

Dynamic Host Configuration (DHC)  
Internet-Draft  
Obsoletes: 3315,3633,3736,7083 (if  
approved)  
Intended status: Standards Track  
Expires: August 26, 2015

T. Mrugalski, Ed.  
M. Siodelski  
ISC  
B. Volz, Ed.  
A. Yourtchenko  
Cisco  
M. Richardson  
SSW  
S. Jiang  
Huawei  
T. Lemon  
Nominum  
February 22, 2015

Dynamic Host Configuration Protocol for IPv6 (DHCPv6) bis  
draft-dhcwg-dhc-rfc3315bis-04

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCP) enables DHCP servers to pass configuration parameters such as IPv6 network addresses to IPv6 nodes. It offers the capability of automatic allocation of reusable network addresses and additional configuration flexibility. This protocol is a stateful counterpart to "IPv6 Stateless Address Autoconfiguration" (RFC 4862), and can be used separately or concurrently with the latter to obtain configuration parameters.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 26, 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 and Overview . . . . .	6
1.1. Protocols and Addressing . . . . .	7
1.2. Client-server Exchanges Involving Two Messages . . . . .	7
1.3. Client-server Exchanges Involving Four Messages . . . . .	8
2. Requirements . . . . .	8
3. Background . . . . .	9
4. Terminology . . . . .	10
4.1. IPv6 Terminology . . . . .	10
4.2. DHCP Terminology . . . . .	11
5. Operational Models . . . . .	14
5.1. Stateless DHCP . . . . .	14
5.2. DHCP for Non-Temporary Address Assignment . . . . .	15
5.3. DHCP for Prefix Delegation . . . . .	15
5.4. DHCP for Customer Edge Routers . . . . .	18
5.5. DHCP for Temporary Addresses . . . . .	18
6. DHCP Constants . . . . .	18
6.1. Multicast Addresses . . . . .	18
6.2. UDP Ports . . . . .	19
6.3. DHCP Message Types . . . . .	19

6.4.	Status Codes . . . . .	21
6.5.	Transmission and Retransmission Parameters . . . . .	21
6.6.	Representation of time values and "Infinity" as a time value . . . . .	22
7.	Client/Server Message Formats . . . . .	22
8.	Relay Agent/Server Message Formats . . . . .	23
8.1.	Relay-forward Message . . . . .	24
8.2.	Relay-reply Message . . . . .	25
9.	Representation and Use of Domain Names . . . . .	25
10.	DHCP Unique Identifier (DUID) . . . . .	25
10.1.	DUID Contents . . . . .	26
10.2.	DUID Based on Link-layer Address Plus Time, DUID-LLT . .	26
10.3.	DUID Assigned by Vendor Based on Enterprise Number, DUID-EN . . . . .	28
10.4.	DUID Based on Link-layer Address, DUID-LL . . . . .	29
11.	Identity Association . . . . .	30
11.1.	Identity Associations for Address Assignment . . . . .	30
11.2.	Identity Associations for Prefix Delegation . . . . .	30
12.	Selecting Addresses for Assignment to an IA . . . . .	31
13.	Management of Temporary Addresses . . . . .	32
14.	Transmission of Messages by a Client . . . . .	33
14.1.	Rate Limiting . . . . .	33
15.	Reliability of Client Initiated Message Exchanges . . . . .	34
16.	Message Validation . . . . .	35
16.1.	Use of Transaction IDs . . . . .	36
16.2.	Solicit Message . . . . .	36
16.3.	Advertise Message . . . . .	36
16.4.	Request Message . . . . .	37
16.5.	Confirm Message . . . . .	37
16.6.	Renew Message . . . . .	37
16.7.	Rebind Message . . . . .	37
16.8.	Decline Messages . . . . .	38
16.9.	Release Message . . . . .	38
16.10.	Reply Message . . . . .	38
16.11.	Reconfigure Message . . . . .	39
16.12.	Information-request Message . . . . .	39
16.13.	Relay-forward Message . . . . .	39
16.14.	Relay-reply Message . . . . .	40
17.	Client Source Address and Interface Selection . . . . .	40
17.1.	Address Assignment . . . . .	40
17.2.	Prefix Delegation . . . . .	40
18.	DHCP Server Solicitation . . . . .	41
18.1.	Client Behavior . . . . .	41
18.1.1.	Creation of Solicit Messages . . . . .	41
18.1.2.	Transmission of Solicit Messages . . . . .	42
18.1.3.	Receipt of Advertise Messages . . . . .	43
18.1.4.	Receipt of Reply Message . . . . .	44
18.2.	Server Behavior . . . . .	45

