS Homenet Zone, it signs the zone and generates the DNSSEC Public Zone. Then the Public Authoritative Server hosts the zone as an authoritative server on the Public Authoritative Master(s).
- DNSSEC Public Zone: corresponds to the signed version of the DNS Homenet Zone. It is hosted by the Public Authoritative Server, which is authoritative for this zone, and is reachable on the Public Authoritative Master(s).

- Public Authoritative Master(s): are the visible name server hosting the DNSSEC Public Zone. End users' resolutions for the Homenet Domain are sent to this server, and this server is a master for the zone.
- Public Authoritative Name Server Set: is the server the CPE synchronizes the DNS Homenet Zone. It is configured as a slave and the CPE acts as master. The CPE sends information so the DNSSEC zone can be set and served.

### 3. Introduction

With IPv6, nodes inside the home network are expected to be globally reachable. CPEs are already providing connectivity to the home network, and most of the time already assigns IP addresses to the nodes of the home network using for example DHCPv6.

This makes CPE good candidate for defining the DNS zone file of the home network. However, CPEs have not been designed to handle heavy traffic, nor heavy operations. As a consequence, CPE SHOULD neither host the authoritative naming service of the home network, nor handle DNSSEC operations such as zone signing. In addition, CPE are usually managed by end users, and the average end user is most likely not mastering DNSSEC to administrate its DNSSEC zone. As a result, CPE SHOULD outsource both the naming authoritative service and its DNSSEC management operations to a third party. This architecture, designated as the homenet naming architecture is described in [I-D.mglt-homenet-front-end-naming-delegation], and the third party is designated as the Public Authoritative Servers.

The home network naming architecture defines how the CPE and the Public Authoritative Servers interact together, so to leverage some of the issues related to the CPE, and the DNSSEC understanding of the average end user. Even though most of the DNSSEC issues are outsourced to the Public Authoritative Servers, setting the homenet naming architecture still requires some configurations.

Configuration is fine as it provides the opportunity for advanced end users to make the naming architecture fit their specific needs. However most of the end users do not want to configure the homenet naming architecture. In most cases, the end users wants to subscribe and plug its CPE. The CPE is expected to be configured to set automatically and transparently the appropriated home network naming architecture.

Using DHCP options to provide the necessary parameters for setting the homenet naming architecture provides multiple advantages. Firstly, it makes the network configuration independent of the CPE.

Any new plugged CPE configures itself according to the provided configuration parameters. Secondly, it saves the configuration outside the CPE, which prevents re-configuring the CPE when it is replaced or reset. Finally ISPs are likely to propose a default homenet naming architecture that may address most of the end users needs. For these end users, no configuration will be performed at any time. This avoids unnecessary configurations or misconfiguration that could result in isolating the home network. For more advanced end users, the configuration MAY be provided also via the web GUI of the ISP's customer area for example. This configuration MAY enable third party Public Authoritative Servers. By doing so, these end users will also benefit from CPE-independent configuration and configuration backup.

This document considers the architecture described in [I-D.mglt-homenet-front-end-naming-delegation]. The DNS(SEC) zone related to the home network is configured and set by the CPE and hosted on a Public Authoritative Server. [I-D.mglt-homenet-front-end-naming-delegation] describes how the synchronization between the CPE and the Public Authoritative Server is performed. This document describes the DNS Public Authoritative Server DHCP option (DNS\_PUBLIC\_AUTHORITY\_SERVER) that provides the necessary parameters to the CPE to set the architecture described in [I-D.mglt-homenet-front-end-naming-delegation].

Section 4 presents an overview of the DNS Public Authoritative Server DHCP option (DNS\_PUBLIC\_AUTHORITY\_SERVER) and Section 5 describes the format of this option and Section 6 details the exchange between the CPE and the DHCPv6 Server.

This document assumes the reader is familiar with [I-D.mglt-homenet-front-end-naming-delegation].

This document assumes that the communication between the CPE and the ISP DHCP Server is protected. This document does not specify which mechanism should be used. [RFC3315] proposes a DHCP authentication and message exchange protection, [RFC4301], [RFC5996] proposes to secure the channel at the IP layer.

This document only deals with IPv6 IP addresses and DHCPv6 [RFC3315]. When we mention DHCP, it MUST be understood as DHCPv6.

#### 4. Protocol Overview

The CPE requests the necessary parameters to set its home network naming configuration to the DHCP server. The DHCP server MAY be, for example, the one of its ISP, that already provides the IPv6 prefix to the CPE.

The CPE sends an Option Request DHCP Option (ORO) [RFC3315] for the DHCP DNS Public Authoritative Server Option (DNS\_PUBLIC\_AUTHORITATIVE\_SERVER)

If available, the DHCP server sends back one or more DHCP DNS Public Authoritative Server Option (DNS\_PUBLIC\_AUTHORITATIVE\_SERVER), depending if the end user has registered to one or multiple Public Authoritative Servers.

A CPE MAY be associated to one or multiple Registered Homenet Domain and one or multiple Public Authoritative Servers. The CPE builds a zone for each Registered Homenet Domain. These zones are uploaded / synchronized with their associated Public Authoritative Servers. Note that synchronization is performed through master / slave configuration of the DNS servers, thus Public Authoritative Servers are configured to host specific Registered Homenet Domains. On the other hand, a given Homenet Zone MAY be hosted by multiple Public Authoritative Servers. The CPE MUST build properly the DNS Homenet Zone and synchronize properly the hidden master and slaves for the synchronizations.

In order to configure properly the DNS Homenet Zone, the CPE SHOULD collect the list of Registered Homenet Domain and their associated Public Authoritative Servers. For each Registered Homenet Domain, the CPE lists the Public Authoritative Servers FQDN and set them as authoritative Name Server (RRset NS) for the zone. The CPE MAY also add in the DNS Homenet Zone their IP addresses (RRsets A or AAAA). This FQDN and IP addresses associated are designated as the Public Authoritative Master(s).

Note that how the CPE manage the multiple DNS Homenet Zones is implementation dependent. It MAY synchronize all DNS Homenet Zone with the Public Authoritative Servers, or use zone redirection mechanisms like CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsext-cname-dname]. In the first case, any update requires to update all zone, whereas redirection MAY require only updating a single DNS Homenet Zone.

In order to synchronize the DNS Homenet Zone with a Public Authoritative Server, the CPE needs to know how to establish a secure channel with the Public Authoritative Server. The Public Authoritative Server MAY have dedicated servers working as slave to synchronize the DNS Homenet Zone with the CPE. These servers are called Public Authoritative Name Server Set. In addition to these servers, the CPE MUST know which security protocol can be used to secure the channel as well as the security credential. In this document, we only considered Pre-Shared Key (PSK).

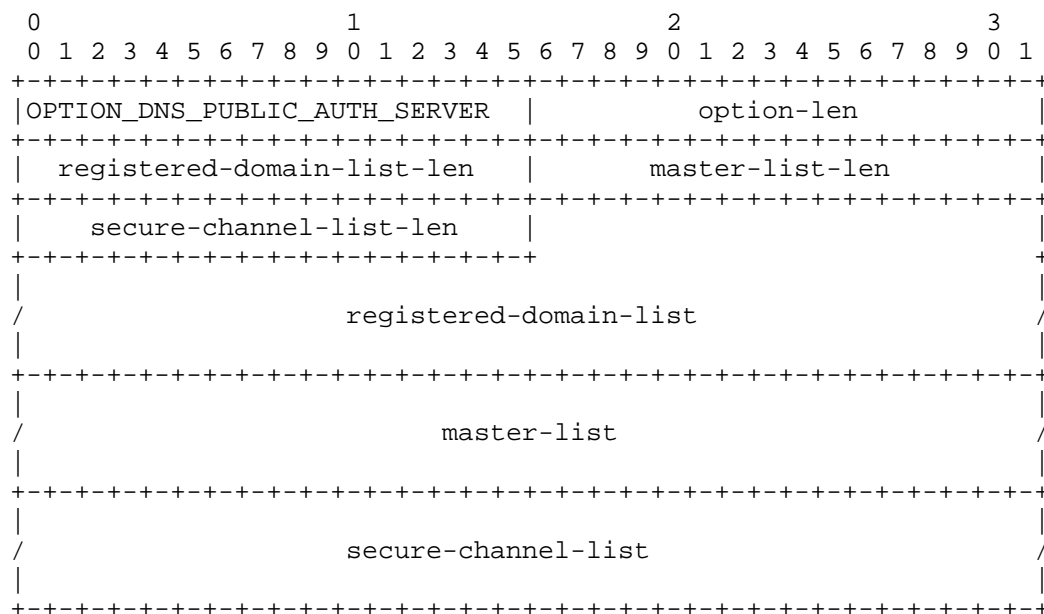
As a result, the DHCP DNS Public Authoritative Server Option (DNS\_PUBLIC\_AUTHORITY\_SERVER) carries:

- Registered Homenet Domain List: the list of Registered Homenet Domain the Public Authoritative Server hosts.
- Master : the Public Authoritative Master(s), that is to say the FQDNs provided in the NS RRsets of the Homenet Zones associated to each of the Registered Homenet Domains. IP addresses are derived by the CPE thanks to a DNS(SEC) resolutions.
- Secure Channel: the Public Authoritative Name Server Set , that is the FQDN the CPE MUST securely synchronize its DNS Homenet Zone with. This field MUST also specify, the security protocol as well as the security material.

## 5. Payload Description

### 5.1. DNS Public Authoritative Server Option

The DHCP DNS Public Authoritative Server Option is constituted of three ordered sub payloads: the registered-domain-list, the master-list and the secure-channel-list payload.



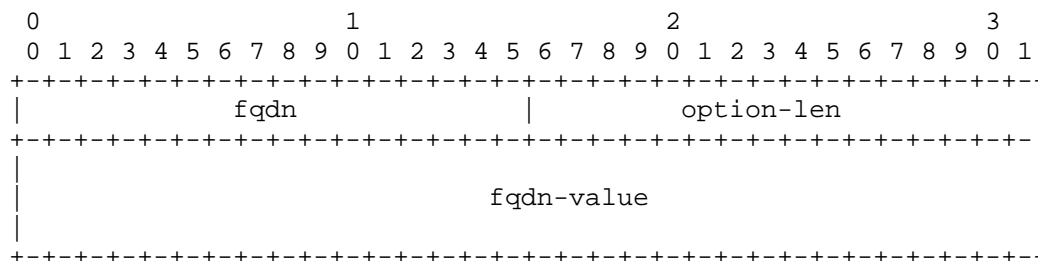
- option-code: OPTION\_DNS\_PUBLIC\_AUTH\_SERVER



- option-len: length of the OPTION\_DNS\_PUBLIC\_AUTH\_SERVER field, the option-code and the option-message in octets.
- registered-domain-list-len: Length in octets of the list of Registered Homenet Domains field.
- master-list-len: Length in octets of the list of the master-list field.
- secure-channel-list-len: Length in octets of the Secure Channels field.
- registered-domain-list: The list of Registered Homenet Domains.
- master-list: The list of Public Authoritative Master(s).
- secure-channel-list: The list of Secure Channels

## 5.2. registered-domain-list

The registered-domain-list contains a list of fqdn payloads. The fqdn payload is as described below:



- payload-code: fqdn
- option-len: length of the fqdn field, the payload-code and the payload-message in octets.
- fqdn-value: fqdn value as specified in [RFC1035]

## 5.3. master-list payload

The master-list payload contains a list of fqdn payloads. The fqdn payload is defined in Section 5.2.

## 5.4. secure-channel-list payload

The registered-domain-list contains a list of secure-channel payloads. The secure-channel payload is described below.

```

      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
+-----+-----+-----+-----+-----+-----+-----+-----+
|               secure-channel               |               option-len |
+-----+-----+-----+-----+-----+-----+-----+-----+
| sec-protocol | sec-cred-type | security-credential-len |
+-----+-----+-----+-----+-----+-----+-----+-----+
|               name-server-set-len |
+-----+-----+-----+-----+-----+-----+-----+-----+
|
/               security-credential (PSK)               /
|
+-----+-----+-----+-----+-----+-----+-----+-----+
/               name-server-set               |
|
+-----+-----+-----+-----+-----+-----+-----+-----+

```

- payload-code: secure-channel-list. Although not necessary, since payloads are ordered, we provide this code to ease implementation and future options.
- option-len: Length of the delegated-naming-action-list field, the status-code and the status-message in octets.
- sec-protocol: the protocol that secures the exchanges between the CPE and the Public Authoritative Server. Possible protocols are NONE, TSIG, IPSEC.
- sec-cred-type: the type of credential used between the CPE and the Public Authoritative Server. The current document considers only PSK that can be used with any of the sec-protocol.
- security-credential-len: length of the delegated-naming-action-list field, the sec-cred-type and the security-credential-len in octets.
- security-credential: the security credential. In this document, the security credential is of type PSK.
- name-server-set-len: length of the name-server-set field and the name-server-set-len in octets.

- name-server-set: Public Authoritative Name Server Set encoded as specified in [RFC1035]. Only the FQDN representation is considered in this document.

## 6. Exchange Details

This section details the DHCPv6 exchange between the DHCPv6 server and the CPE.

### 6.1. DHCPv6 Server

The DHCPv6 server MUST NOT send a DHCP DNS Public Authoritative Server Option (DNS\_PUBLIC\_AUTHORITY\_SERVER) if not requested by the CPE through the DHCP Option Request Option (ORO).

In the remaining of this section we suppose the DHCPv6 Server has received a DHCP Option Request Option (ORO) from the CPE for a DHCP DNS Public Authoritative Server Option (DNS\_PUBLIC\_AUTHORITY\_SERVER).

If the DHCPv6 Server does not understand the DHCP DNS Public Authoritative Server Option, it ignores the Option as specified in [RFC3315].

If the DHCPv6 has no specific configuration, it MUST return a DHCP DNS Public Authoritative Server Option with the len registered-domain-list-len, master-list-len and secure-channel-list-len set to zero. This response is called the Zero Response and indicates that there are not enough arguments to set the home network architecture.

The registered-domain-list is mandatory. If it does not exist, and Zero Response MUST be sent.

A zero length for master-list indicates the CPE hosts the zone. In this case, a zero length secure-channel-list is expected.

### 6.2. CPE

In this section we assume the CPE has sent a DHCP Option Request Option (ORO) from the CPE for a DHCP DNS Public Authoritative Server Option (DNS\_PUBLIC\_AUTHORITY\_SERVER).

An Zero Response indicates the DHCPv6 Server has not a specific configuration for the CPE.

A response with an registered-domain-list set to zero MUST be ignored.

A zero length for master-list indicates the CPE hosts the zone. A non zero length secure-channel-list MUST be ignored.

## 7. IANA Considerations

The DHCP options detailed in this document is:

- OPTION\_DNS\_PUBLIC\_AUTH\_SERVER: TBD

The payload detailed in this document are:

- fqdn: TBD
- secure-channel: TBD

The security-protocol detailed in this document are:

- NONE: TBD
- TSIG: TBD
- IPSEC: TBD

The security-credential detailed in this document are:

- NONE: TBD
- PSK: TBD

## 8. Security Considerations

### 8.1. DNSSEC is recommended to authenticate DNS hosted data

The document describes how the Secure Delegation can be set between the Delegating DNS Server and the Delegated DNS Server.

Deploying DNSSEC is recommended since in some cases the information stored in the DNS is used by the ISP or an IT department to grant access. For example some Servers may performed a PTR DNS query to grant access based on host names. With the described Delegating Naming Architecture, the ISP or the IT department MUST take into consideration that the CPE is outside its area of control. As such, with DNS, DNS responses may be forged, resulting in isolating a Service, or not enabling a host to access a service. ISPs or IT department may not base their access policies on PTR or any DNS information. DNSSEC fulfills the DNS lack of trust, and we recommend to deploy DNSSEC on CPEs.

## 8.2. Channel between the CPE and ISP DHCP Server MUST be secured

In the document we consider that the channel between the CPE and the ISP DHCP Server is trusted. More specifically, we suppose the CPE is authenticated and the exchanged messages are protected. The current document does not specify how to secure the channel. [RFC3315] proposes a DHCP authentication and message exchange protection, [RFC4301], [RFC5996] propose to secure the channel at the IP layer.