18.2.1.	Receipt of Solicit Messages . . . . .	45
18.2.2.	Creation and Transmission of Advertise Messages . .	45
18.2.3.	Creation and Transmission of Reply Messages . . . .	47
18.3.	Client behavior for Prefix Delegation . . . . .	47
18.4.	Server Behavior for Prefix Delegation . . . . .	48
19.	DHCP Client-Initiated Configuration Exchange . . . . .	48
19.1.	Client Behavior . . . . .	49
19.1.1.	Creation and Transmission of Request Messages . . .	49
19.1.2.	Creation and Transmission of Confirm Messages . . .	50
19.1.3.	Creation and Transmission of Renew Messages . . . .	52
19.1.4.	Creation and Transmission of Rebind Messages . . . .	53
19.1.5.	Creation and Transmission of Information-request Messages . . . . .	54
19.1.6.	Creation and Transmission of Release Messages . . .	55
19.1.7.	Creation and Transmission of Decline Messages . . .	56
19.1.8.	Receipt of Reply Messages . . . . .	57
19.2.	Server Behavior . . . . .	59
19.2.1.	Receipt of Request Messages . . . . .	59
19.2.2.	Receipt of Confirm Messages . . . . .	60
19.2.3.	Receipt of Renew Messages . . . . .	61
19.2.4.	Receipt of Rebind Messages . . . . .	62
19.2.5.	Receipt of Information-request Messages . . . . .	62
19.2.6.	Receipt of Release Messages . . . . .	63
19.2.7.	Receipt of Decline Messages . . . . .	64
19.2.8.	Transmission of Reply Messages . . . . .	64
19.3.	Requesting Router Behavior for Prefix Delegation . . . .	65
19.4.	Delegating Router Behavior for Prefix Delegation . . . .	66
20.	DHCP Server-Initiated Configuration Exchange . . . . .	67
20.1.	Server Behavior . . . . .	68
20.1.1.	Creation and Transmission of Reconfigure Messages .	68
20.1.2.	Time Out and Retransmission of Reconfigure Messages	69
20.2.	Receipt of Renew or Rebind Messages . . . . .	69
20.3.	Receipt of Information-request Messages . . . . .	69
20.4.	Client Behavior . . . . .	70
20.4.1.	Receipt of Reconfigure Messages . . . . .	70
20.4.2.	Creation and Transmission of Renew or Rebind Messages . . . . .	71
20.4.3.	Creation and Transmission of Information-request Messages . . . . .	71
20.4.4.	Time Out and Retransmission of Renew, Rebind or Information-request Messages . . . . .	71
20.4.5.	Receipt of Reply Messages . . . . .	71
20.5.	Prefix Delegation Reconfiguration . . . . .	72
20.5.1.	Delegating Router Behavior . . . . .	72
20.5.2.	Requesting Router Behavior . . . . .	72
21.	Relay Agent Behavior . . . . .	72
21.1.	Relaying a Client Message or a Relay-forward Message . .	72
21.1.1.	Relaying a Message from a Client . . . . .	73

21.1.2.	Relaying a Message from a Relay Agent . . . . .	73
21.1.3.	Relay Agent Behavior with Prefix Delegation . . . . .	74
21.2.	Relaying a Relay-reply Message . . . . .	74
21.3.	Construction of Relay-reply Messages . . . . .	74
22.	Authentication of DHCP Messages . . . . .	75
22.1.	Security of Messages Sent Between Servers and Relay Agents . . . . .	76
22.2.	Summary of DHCP Authentication . . . . .	77
22.3.	Replay Detection . . . . .	77
22.4.	Delayed Authentication Protocol . . . . .	78
22.4.1.	Use of the Authentication Option in the Delayed Authentication Protocol . . . . .	78
22.4.2.	Message Validation . . . . .	80
22.4.3.	Key Utilization . . . . .	80
22.4.4.	Client Considerations for Delayed Authentication Protocol . . . . .	80
22.4.4.1.	Sending Solicit Messages . . . . .	80
22.4.4.2.	Receiving Advertise Messages . . . . .	81
22.4.4.3.	Sending Request, Confirm, Renew, Rebind, Decline or Release Messages . . . . .	81
22.4.4.4.	Sending Information-request Messages . . . . .	82
22.4.4.5.	Receiving Reply Messages . . . . .	82
22.4.4.6.	Receiving Reconfigure Messages . . . . .	82
22.4.5.	Server Considerations for Delayed Authentication Protocol . . . . .	82
22.4.5.1.	Receiving Solicit Messages and Sending Advertise Messages . . . . .	82
22.4.5.2.	Receiving Request, Confirm, Renew, Rebind or Release Messages and Sending Reply Messages . . . . .	83
22.5.	Reconfigure Key Authentication Protocol . . . . .	83
22.5.1.	Use of the Authentication Option in the Reconfigure Key Authentication Protocol . . . . .	83
22.5.2.	Server considerations for Reconfigure Key protocol . . . . .	84
22.5.3.	Client considerations for Reconfigure Key protocol . . . . .	85
23.	DHCP Options . . . . .	85
23.1.	Format of DHCP Options . . . . .	86
23.2.	Client Identifier Option . . . . .	86
23.3.	Server Identifier Option . . . . .	87
23.4.	Identity Association for Non-temporary Addresses Option . . . . .	88
23.5.	Identity Association for Temporary Addresses Option . . . . .	90
23.6.	IA Address Option . . . . .	92
23.7.	Option Request Option . . . . .	93
23.8.	Preference Option . . . . .	94
23.9.	Elapsed Time Option . . . . .	95
23.10.	Relay Message Option . . . . .	95
23.11.	Authentication Option . . . . .	96
23.12.	Server Unicast Option . . . . .	97
23.13.	Status Code Option . . . . .	98