In fact, the channel MUST be secured because the CPE provides necessary information for the configuration of the Naming Delegation. Unsecured channel may result in setting the Naming Delegation with an non legitimate CPE. The non legitimate CPE would then be redirected the DNS traffic that is intended for the legitimate CPE. This makes the CPE sensitive to three types of attacks. The first one is the Deny Of Service Attack, if for example DNS traffic for a lot of CPEs are redirected to a single CPE. CPE are even more sensitive to this attack since they have been designed for low traffic. The other type of traffic is the DNS traffic hijacking. A malicious CPE may redirect the DNS traffic of the legitimate CPE to one of its server. In return, the DNS Servers would be able to provide DNS Responses and redirect the End Users on malicious Servers. This is particularly used in Pharming Attacks. A third attack may consists in isolating a Home Network by misconfiguring the Naming Delegation for example to a non-existing DNS Server, or with a bad DS value.

## 8.3. CPEs are sensitive to DoS

The Naming Delegation Architecture involves the CPE that hosts a DNS Server for the Home Network. CPE have not been designed for handling heavy load. The CPE are exposed on the Internet, and their IP address is publicly published on the Internet via the DNS. This makes the Home Network sensitive to Deny of Service Attacks. The Naming Delegation Architecture described in this document does not address this issue. The issue is addressed by [I-D.mglt-homenet-front-end-naming-delegation].

## 9. Acknowledgment

## 10. References

### 10.1. Normative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.
- [RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, June 2012.

## 10.2. Informational References

- [I-D.mglt-homenet-front-end-naming-delegation]  
Migault, D., Cloetens, W., Lemordant, P., and C. Griffiths, "IPv6 Home Network Front End Naming Delegation", draft-mglt-homenet-front-end-naming-delegation-01 (work in progress), November 2012.
- [I-D.sury-dnsextns-cname-dname]  
Sury, O., "CNAME+DNAME Name Redirection", draft-sury-dnsextns-cname-dname-00 (work in progress), April 2010.

## 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

Wouter Cloetens  
SoftAtHome  
vaartdijk 3 701  
3018 Wijkmaal  
Belgium

Email: [wouter.cloetens@softathome.com](mailto:wouter.cloetens@softathome.com)

Chris Griffiths  
Dyn  
150 Dow Street  
Manchester, NH 03101  
US

Email: [cgriffiths@dyn.com](mailto:cgriffiths@dyn.com)  
URI: <http://dyn.com>

Ralf Weber  
Nominum  
2000 Seaport Blvd #400  
Redwood City, CA 94063  
US

Email: [ralf.weber@nominum.com](mailto:ralf.weber@nominum.com)  
URI: <http://www.nominum.com>

Network Working Group  
Internet-Draft  
Expires: March 30, 2014

B. Sarikaya  
F. Xia  
Huawei USA  
September 26, 2013

DHCP Options for Configuring Multicast Addresses in VXLAN  
draft-sarikaya-dhc-vxlan-multicast-02.txt

Abstract

This document defines DHCPv4 and DHCPv6 options for assigning multicast addresses for the Tunnel End Point in the Virtual eXtensible Local Area Network (VXLAN) environments. New DHCP options are defined which allow a VXLAN Tunnel End Point to request any source multicast address for the newly created virtual machine, the address of the Rendezvous Point (RP) and possibly address(es) for the virtual machine.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). 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 March 30, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

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



the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	4
3. Overview of the protocol . . . . .	4
4. DHCPv6 Options . . . . .	5
4.1. VXLAN Network Identifier Option . . . . .	5
4.2. IPv6 multicast address for the VNI Option . . . . .	6
4.3. IPv6 Rendezvous Point Address Option . . . . .	7
5. DHCPv4 Options . . . . .	7
5.1. VXLAN Network Identifier Option . . . . .	7
5.2. VXLAN Multicast Address Option . . . . .	8
5.3. VXLAN Rendezvous Point Address Option . . . . .	8
6. Client Operation . . . . .	9
7. Server Operation . . . . .	10
8. Security Considerations . . . . .	11
9. IANA considerations . . . . .	11
10. Acknowledgements . . . . .	11
11. References . . . . .	11
11.1. Normative References . . . . .	11
11.2. Informative References . . . . .	12
Authors' Addresses . . . . .	13

## 1. Introduction

Data center networks are being increasingly used by telecom operators as well as by enterprises. Currently these networks are organized as one large Layer 2 network in a single building. In some cases such a network is extended geographically using virtual Local Area Network (VLAN) technologies still as an even larger Layer 2 network connecting the virtual machines (VM), each with its own MAC address.

Another important requirement was growing demand for multitenancy, i.e. multiple tenants each with their own isolated network domain. In a data center hosting multiple tenants, each tenant may independently assign MAC addresses and VLAN IDs and this may lead to potential duplication.

What we need is IP based tunneling scheme based overlay network called Virtual eXtensible Local Area Network (VXLAN). VXLAN overlays a Layer 2 network over a Layer 3 network. Each overlay, identified by the VXLAN Network Identifier (VNI). This allows up to 16M VXLAN segments to coexist within the same administrative domain [I-D.mahalingam-dutt-dcops-vxlan]. In VXLAN, each MAC frame is transmitted after encapsulation, i.e. an outer Ethernet header, an IPv4/IPv6 header, UDP header and VXLAN header are added. Outer Ethernet header indicates an IPv4 or IPv6 payload. VXLAN header contains 24-bit VNI.

VXLAN tunnel end point (VTEP) is the hypervisor on the server which houses the VM. VXLAN encapsulation is only known to the VTEP, the VM never sees it. Also the tunneling is stateless, each MAC frame is encapsulated independent on any other MAC frame.

Instead of using UDP header, Generic Routing Encapsulation (GRE) encapsulation can be used. A 24-bit Virtual Subnet Identifier (VSID) is placed in the GRE key field. The resulting encapsulation is called Network Virtualization using Generic Routing Encapsulation (NVGRE) [I-D.sridharan-virtualization-nvgre]. Note that VSID is similar to VNI. Although VXLAN terminology is used throughout, the protocol defined in this document applies to VXLAN as well as NVGRE.

In this document, we develop a protocol to assign multicast addresses to the VXLAN tunnel end points using Dynamic Host Configuration Protocol (DHCP). Multicast communication in VXLAN is used for sending broadcast MAC frames, e.g. the Address Resolution Protocol (ARP) broadcast frame. Multicast communication can also be used to transmit multicast frames and unknown MAC destination frames.

## 2. 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]. The terminology in this document is based on the definitions in [I-D.mahalingam-dutt-dcops-vxlan]

## 3. Overview of the protocol

Multicast addresses to be assigned by the DHCP server are administratively scoped multicast addresses, in IPv4 [RFC2365] and in IPv6 [RFC4291]. The steps involved in the protocol are explained below for IPv4:

### Creation of a VM

In this step, VTEP receives a request from the Management Node to create a Virtual Machine with a VXLAN Network Identifier.

### DHCP Operation

VTEP starts DHCP state machine by sending DHCPDISCOVER message to the default router, e.g. the Top of Rack (ToR) switch. ToR switch could be DHCP server or most possibly DHCP relay with DHCP server located upstream. VTEP MUST include the VXLAN Multicast Address and VXLAN Rendezvous Point Address options defined in this document. VTEP sends the VXLAN Network Identifier in the newly defined VNI DHCP Option. DHCP server replies with DHCPOFFER message. DHCP server sends administratively scope IPv4 multicast address and RP address to VTEP. VTEP checks this message and if it sees the options it requested, DHCP server is confirmed to support the multicast address options. DHCPREQUEST message from VTEP and DHCPACK message from DHCP server complete DHCP message exchange.

### VTEP as Multicast Source

After receiving the required information, the VTEP as multicast source communicates with the Rendezvous Point in order to build the multicast tree.

### VTEP as Listener

After receiving the required information, the VTEP as listener communicates with the edge router by sending MLD Report to join the multicast group.

IPv6 operation is slightly different:

Creation of a VM

In this step, VTEP receives a request from the Management Node to create a Virtual Machine with a VXLAN Network Identifier.

DHCP Operation

VTEP starts DHCP state machine by sending DHCPv6 Solicit message to the default router, e.g. the Top of Rack (ToR) switch. ToR switch could be DHCP server or most possibly DHCP relay with DHCP server located upstream. VTEP MUST include the options defined in this document. DHCP server replies with DHCPv6 Advertise message. VTEP checks this message and if it sees the options it requested, DHCP server is confirmed to support multicast address options. DHCPv6 Request message from VTEP and DHCPv6 Reply message from DHCPv6 server complete DHCP message exchange.

VTEP as Multicast Source

After receiving the required information, the VTEP as multicast source communicates with the Rendezvous Point in order to build the multicast tree.

VTEP as Listener

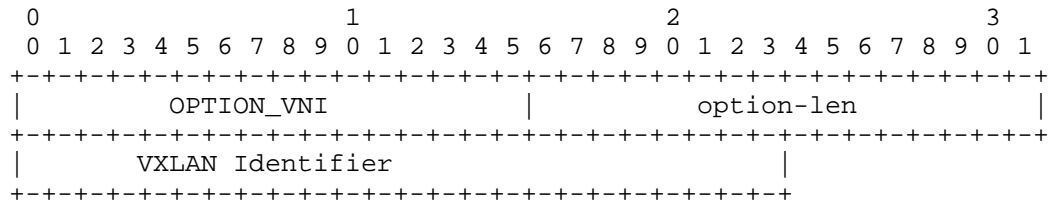
After receiving the required information, the VTEP as listener communicates with the edge router by sending MLD Report to join the multicast group.

#### 4. DHCPv6 Options

##### 4.1. VXLAN Network Identifier Option

Different VXLAN Network Identifiers (VNI) need different multicast groups, and even rendezvous point addresses (for load balancing). At the same time, different VNIs need different address spaces for VM, that is, two VMs belongs to different VNIs probably have the same IP address.

Because of the reasons stated above, a DHCP VNI Option is defined as follows.



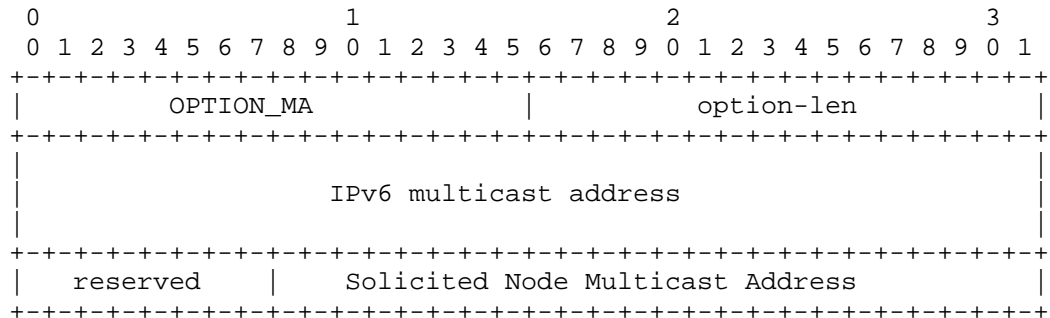
option-code    OPTION\_VNI (TBD).

option-len    7.

VXLAN Network Identifier 3.

#### 4.2. IPv6 multicast address for the VNI Option

The option allows the VTEP to send the VNI and solicited-node multicast address to DHCP server and receive administratively scoped IPv6 multicast address.



option-code    OPTION\_MA (TBD).

option-len    24.

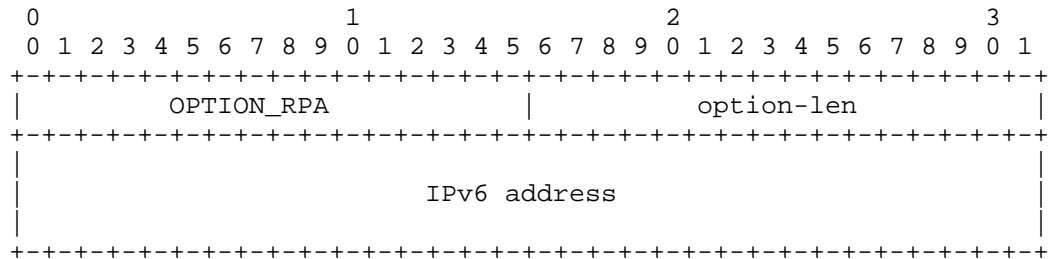
IPv6 multicast address    An IPv6 address.

reserved    must be set to zero

Solicited Node Multicast Address as in RFC 4861.

#### 4.3. IPv6 Rendezvous Point Address Option

The option allows the VTEP to receive RP address for Any Source Multicast group from DHCP server.



option-code    OPTION\_RPA (TBD).

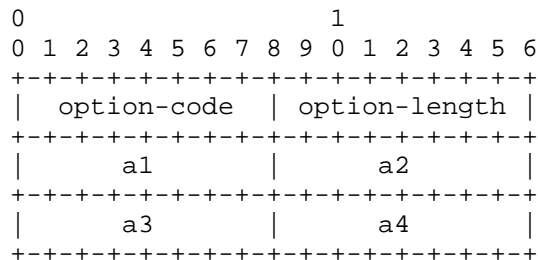
option-len    20.

IPv6 multicast address    An IPv6 address

### 5. DHCPv4 Options

#### 5.1. VXLAN Network Identifier Option

The option allows the VTEP to send the VNI to DHCP server.



Option-code  
VXLAN Network Identifier Option (TDB)

Option-len  
4.

a1-a4

VTEP as DHCP Client sets a1-a3 to VNI and a4 to zero.

## 5.2. VXLAN Multicast Address Option

The option allows the VTEP to send the VNI DHCP server and receive administratively scoped IPv4 multicast address.

```

0                               1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
+---+---+---+---+---+---+---+---+
| option-code | option-length |
+---+---+---+---+---+---+---+---+
|      a1      |      a2      |
+---+---+---+---+---+---+---+---+
|      a3      |      a4      |
+---+---+---+---+---+---+---+---+

```

Option-code

VXLAN Multicast Address Option (TDB)

Option-len

4.

a1-a4

VTEP as DHCP Client sets a1-a4 to zero, DHCP server sets a1-a4 to the multicast address.

## 5.3. VXLAN Rendezvous Point Address Option

This option is used to receive VXLAN Rendezvous Point address from DHCP server.

```

0                                     1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
+---+---+---+---+---+---+---+---+---+
| option-code | option-length |
+---+---+---+---+---+---+---+---+
|      a1      |      a2      |
+---+---+---+---+---+---+---+---+
|      a3      |      a4      |
+---+---+---+---+---+---+---+---+

```

Option-code

VXLAN Rendezvous Point Address Option (TBD)

Option-len

4.

a1-a4

VXLAN Rendezvous Point Address an IPv4 address

## 6. Client Operation

In DHCPv4, the client, VTEP MUST set 'htype' and 'chaddr' fields to specify the client link-layer address type and the link-layer address. The client must set the hardware type, 'htype' to 1 for Ethernet [RFC1700] and 'chaddr' is set to the MAC address of the virtual machine.

The client MUST set VXLAN Multicast Address Option to zero. The client MUST set VXLAN Rendezvous Point Address Option to zero. The client MUST set VXLAN Network Identifier Option to the VXLAN network identifier assigned to the virtual machine.

In DHCPv6, the client MUST use OPTION\_CLIENT\_LINKLAYER\_ADDR defined in [RFC6939] to send the MAC address. In this option, link-layer type MUST be set to 1 for Ethernet and link-layer address MUST be set to the MAC address of VM. Note that in [RFC6939], OPTION\_CLIENT\_LINKLAYER\_ADDR is defined to be used in Relay-Forward DHCP message. In this document this option MUST be sent in DHCPv6 Solicit message.

The client MUST set IPv6 VNI Option OPTION\_VNI to the VXLAN network identifier assigned to the virtual machine.

The Client MUST set IPv6 multicast address for the VNI Option's multicast address field to zero.



The client MUST set IPv6 Rendezvous Point Address Option's IPv6 multicast address field to zero.

The client MUST set Solicited Node Multicast Address to zero if the neighbor discovery message is sent to all-nodes multicast address. The client MUST set Solicited Node Multicast Address to the low-order 24 bits of an address of the destination if the neighbor discovery message is sent to the solicited-node multicast address.