23.14. Rapid Commit Option . . . . .	100
23.15. User Class Option . . . . .	101
23.16. Vendor Class Option . . . . .	102
23.17. Vendor-specific Information Option . . . . .	104
23.18. Interface-Id Option . . . . .	106
23.19. Reconfigure Message Option . . . . .	107
23.20. Reconfigure Accept Option . . . . .	107
23.21. Identity Association for Prefix Delegation Option . . . . .	108
23.22. IA Prefix Option . . . . .	110
23.23. SOL_MAX_RT Option . . . . .	111
23.24. INF_MAX_RT Option . . . . .	112
24. Security Considerations . . . . .	113
25. IANA Considerations . . . . .	116
26. Acknowledgments . . . . .	116
27. References . . . . .	117
27.1. Normative References . . . . .	117
27.2. Informative References . . . . .	119
Appendix A. Changes since RFC3315 . . . . .	120
Appendix B. Changes since RFC3633 . . . . .	123
Appendix C. Appearance of Options in Message Types . . . . .	123
Appendix D. Appearance of Options in the Options Field of DHCP Options . . . . .	124
Authors' Addresses . . . . .	125

## 1. Introduction and Overview

This document describes DHCP for IPv6 (DHCP), a client/server protocol that provides managed configuration of devices.

DHCP can provide a device with addresses assigned by a DHCP server and other configuration information, which are carried in options. DHCP can be extended through the definition of new options to carry configuration information not specified in this document.

DHCP is the "stateful address autoconfiguration protocol" and the "stateful autoconfiguration protocol" referred to in "IPv6 Stateless Address Autoconfiguration" [RFC4862].

This document also provides a mechanism for automated delegation of IPv6 prefixes using DHCP. Through this mechanism, a delegating router can delegate prefixes to requesting routers.

The operational models and relevant configuration information for DHCPv4 [RFC2132][RFC2131] and DHCPv6 are sufficiently different that integration between the two services is not included in this document. [RFC3315] suggested that future work might be to extend DHCPv6 to carry IPv4 address and configuration information. However, the current consensus of the IETF is that DHCPv4 should be used

rather than DHCPv6 when conveying IPv4 configuration information to nodes. [RFC7341] describes a transport mechanism to carry DHCPv4 messages using the DHCPv6 protocol for the dynamic provisioning of IPv4 address and configuration information across IPv6-only networks.

The remainder of this introduction summarizes DHCP, explaining the message exchange mechanisms and example message flows. The message flows in Section 1.2 and Section 1.3 are intended as illustrations of DHCP operation rather than an exhaustive list of all possible client-server interactions. Section 5 provides an overview of common operational models. Section 18, Section 19, and Section 20 explain client and server operation in detail.

### 1.1. Protocols and Addressing

Clients and servers exchange DHCP messages using UDP [RFC0768]. The client uses a link-local address or addresses determined through other mechanisms for transmitting and receiving DHCP messages.

A DHCP client sends most messages using a reserved, link-scoped multicast destination address so that the client need not be configured with the address or addresses of DHCP servers.

To allow a DHCP client to send a message to a DHCP server that is not attached to the same link, a DHCP relay agent on the client's link will relay messages between the client and server. The operation of the relay agent is transparent to the client and the discussion of message exchanges in the remainder of this section will omit the description of message relaying by relay agents.

Once the client has determined the address of a server, it may under some circumstances send messages directly to the server using unicast.

### 1.2. Client-server Exchanges Involving Two Messages

When a DHCP client does not need to have a DHCP server assign it IP addresses, the client can obtain configuration information such as a list of available DNS servers [RFC3646] or NTP servers [RFC4075] through a single message and reply exchanged with a DHCP server. To obtain configuration information the client first sends an Information-request message to the All\_DHCP\_Relay\_Agents\_and\_Servers multicast address. Servers respond with a Reply message containing the configuration information for the client.