## 7. Server Operation

If DHCPv4 server is configured to support VXLAN multicast address assignments, it SHOULD look for VXLAN Multicast Address Option and VXLAN Rendezvous Point Address Option in DHCPDISCOVER message. The server MUST return in VXLAN Multicast Address Option's a1-a4 an organization-local scope IPv4 multicast address (239.192.0.0/14) [RFC2365]. The server MUST use the VNI value for obtaining the organization-local scope IPv4 multicast address. VNI value is directly copied to 239.192.0.0/14 if the first 6 bits are zero, i.e. no overflow ranges need to be used. Otherwise, either of 239.0.0.0/10, 239.64.0.0/10 and 239.128.0.0/10 overflow ranges SHOULD be used. Note that these ranges can accomodate the VNI in its entirety.

The server MUST set VXLAN Rendezvous Point Address Option's VXLAN Rendezvous Point Address field to IPv4 unicast address of the Rendezvous Point for the any source multicast Rendezvous Point router. How this assignment is done is out of scope.

If DHCPv6 server is configured to support VXLAN multicast address assignments it SHOULD look for IPv6 multicast address for the VNI Option and IPv6 Rendezvous Point Address Option in DHCPv6 Solicit message. The server MUST return in IPv6 multicast address field an Admin-Local scope IPv6 multicast address (FF04/16) by copying the VNI of the virtual machine to the least significant 24 bits of the group ID field and setting all other bits to zero if Solicited Node Multicast Address field received from the client was set to zero. Otherwise the Solicited Node Multicast Address field is copied to bits 47-24 of the group ID field and all leading bits are set to zero.

The server MUST assign IPv6 Rendezvous Point Address Option's IPv6 address field to the Rendezvous Point router's address in charge of this multicast group. The unicast address MUST BE assigned according to the rules defined in [RFC3956].

## 8. Security Considerations

The security considerations in [RFC2131], [RFC2132] and [RFC3315] apply. Special considerations in [I-D.mahalingam-dutt-dcops-vxlan] are also applicable.

## 9. IANA considerations

IANA is requested to assign the OPTION\_VNI and OPTION\_MA and OPTION\_RPA and VXLAN Network Identifier and VXLAN Multicast Address and VXLAN Rendezvous Point Address Option Codes in the registry maintained for DHCPv4 and DHCPv6.

## 10. Acknowledgements

The authors are grateful to Bernie Volz for providing comments that helped us improve the document.

## 11. References

### 11.1. Normative References

- [RFC1700] Reynolds, J. and J. Postel, "Assigned Numbers", RFC 1700, October 1994.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3956] Savola, P. and B. Haberman, "Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address", RFC 3956, November 2004.
- [RFC2365] Meyer, D., "Administratively Scoped IP Multicast", BCP 23, RFC 2365, July 1998.

- [RFC4291]    Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.
- [RFC4861]    Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.
- [RFC6939]    Halwasia, G., Bhandari, S., and W. Dec, "Client Link-Layer Address Option in DHCPv6", RFC 6939, May 2013.

#### 11.2. Informative References

- [I-D.mahalingam-dutt-dcops-vxlan]  
Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-04 (work in progress), May 2013.
- [I-D.sridharan-virtualization-nvgre]  
Sridharan, M., Greenberg, A., Wang, Y., Garg, P., Venkataramiah, N., Duda, K., Ganga, I., Lin, G., Pearson, M., Thaler, P., and C. Tumuluri, "NVGRE: Network Virtualization using Generic Routing Encapsulation", draft-sridharan-virtualization-nvgre-03 (work in progress), August 2013.

Authors' Addresses

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

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

Frank Xia  
Huawei USA  
Nanjing, China

Phone: +1 972-509-5599  
Email: xiayangsong@huawei.com



DHC Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: March 17, 2014

P. Patil  
Cisco  
M. Boucadair  
France Telecom  
D. Wing  
T. Reddy  
Cisco  
September 13, 2013

DHCPv6 Dynamic Reconfiguration  
draft-wing-dhc-dns-reconfigure-02

Abstract

This specification extends DHCPv6 so that a DHCPv6 Relay Agent can dynamically indicate end host connectivity to a DHCPv6 Server. This information is also triggered by any change in connectivity type provided to the host. The DHCPv6 server uses this information as an input to its decision-making about configuration parameters to be conveyed to that host.

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 March 17, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
3. Problem Statement: Focus on DNS Reconfiguration . . . . .	3
4. Host Connectivity Status Option . . . . .	4
5. DHCPv6 Relay Agent Behavior . . . . .	5
5.1. Relay Forward . . . . .	5
5.2. Reconfigure Request . . . . .	6
6. DHCPv6 Server Behavior . . . . .	6
6.1. Relay Forward . . . . .	6
6.2. Reconfigure Request . . . . .	6
7. Host Tracking . . . . .	7
8. Security Considerations . . . . .	7
9. IANA Considerations . . . . .	7
10. References . . . . .	7
10.1. Normative References . . . . .	8
10.2. Informative References . . . . .	8
Authors' Addresses . . . . .	9

## 1. Introduction

Some networks are expected to support IPv4-only, dual-stack, and IPv6-only hosts at the same time. Due to devices capabilities and available connectivity types, providing generic configuration from a DHCP server to connected hosts is sub-optimal in most cases, and may even break functionality in some cases. Network infrastructure is usually well equipped to be aware of single/dual-stack nature of hosts. The network can also track and detect transitions from single to dual-stack or vice-versa.

This specification describes a DHCPv6 extension for relay agents to indicate host characteristics pertaining to host connectivity to DHCPv6 servers. The information passed by a relay is generic and a DHCPv6 server can interpret and process this information to make a more informed decision on the configuration parameters that a client is to receive.

The DHCPv6 server can either be configured or have built-in logic to use this information as desired, which is outside the scope of this document.

Section 3 describes a typical problem that can be addressed using the mechanism described in this specification. A DHCPv6 server makes a decision on priority of DNS servers to be sent back to the client based on host connectivity characteristics provided by the relay agent.

While the host stack can be upgraded to send this information to the DHCPv6 server on its own, a generalized upgrade of all DHCPv6 client implementations on all operating systems is extremely difficult.

[DISCUSSION NOTE: A companion solution could be to define a container that can be used to return per-AF specific configuration parameters to the client. In such a scheme, the server blindly returns all pieces of configuration and it is up to the client to make use of appropriate set of parameters according to its available connectivity. This alternative assumes an update of dhcp client. This approach can be seen as complimentary to the one defined in this specification. The document will be updated to reflect consensus of the WG on whether the additional option is to be specified.]

## 2. 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].

Dual-Stack host: Denotes a host that is configured with both an IPv4 address and IPv6 prefix and is reachable using both IPv4 and IPv6 connectivity.

## 3. Problem Statement: Focus on DNS Reconfiguration

Default address selection rules specified in [RFC6724] prefers IPv6 over IPv4. If a dual-stack host is configured to use a DNS64 server [RFC6147], it will send its DNS queries to that DNS64 server which will synthesize a AAAA response if no A record is found. Thus, the dual-stack host will always use IPv6 if a DNS lookup was involved, even if IPv4 could have been used more optimally.

In some deployments, if NAT44 [RFC3022] and NAT64 [RFC6146] are deployed on the same network, it is preferable to use NAT44 over NAT64 because of scale, performance and application incompatibility issues (e.g., FTP) [RFC6384]. At the same time, native IPv6 can still be preferred over IPv4.

A DHCPv6 Relay Agent can observe host characteristics on a network to determine if a host is IPv4-only, dual-stack or IPv6-only and also

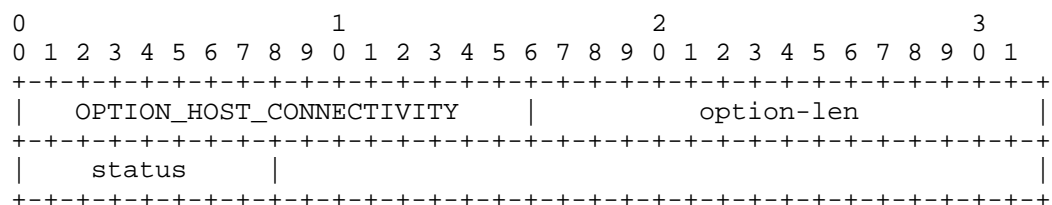


detect transitions from single to dual-stack or vice-versa. This information can be used by the DHCPv6 Relay Agent to influence the DHCPv6 Server to send appropriately prioritized DNS Servers to the client. The DHCPv6 server can implement the following based on connectivity information received from the relay agent.

- o IPv6-only transition to Dual-Stack: In case a host is IPv6-only, it is provided with a DNS64 server. When transitioning to dual-stack, an IPv4 DNS server is assigned as a consequence of obtaining an IPv4 Address. The DHCPv6 Relay Agent can detect this and send a RECONFIGURE\_REQUEST message [RFC6977] to the DHCPv6 Server indicating that the host needs to be provided with a regular DNS server. In lieu of this mechanism, the host would continue to use the DNS64 server until the host stack reinitializes.
- o Dual-Stack to IPv6-only: In case a host is dual-stack, it is provided with a regular DNS server followed by DNS64 server. When transitioning to IPv6-only, the DHCPv6 Relay Agent can detect this change and send a RECONFIGURE\_REQUEST message to the DHCPv6 server indicating that the host needs to be assigned a DNS64 server only. In lieu of this mechanism, the host would continue to use the regular DNS Server which is inaccessible and eventually time out to fail over to the DNS64 Server. The host will take additional time to fully initialize causing delays in connection.

#### 4. Host Connectivity Status Option

The option (Figure 1) includes an 8-bit status code that indicates specific host connectivity characteristics. The option can be included by a DHCPv6 Relay Agent in RELAY-FORW and RECONFIGURE-REQUEST.



option-code	OPTION_HOST_CONNECTIVITY (TBA).
option-len	1.
status	8-bit integer value carrying the connectivity status of a host. The following codes are defined:

Value	Name
-------	------

	1		IPv4_TO_DUAL_STACK	
	2		IPv6_TO_DUAL_STACK	
	3		DUAL_STACK_TO_IPv4	
	4		DUAL_STACK_TO_IPv6	
+-----+-----+-----+-----+				

Figure 1: Relay Agent Host Connectivity Option message format

- o IPv4\_TO\_DUAL\_STACK: Host is transitioning from IPv4-Only to Dual-Stack mode.
- o IPv6\_TO\_DUAL\_STACK: Host is transitioning from IPv6-Only to Dual-Stack mode.
- o DUAL\_STACK\_TO\_IPv4: Host is transitioning from Dual-Stack to IPv4-Only mode.
- o DUAL\_STACK\_TO\_IPv6: Host is transitioning from Dual-Stack to IPv6-Only mode.

## 5. DHCPv6 Relay Agent Behavior

DHCPv6 relay agents that implement this specification MUST be configurable for tracking host connectivity and inserting the OPTION\_HOST\_CONNECTIVITY option in RELAY-FORW and RECONFIGURE-REQUEST messages.

To be able to notify details of hosts' connectivity, a relay agent must be able to track host connectivity. A Relay Agent can detect host connectivity type using mechanisms discussed in Section 7. The Relay Agent then includes this information in the appropriate DHCPv6 message.

Relay agents need to maintain connectivity state of each host it can track. This ensures that notifications to the DHCPv6 server, especially DHCPv6 RECONFIGURE-REQUEST, are accurately sent when there is a change in status. If a relay agent loses state due to some reason (e.g., during restart events), it will build state again using the mechanisms described in Section 7 and then send appropriate notifications to the server. Such notifications are redundant and a DHCPv6 Server can choose to ignore such redundant notifications from the relay agent. Redundant notifications are also possible when relay agents are deployed in fault tolerant mode.

### 5.1. Relay Forward

DHCPv6 relay agents that implement this specification MAY include the option `OPTION_HOST_CONNECTIVITY` in the `RELAY_FORW` to indicate status of host connectivity.

## 5.2. Reconfigure Request

DHCPv6 relay agents that implement this specification MUST be configurable for sending the `RECONFIGURE_REQUEST` message. The relay agent generates a Reconfigure-Request [RFC6977] anytime status of host connectivity changes by including `OPTION_HOST_CONNECTIVITY` in the request.

## 6. DHCPv6 Server Behavior

A DHCPv6 Server that supports `OPTION_HOST_CONNECTIVITY` may either have specific configuration or built-in logic to process information available in the option and send configuration parameters in DHCPv6 responses. How the server consumes and acts on the information obtained in the option are outside the scope of this document.

The DHCPv6 server may use this connectivity information, if available, in addition to other relay agent option data, other options included in the DHCPv6 client messages, server configuration, and physical network topology information in order to assign appropriate configuration to the client.

The server MUST ignore the option if it doesn't recognize the status in the `OPTION_HOST_CONNECTIVITY` option. The server SHOULD maintain the latest status received from the relay agent. The server can use this state to match against subsequent notifications and only further process if there is change in status. A relay agent could, for reasons such as restart, fault-tolerant mode etc, send redundant notifications and matching of status at the server will avoid unnecessary processing and message exchanges.

### 6.1. Relay Forward

Upon receiving a `RELAY-FORW` message containing `OPTION_HOST_CONNECTIVITY`, the server can send appropriate configuration in the `RELAY-REPLY` response. The server MUST NOT return this option in a `RELAY-REPLY` message.

### 6.2. Reconfigure Request

Upon receiving a `RECONIFURE-REQUEST` message containing an `OPTION_HOST_CONNECTIVITY` option, the server MUST follow the mechanism described in [RFC6977] to create and send Reconfigure message. The server MUST NOT return this option in a `RECONFIGURE-REPLY` message.

## 7. Host Tracking

Relay Agents can actively keep track of all IPv4/IPv6 addresses and associated lease times assigned to hosts via the respective DHCP servers. Relay Agents can therefore detect transitions from single to dual-stack and vice-versa efficiently. In addition to this technique, relay agents closest to the client can detect transitions using snooping mechanisms. Network devices today use mechanisms such as ARP and NDP snooping (bindings learnt by snooping all NDP traffic, NS, NA, RS, RA) to determine host characteristics such as IPv4/IPv6 - MAC - DUID bindings. IPv4/IPv6 and MAC counters are also used to determine host liveliness.

First hop devices that implement first hop security also track IP address bindings and determine binding updates such as temporary addresses, deprecated addresses, etc. Existing work such as [I-D.ietf-savi-dhcp] and [I-D.levy-abegnoli-savi-plbt] also aim to active current host bindings, all of which can be leveraged to track host addresses.

These mechanisms help determine if a particular IP address family is inactive, has reverted to using a single stack even though it initially had dual-stack capabilities and detect active dual-stack usage after long periods of single-stack activity.

Other techniques to track host connectivity can be envisaged. It is out of scope of this document to provide an exhaustive list of host tracking techniques.

## 8. Security Considerations

This document describes an application of the mechanism specified in [RFC6977]. Host tracking mechanisms MUST be reliable. If a relay is compromised, it may be used to force an uncompromised server abuse clients by triggering repetitive reconfigurations. Security considerations described in [RFC6977] are applicable to this mechanism.

## 9. IANA Considerations

IANA is requested to assign the following new DHCPv6 Option Code in the registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>:

- o OPTION\_HOST\_CONNECTIVITY

## 10. References

## 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6384] van Beijnum, I., "An FTP Application Layer Gateway (ALG) for IPv6-to-IPv4 Translation", RFC 6384, October 2011.
- [RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.
- [RFC6977] Boucadair, M. and X. Pournard, "Triggering DHCPv6 Reconfiguration from Relay Agents", RFC 6977, July 2013.

## 10.2. Informative References

- [I-D.ietf-savi-dhcp] Bi, J., Wu, J., Yao, G., and F. Baker, "SAVI Solution for DHCP", draft-ietf-savi-dhcp-18 (work in progress), June 2013.
- [I-D.levy-abegnoli-savi-plbt] Levy-Abegnoli, E., "Preference Level based Binding Table", draft-levy-abegnoli-savi-plbt-02 (work in progress), March 2010.
- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3646] Droms, R., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, December 2003.
- [RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.
- [RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, April 2011.