This message exchange assumes that the client requires only configuration information and does not require the assignment of any IPv6 addresses.

When a server has IPv6 addresses and other configuration information committed to a client, the client and server may be able to complete the exchange using only two messages, instead of four messages as described in the next section. In this case, the client sends a Solicit message to the All\_DHCP\_Relay\_Agents\_and\_Servers requesting the assignment of addresses and other configuration information. This message includes an indication that the client is willing to accept an immediate Reply message from the server. The server that is willing to commit the assignment of addresses to the client immediately responds with a Reply message. The configuration information and the addresses in the Reply message are then immediately available for use by the client.

Each address assigned to the client has associated preferred and valid lifetimes specified by the server. To request an extension of the lifetimes assigned to an address, the client sends a Renew message to the server. The server sends a Reply message to the client with the new lifetimes, allowing the client to continue to use the address without interruption.

### 1.3. Client-server Exchanges Involving Four Messages

To request the assignment of one or more IPv6 addresses, a client first locates a DHCP server and then requests the assignment of addresses and other configuration information from the server. The client sends a Solicit message to the All\_DHCP\_Relay\_Agents\_and\_Servers address to find available DHCP servers. Any server that can meet the client's requirements responds with an Advertise message. The client then chooses one of the servers and sends a Request message to the server asking for confirmed assignment of addresses and other configuration information. The server responds with a Reply message that contains the confirmed addresses and configuration.

As described in the previous section, the client sends a Renew message to the server to extend the lifetimes associated with its addresses, allowing the client to continue to use those addresses without interruption.

## 2. Requirements

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC2119].

This document also makes use of internal conceptual variables to describe protocol behavior and external variables that an implementation must allow system administrators to change. The



specific variable names, how their values change, and how their settings influence protocol behavior are provided to demonstrate protocol behavior. An implementation is not required to have them in the exact form described here, so long as its external behavior is consistent with that described in this document.

### 3. Background

The IPv6 Specification provides the base architecture and design of IPv6. Related work in IPv6 that would best serve an implementor to study includes the IPv6 Specification [RFC2460], the IPv6 Addressing Architecture [RFC4291], IPv6 Stateless Address Autoconfiguration [RFC4862], IPv6 Neighbor Discovery Processing [RFC4861], and Dynamic Updates to DNS [RFC2136]. These specifications enable DHCP to build upon the IPv6 work to provide both robust stateful autoconfiguration and autoregistration of DNS Host Names.

The IPv6 Addressing Architecture specification [RFC4291] defines the address scope that can be used in an IPv6 implementation, and the various configuration architecture guidelines for network designers of the IPv6 address space. Two advantages of IPv6 are that support for multicast is required and nodes can create link-local addresses during initialization. The availability of these features means that a client can use its link-local address and a well-known multicast address to discover and communicate with DHCP servers or relay agents on its link.

IPv6 Stateless Address Autoconfiguration [RFC4862] specifies procedures by which a node may autoconfigure addresses based on router advertisements [RFC4861], and the use of a valid lifetime to support renumbering of addresses on the Internet. In addition, the protocol interaction by which a node begins stateless or stateful autoconfiguration is specified. DHCP is one vehicle to perform stateful autoconfiguration. Compatibility with stateless address autoconfiguration is a design requirement of DHCP.

IPv6 Neighbor Discovery [RFC4861] is the node discovery protocol in IPv6 which replaces and enhances functions of ARP [RFC0826]. To understand IPv6 and stateless address autoconfiguration, it is strongly recommended that implementors understand IPv6 Neighbor Discovery.

Dynamic Updates to DNS [RFC2136] is a specification that supports the dynamic update of DNS records for both IPv4 and IPv6. DHCP can use the dynamic updates to DNS to integrate addresses and name space to not only support autoconfiguration, but also autoregistration in IPv6.

## 4. Terminology

This section defines terminology specific to IPv6 and DHCP used in this document.

### 4.1. IPv6 Terminology

IPv6 terminology relevant to this specification from the IPv6 Protocol [RFC2460], IPv6 Addressing Architecture [RFC4291], and IPv6 Stateless Address Autoconfiguration [RFC4862] is included below.

address	An IP layer identifier for an interface or a set of interfaces.
host	Any node that is not a router.
IP	Internet Protocol Version 6 (IPv6). The terms IPv4 and IPv6 are used only in contexts where it is necessary to avoid ambiguity.
interface	A node's attachment to a link.
link	A communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IP. Examples are Ethernet (simple or bridged); Token Ring; PPP links, X.25, Frame Relay, or ATM networks; and Internet (or higher) layer "tunnels", such as tunnels over IPv4 or IPv6 itself.
link-layer identifier	A link-layer identifier for an interface. Examples include IEEE 802 addresses for Ethernet or Token Ring network interfaces, and E.164 addresses for ISDN links.
link-local address	An IPv6 address having a link-only scope, indicated by having the prefix (FE80::/10), that can be used to reach neighboring nodes attached to the same link. Every interface has a link-local address.
multicast address	An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

neighbor	A node attached to the same link.
node	A device that implements IP.
packet	An IP header plus payload.
prefix	The initial bits of an address, or a set of IP addresses that share the same initial bits.
prefix length	The number of bits in a prefix.
router	A node that forwards IP packets not explicitly addressed to itself.
unicast address	An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.

#### 4.2. DHCP Terminology

Terminology specific to DHCP can be found below.

allocatable resource	(or resource). It is an address, a prefix or any other allocatable resource that may be defined in the future. Currently there are three defined allocatable resources: non-temporary addresses, temporary addresses and delegated prefixes.
appropriate to the link	An address is "appropriate to the link" when the address is consistent with the DHCP server's knowledge of the network topology, prefix assignment and address assignment policies.
binding	A binding (or, client binding) is a group of server data records containing the information the server has about the addresses in an IA or configuration information explicitly assigned to the client. Configuration information that has been returned to a client through a policy - for example, the information returned to all clients on the same link - does not require a binding. A binding containing information about an IA is indexed by the

tuple <DUID, IA-type, IAID> (where IA-type is the type of address in the IA; for example, temporary). A binding containing configuration information for a client is indexed by <DUID>.

configuration parameter	An element of the configuration information set on the server and delivered to the client using DHCP. Such parameters may be used to carry information to be used by a node to configure its network subsystem and enable communication on a link or internetwork, for example.
delegating router:	The router that acts as a DHCP server, and is responding to the prefix request.
DHCP	Dynamic Host Configuration Protocol for IPv6. The terms DHCPv4 and DHCPv6 are used only in contexts where it is necessary to avoid ambiguity.
DHCP client (or client)	A node that initiates requests on a link to obtain configuration parameters from one or more DHCP servers. Depending on the purpose of the client, it may feature the requesting router functionality, if it supports prefix delegation.
DHCP domain	A set of links managed by DHCP and operated by a single administrative entity.
DHCP realm	A name used to identify the DHCP administrative domain from which a DHCP authentication key was selected.
DHCP relay agent (or relay agent)	A node that acts as an intermediary to deliver DHCP messages between clients and servers. In certain configurations there may be more than one relay agent between clients and servers, so a relay agent may send DHCP messages to another relay agent.
DHCP server (or server)	A node that responds to requests from clients, and may or may not be on the same link as the client(s). Depending on its capabilities, it may also feature the

functionality of delegating router, if it supports prefix delegation.

DUID	A DHCP Unique IDentifier for a DHCP participant; each DHCP client and server has exactly one DUID. See Section 10 for details of the ways in which a DUID may be constructed.
IA	Identity Association: A collection of allocatable resources assigned to a client. Each IA has an associated IAID. A client may have more than one IA assigned to it; for example, one for each of its interfaces. Each IA holds one type of address; for example, an identity association for temporary addresses (IA_TA) holds temporary addresses (see "identity association for temporary addresses") and identity association for prefix delegation (IA_PD) holds delegated prefixes. Throughout this document, "IA" is used to refer to an identity association without identifying the type of allocatable resources in the IA. At the time of writing this document, there are 3 IA types defined: IA_NA, IA_TA and IA_PD. New IA types may be defined in the future.
IAID	Identity Association IDentifier: An identifier for an IA, chosen by the client. Each IA has an IAID, which is chosen to be unique among IAIDs for IAs of a specific type, belonging to that client.
IA_NA	Identity association for Non-temporary Addresses: An IA that carries assigned addresses that are not temporary addresses (see "identity association for temporary addresses")
IA_TA	Identity Association for Temporary Addresses: An IA that carries temporary addresses (see [RFC4941]).
IA_PD	Identity Association for Prefix Delegation: A collection of prefixes assigned to the requesting router. Each IA_PD has an

	associated IAID. A requesting router may have more than one IA_PD assigned to it; for example, one for each of its interfaces.
message	A unit of data carried as the payload of a UDP datagram, exchanged among DHCP servers, relay agents and clients.
Reconfigure key	A key supplied to a client by a server used to provide security for Reconfigure messages.
requesting router:	The router that acts as a DHCP client and is requesting prefix(es) to be assigned.
singleton option:	An option that is allowed to appear only once. Most options are singletons.
relaying	A DHCP relay agent relays DHCP messages between DHCP participants.
transaction ID	An opaque value used to match responses with replies initiated either by a client or server.

## 5. Operational Models

This section describes some of the current most common DHCP operational models. The described models are not mutually exclusive and are sometimes used together. For example, a device may start in stateful mode to obtain an address, and at a later time when an application is started, request additional parameters using stateless mode.