Authors' Addresses

Prashanth Patil  
Cisco Systems, Inc.  
Bangalore  
India

Email: [praspati@cisco.com](mailto:praspati@cisco.com)

Mohamed Boucadair  
France Telecom  
Rennes 35000  
France

Email: [mohamed.boucadair@orange.com](mailto:mohamed.boucadair@orange.com)

Dan Wing  
Cisco Systems, Inc.  
170 West Tasman Drive  
San Jose, California 95134  
USA

Email: [dwing@cisco.com](mailto:dwing@cisco.com)

Tirumaleswar Reddy  
Cisco Systems, Inc.  
Cessna Business Park, Varthur Hobli  
Sarjapur Marathalli Outer Ring Road  
Bangalore, Karnataka 560103  
India

Email: [tiredddy@cisco.com](mailto:tiredddy@cisco.com)

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 27, 2015

L. Xue  
D. Guo  
Huawei  
October 24, 2014

Dynamic Stateless GRE Tunnel  
draft-xue-dhc-dynamic-gre-03

Abstract

Generic Routing Encapsulation (GRE) is regarded as a popular encapsulation tunnel technology. When a node tries to encapsulate the user traffic in GRE, it needs the IP address of the destination node which decapsulates the GRE packets. In practice, the GRE tunnel destination IP address may be manually configured. This configuration may introduce efficiency issues for operators. This work proposes an approach to configure the GRE information dynamically.

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 [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 April 27, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	2
3. GRE Use Case - WLAN Network . . . . .	3
4. DHCP Options Definition . . . . .	4
4.1. GRE Discovery DHCPv4 Option . . . . .	4
4.2. GRE Information DHCPv4 Option . . . . .	5
4.3. GRE Discovery DHCPv6 Option . . . . .	5
4.4. GRE Information DHCPv6 Option . . . . .	6
5. Dynamic GRE Tunnel . . . . .	6
6. IANA Considerations . . . . .	8
7. References . . . . .	8
7.1. Normative References . . . . .	8
7.2. Informative References . . . . .	8
Authors' Addresses . . . . .	9

## 1. Introduction

Generic Routing Encapsulation (GRE, see [RFC1701] and [RFC2784]) is widely deployed in the operators' networks. When a node tries to encapsulate the user traffic in a GRE tunnel, it needs the IP address of the destination node which can decapsulate the GRE packets. In practice, the manual configuration happens on the nodes. This may introduce efficiency issues for operators. As an example, if GRE tunneling is used in the access network, there may a large amount of configuration needed at the access side. This specification introduces a use case requiring the deployment of a large amount of GRE tunnels, which motivates a dynamic approach. The specification proposes a solution to enable the dynamic discovery of the GRE decapsulation device through use of a Dynamic Host Configuration Protocol (DHCP) option.

## 2. Terminology

The following terms are used in this document:



**Access Controller (AC):** The network entity that provides Wireless Termination Point (WTP) access to the network infrastructure in the data plane, control plane, management plane, or a combination therein.

**Customer Premises Equipment (CPE):** The box that a provider may distribute to the customers. When CPE is using DHCP to obtain network address, CPE is acting as "DHCP Client".

**Wireless Termination Point (WTP):** The physical or logical network entity that contains an RF antenna and wireless physical layer (PHY) to transmit and receive station traffic for wireless access networks.

### 3. GRE Use Case - WLAN Network

Wireless Local Area Network (WLAN) has emerged as an important access technology for service operators. A typical WLAN network contains a large number of WTPs, centrally managed and controlled by the Access Controller (AC). It is desirable to distribute customer data frames to an endpoint through an Access Router (AR) different from the AC. GRE encapsulation can be used between a WTP and an AR as one of the optional tunneling technologies shown in [I-D.ietf-opsawg-capwap-alt-tunnel].

An illustration of a WLAN network is shown in Figure 1. In order for a WTP to encapsulate the user traffic in a GRE tunnel, it needs to know the Access Router (AR) IP address. This IP address is usually deployed on WTPs manually, which may introduce efficiency issues for operators. An AC may dynamically configure the WTP with the AR address via extended CAPWAP message elements (see [I-D.ietf-opsawg-capwap-alt-tunnel]). However, this approach does not apply to a WLAN network where the CAPWAP protocol is not deployed, as the network shown in Figure 2. In fact, it is quite common for operators to have their own private control plane between the WTP and the AC rather than CAPWAP. Moreover, there are also WLAN deployments without AC, as in the FAT WTPs scenario (see Figure 3). A general approach to resolve this problem is desirable.

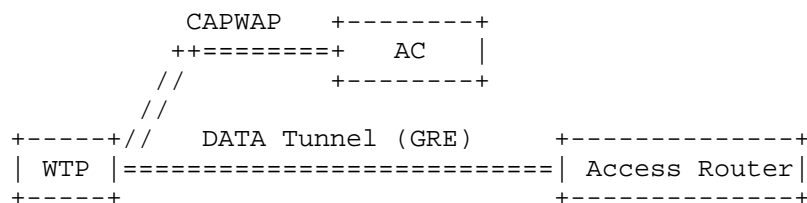


Figure 1: GRE Use Case - WLAN Network 1

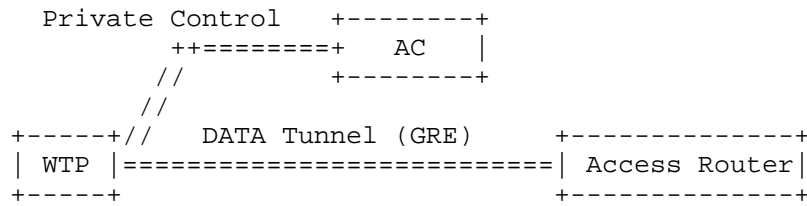


Figure 2: GRE Use Case - WLAN Network 2

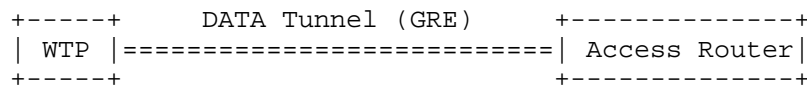


Figure 3: GRE Use Case - WLAN Network 3

#### 4. DHCP Options Definition

##### 4.1. GRE Discovery DHCPv4 Option

The GRE Discovery DHCPv4 option provides to a GRE encapsulator a list of one or more IPv4 addresses of a GRE decapsulator. According to [RFC2131], the GRE Discovery DHCPv4 Option is structured as shown in Figure 4.

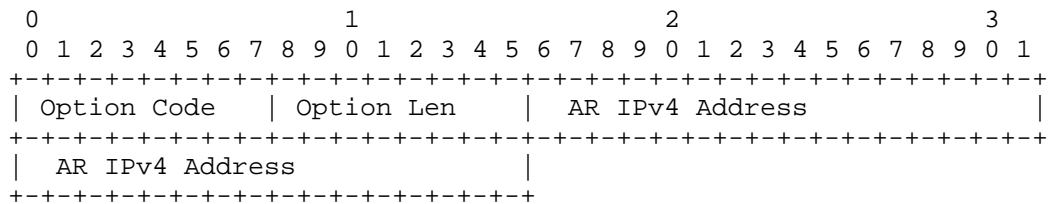


Figure 4: GRE Discovery DHCPv4 Option

Code: TBD

Len: 4

AR IPv4 Address: AR IPv4 address, an endpoint of GRE tunnel. More than one AR IPv4 addresses may be provided for redundancy reasons. The default priority of the listed AR IPv4 addresses may be from highest to lowest.

#### 4.2. GRE Information DHCPv4 Option

The GRE Information DHCPv4 option provides a list of the GRE information as defined in and [RFC2784][RFC2890]. The GRE information may include the key.

According to [RFC2131], the GRE Information DHCPv4 Option is structured as shown in Figure 5.

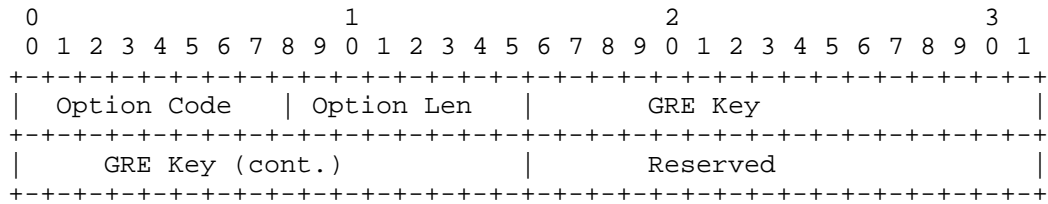


Figure 5: GRE Information DHCPv4 Option

Code: TBD

Len: 6

GRE Key: The Key field contains a four octet number which is inserted by the GRE encapsulator according to [RFC2890].

Reserved: This field is reserved for future use. These bits MUST be sent as zero and MUST be ignored on receipt.