### 5.1. Stateless DHCP

Stateless DHCP [RFC3736] is used when DHCP is not used for obtaining an allocatable resource, but a node (DHCP client) desires one or more DHCP "other configuration" parameters, such as a list of DNS recursive name servers or DNS domain search lists [RFC3646]. Stateless may be used when a node initially boots or at any time the software on the node requires some missing or expired configuration information that is available via DHCP.

This is the simplest and most basic operation for DHCP and requires a client (and a server) to support only two messages - Information-request and Reply. Note that DHCP servers and relay agents typically

also need to support the Relay-Forw and Relay-Reply messages to accommodate operation when clients and servers are not on the same link.

## 5.2. DHCP for Non-Temporary Address Assignment

This model of operation was the original motivation for DHCP and is the "stateful address autoconfiguration protocol" for IPv6 [RFC2462]. It is appropriate for situations where stateless address autoconfiguration is not desired, because of network policy, additional requirements (such as updating the DNS with forward or reverse resource records), or client specific requirements (i.e., some prefixes are only available to some clients) which are not possible using stateless address autoconfiguration.

The model of operation for non-temporary address assignment is as follows. The server is provided with IPv6 prefixes from which it may allocate addresses to clients, as well as any related network topology information as to which prefixes are present on which links. A client requests a non-temporary address to be assigned by the server. The server allocates an address or addresses appropriate for the link on which the client is connected. The server returns the allocated address or addresses to the client.

Each address has an associated preferred and valid lifetime, which constitutes an agreement about the length of time over which the client is allowed to use the address. A client can request an extension of the lifetimes on an address and is required to terminate the use of an address if the valid lifetime of the address expires.

Typically clients request other configuration parameters, such as the domain server addresses and search lists, when requesting addresses.

## 5.3. DHCP for Prefix Delegation

The prefix delegation mechanism, originally described in [RFC3633], is another stateful mode of operation and intended for simple delegation of prefixes from a delegating router (DHCP server) to requesting routers (DHCP clients). It is appropriate for situations in which the delegating router does not have knowledge about the topology of the networks to which the requesting router is attached, and the delegating router does not require other information aside from the identity of the requesting router to choose a prefix for delegation. For example, these options would be used by a service provider to assign a prefix to a Customer Premise Equipment (CPE) device acting as a router between the subscriber's internal network and the service provider's core network.

The design of this prefix delegation mechanism meets the requirements for prefix delegation in [RFC3769].

The model of operation for prefix delegation is as follows. A delegating router is provided IPv6 prefixes to be delegated to requesting routers. Examples of ways in which the delegating router may be provided these prefixes is given in Section 19.4. A requesting router requests prefix(es) from the delegating router, as described in Section 19.3. The delegating router chooses prefix(es) for delegation, and responds with prefix(es) to the requesting router. The requesting router is then responsible for the delegated prefix(es). For example, the requesting router might assign a subnet from a delegated prefix to one of its interfaces, and begin sending router advertisements for the prefix on that link.

Each prefix has an associated valid and preferred lifetime, which constitutes an agreement about the length of time over which the requesting router is allowed to use the prefix. A requesting router can request an extension of the lifetimes on a delegated prefix and is required to terminate the use of a delegated prefix if the valid lifetime of the prefix expires.

This prefix delegation mechanism would be appropriate for use by an ISP to delegate a prefix to a subscriber, where the delegated prefix would possibly be subnetted and assigned to the links within the subscriber's network.

Figure 1 illustrates a network architecture in which prefix delegation could be used.



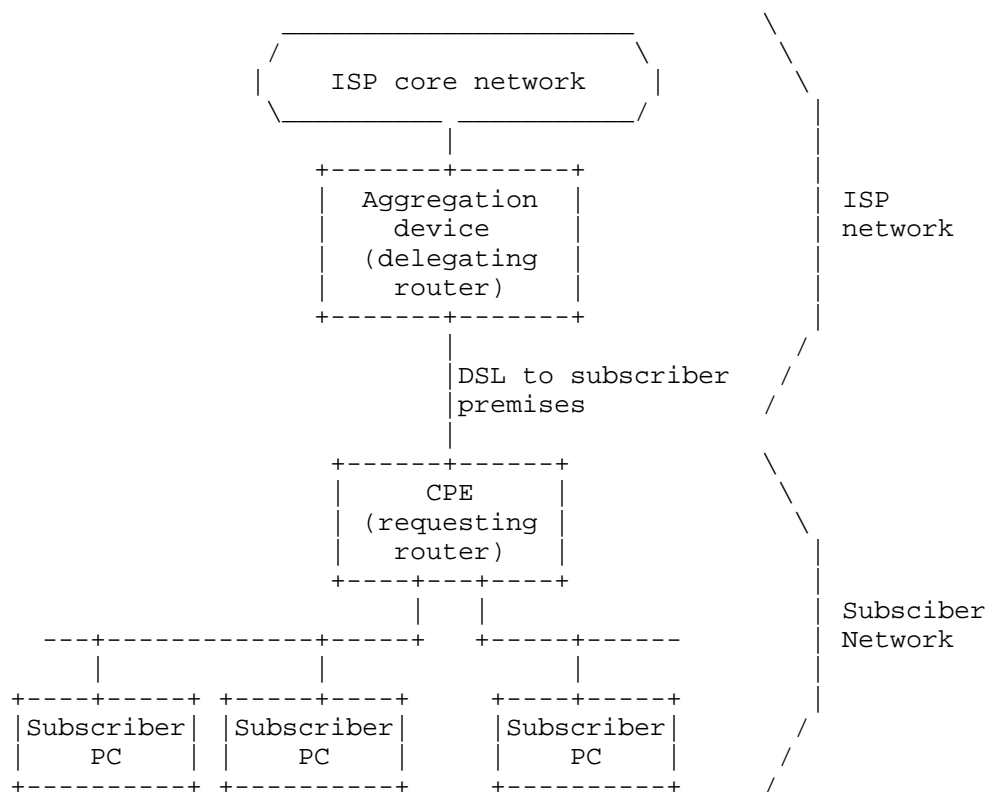


Figure 1: Prefix Delegation Network

In this example, the delegating router is configured with a set of prefixes to be used for assignment to customers at the time of each customer's first connection to the ISP service. The prefix delegation process begins when the requesting router requests configuration information through DHCP. The DHCP messages from the requesting router are received by the delegating router in the aggregation device. When the delegating router receives the request, it selects an available prefix or prefixes for delegation to the requesting router. The delegating router then returns the prefix or prefixes to the requesting router.

The requesting router subnets the delegated prefix and assigns the longer prefixes to links in the subscriber's network. In a typical scenario based on the network shown in Figure 1, the requesting router subnets a single delegated /48 prefix into /64 prefixes and assigns one /64 prefix to each of the links in the subscriber network.

The prefix delegation options can be used in conjunction with other DHCP options carrying other configuration information to the requesting router. The requesting router may, in turn, provide DHCP service to hosts attached to the internal network. For example, the requesting router may obtain the addresses of DNS and NTP servers from the ISP delegating router, and then pass that configuration information on to the subscriber hosts through a DHCP server in the requesting router.

#### 5.4. DHCP for Customer Edge Routers

The DHCP requirements and network architecture for Customer Edge Routers are described in [RFC7084]. This model of operation combines address assignment (see Section 5.2) and prefix delegation (see Section 5.3). In general, this model assumes that a single set of transactions between the client and server will assign or extend the client's non-temporary addresses and delegated prefixes.

#### 5.5. DHCP for Temporary Addresses

Temporary addresses were originally introduced to avoid privacy concerns with stateless address autoconfiguration, which based 64-bits of the address on the EUI-64 (see [RFC3041] and [RFC4941]). They were added to DHCP to provide complementary support when stateful address assignment is used.

Temporary address assignment works mostly like non-temporary address assignment (see Section 5.2), however these addresses are generally intended to be used for a short period of time and not to have their lifetimes extended, though they can be if required.

### 6. DHCP Constants

This section describes various program and networking constants used by DHCP.

#### 6.1. Multicast Addresses

DHCP makes use of the following multicast addresses:

**All\_DHCP\_Relay\_Agents\_and\_Servers (FF02::1:2)** A link-scoped multicast address used by a client to communicate with neighboring (i.e., on-link) relay agents and servers. All servers and relay agents are members of this multicast group.

**All\_DHCP\_Servers (FF05::1:3)** A site-scoped multicast address used by a relay agent to communicate with servers, either

because the relay agent wants to send messages to all servers or because it does not know the unicast addresses of the servers. Note that in order for a relay agent to use this address, it must have an address of sufficient scope to be reachable by the servers. All servers within the site are members of this multicast group.

## 6.2. UDP Ports

Clients listen for DHCP messages on UDP port 546. Servers and relay agents listen for DHCP messages on UDP port 547.

## 6.3. DHCP Message Types

DHCP defines the following message types. More detail on these message types can be found in Section 7 and Section 8. Message types not listed here are reserved for future use. The numeric encoding for each message type is shown in parentheses.