#### 4.3. GRE Discovery DHCPv6 Option

The GRE Discovery DHCPv6 option provides to a GRE encapsulator a list of one or more IPv6 addresses of a GRE decapsulator. According to [RFC7227], the GRE Discovery DHCPv6 Option is structured as shown in Figure 6.

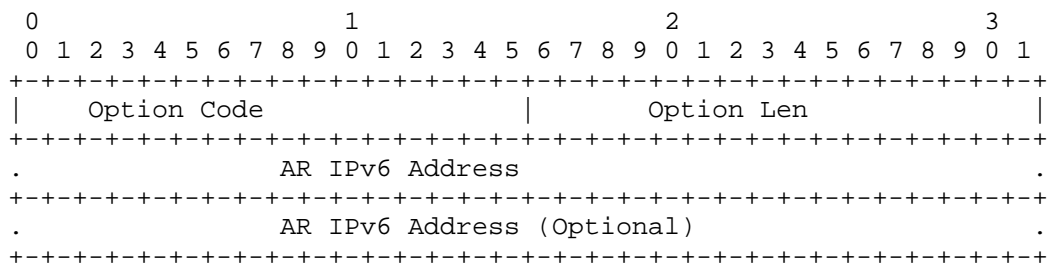


Figure 6: DHCPv6 GRE Discovery Option

Code: TBD

Len: >=16

AR IPv6 Address: AR IPv6 address, an endpoint of GRE tunnel. More than one AR IPv6 addresses may be provided for redundancy reasons. The default priority of the listed AR IPv6 addresses may be from highest to the lowest.

#### 4.4. GRE Information DHCPv6 Option

The GRE Information DHCPv6 option provides a list of the GRE information as defined in and [RFC2784][RFC2890]. The GRE information may include the key.

According to [RFC7227], the GRE Information DHCPv6 Option is structured as shown in Figure 7.

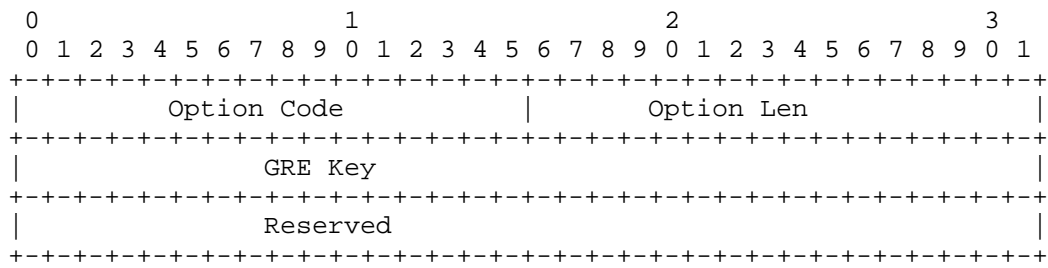


Figure 7: GRE Information DHCPv6 Option

Code: TBD

Len: 8

GRE Key: The Key field contains a four octet number which is inserted by the GRE encapsulator according to [RFC2890].

Reserved: This field is reserved for future use. These bits MUST be sent as zero and MUST be ignored on receipt.

#### 5. Dynamic GRE Tunnel

The DHCP options defined in Section 4 enable an automated way to inform the GRE encapsulator with the GRE destination IP address. Additionally, some other GRE tunnel information may be provided. In this way, a GRE tunnel can be setup dynamically.

Figure 8 illustrates the procedure to set up a dynamic GRE tunnel in the network.

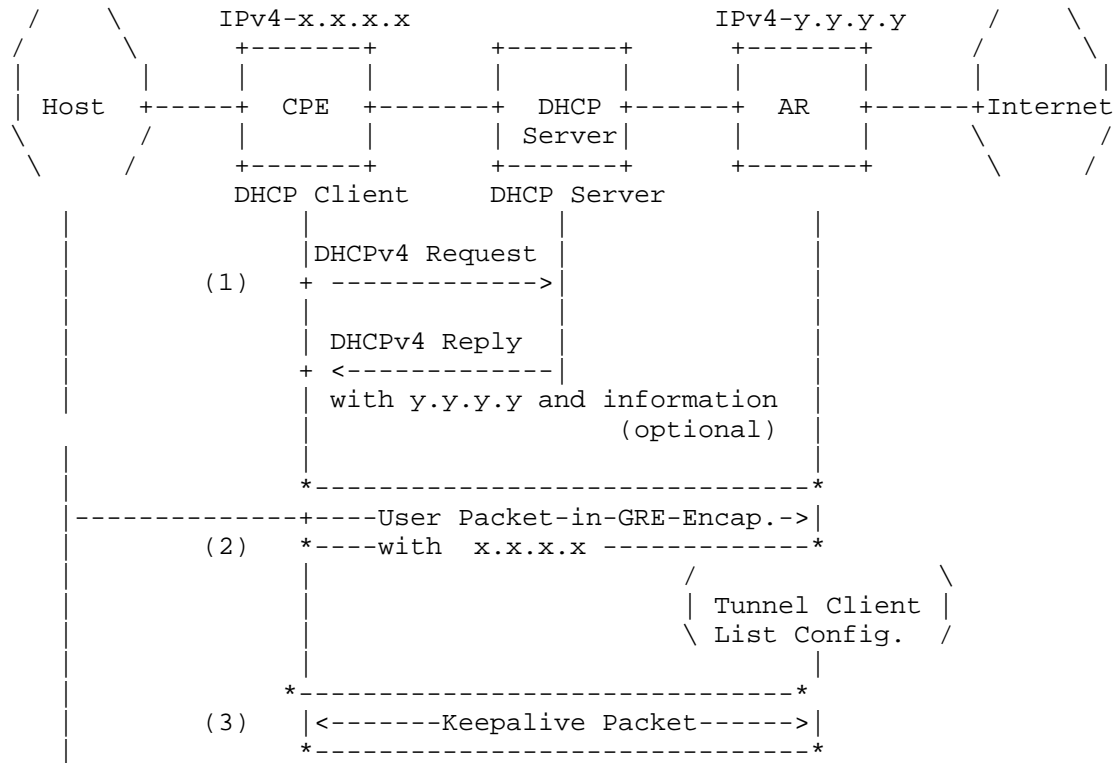


Figure 8: Dynamic GRE Tunnel

The steps to set up a GRE tunnel between the CPE and the AR are as follows:

1. The CPE, as one endpoint of GRE tunnel, sends the DHCP request message to the DHCP server to acquire the AR access. The GRE Discovery DHCP Option should be included, with AR IPv4 address set to zero. When the DHCP server receives this request, it replies to the CPE the DHCP Reply message, containing the AR address and the tunnel information if needed.
2. The CPE can encapsulate the upstream packets from the hosts within GRE packets. Generally, upstream packets are either data packets or control packets. When the AR gets an encapsulated GRE packet, the AR checks whether there is an existing GRE tunnel

with the CPE. If this is a new endpoint without GRE record, the AR should add this CPE into the tunnel client list.

3. A keepalive mechanism may be required for a GRE tunnel between the CPE and the AR. If there is neither keepalive packet nor data packet, when a keepalive timer expires, the AR or the CPE will tear down the tunnel and release resources.

## 6. IANA Considerations

TBD

## 7. References

### 7.1. Normative References

- [RFC1701] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 1701, October 1994.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000.
- [RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, September 2000.
- [RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", BCP 187, RFC 7227, May 2014.

### 7.2. Informative References

- [I-D.ietf-opsawg-capwap-alt-tunnel]  
Zhang, R., Cao, Z., Deng, H., Pazhyannur, R., Gundavelli, S., and L. Xue, "Alternate Tunnel Encapsulation for Data Frames in CAPWAP", draft-ietf-opsawg-capwap-alt-tunnel-03 (work in progress), September 2014.

Authors' Addresses

Li Xue  
Huawei  
No. 156 Beiqing Rd. Z-park, Shi-Chuang-Ke-Ji-Shi-Fan-Yuan  
Beijing, Haidian District 100095  
China

Email: xueli@huawei.com

Dayong Guo  
Huawei  
No. 156 Beiqing Rd. Z-park, Shi-Chuang-Ke-Ji-Shi-Fan-Yuan  
Beijing, Haidian District 100095  
China

Email: guoseu@huawei.com