- |               |  |
|---------------|--|
| SOLICIT (1)   | A client sends a Solicit message to locate servers.  |
| ADVERTISE (2) | A server sends an Advertise message to indicate that it is available for DHCP service, in response to a Solicit message received from a client.  |
| REQUEST (3)   | A client sends a Request message to request configuration parameters, including IP addresses, from a specific server.  |
| CONFIRM (4)   | A client sends a Confirm message to any available server to determine whether the addresses it was assigned are still appropriate to the link to which the client is connected.  |
| RENEW (5)     | A client sends a Renew message to the server that originally provided the client's addresses and configuration parameters to extend the lifetimes on the addresses assigned to the client and to update other configuration parameters.            |
| REBIND (6)    | A client sends a Rebind message to any available server to extend the lifetimes on the addresses assigned to the client and to update other configuration parameters; this message is sent after a client receives no response to a Renew message. |

- REPLY (7) A server sends a Reply message containing assigned addresses and configuration parameters in response to a Solicit, Request, Renew, Rebind message received from a client. A server sends a Reply message containing configuration parameters in response to an Information-request message. A server sends a Reply message in response to a Confirm message confirming or denying that the addresses assigned to the client are appropriate to the link to which the client is connected. A server sends a Reply message to acknowledge receipt of a Release or Decline message.
- RELEASE (8) A client sends a Release message to the server that assigned addresses to the client to indicate that the client will no longer use one or more of the assigned addresses.
- DECLINE (9) A client sends a Decline message to a server to indicate that the client has determined that one or more addresses assigned by the server are already in use on the link to which the client is connected.
- RECONFIGURE (10) A server sends a Reconfigure message to a client to inform the client that the server has new or updated configuration parameters, and that the client is to initiate a Renew/Reply or Information-request/Reply transaction with the server in order to receive the updated information.
- INFORMATION-REQUEST (11) A client sends an Information-request message to a server to request configuration parameters without the assignment of any IP addresses to the client.
- RELAY-FORW (12) A relay agent sends a Relay-forward message to relay messages to servers, either directly or through another relay agent. The received message, either a client message or a Relay-forward message from another relay agent, is encapsulated in an option in the Relay-forward message.
- RELAY-REPL (13) A server sends a Relay-reply message to a relay agent containing a message that the relay agent delivers to a client. The Relay-reply message may be relayed by other relay agents for delivery to the destination relay agent.

The server encapsulates the client message as an option in the Relay-reply message, which the relay agent extracts and relays to the client.

#### 6.4. Status Codes

DHCPv6 uses status codes to communicate the success or failure of operations requested in messages from clients and servers, and to provide additional information about the specific cause of the failure of a message. The specific status codes are defined in Section 23.12.

If the Status Code option does not appear in a message in which the option could appear, the status of the message is assumed to be Success.

#### 6.5. Transmission and Retransmission Parameters

This section presents a table of values used to describe the message transmission behavior of clients and servers.

Parameter	Default	Description
SOL_MAX_DELAY	1 sec	Max delay of first Solicit
SOL_TIMEOUT	1 sec	Initial Solicit timeout
SOL_MAX_RT	3600 secs	Max Solicit timeout value
REQ_TIMEOUT	1 sec	Initial Request timeout
REQ_MAX_RT	30 secs	Max Request timeout value
REQ_MAX_RC	10	Max Request retry attempts
CNF_MAX_DELAY	1 sec	Max delay of first Confirm
CNF_TIMEOUT	1 sec	Initial Confirm timeout
CNF_MAX_RT	4 secs	Max Confirm timeout
CNF_MAX_RD	10 secs	Max Confirm duration
REN_TIMEOUT	10 secs	Initial Renew timeout
REN_MAX_RT	600 secs	Max Renew timeout value
REB_TIMEOUT	10 secs	Initial Rebind timeout
REB_MAX_RT	600 secs	Max Rebind timeout value
INF_MAX_DELAY	1 sec	Max delay of first Information-request
INF_TIMEOUT	1 sec	Initial Information-request timeout
INF_MAX_RT	3600 secs	Max Information-request timeout value
REL_TIMEOUT	1 sec	Initial Release timeout
REL_MAX_RC	4	MAX Release retry attempts
DEC_TIMEOUT	1 sec	Initial Decline timeout
DEC_MAX_RC	4	Max Decline retry attempts
REC_TIMEOUT	2 secs	Initial Reconfigure timeout
REC_MAX_RC	8	Max Reconfigure attempts
HOP_COUNT_LIMIT	32	Max hop count in a Relay-forward message

#### 6.6. Representation of time values and "Infinity" as a time value

All time values for lifetimes, T1 and T2 are unsigned integers. The value 0xffffffff is taken to mean "infinity" when used as a lifetime (as in [RFC4861]) or a value for T1 or T2.

#### 7. Client/Server Message Formats

All DHCP messages sent between clients and servers share an identical fixed format header and a variable format area for options.

All values in the message header and in options are in network byte order.

Options are stored serially in the options field, with no padding between the options. Options are byte-aligned but are not aligned in any other way such as on 2 or 4 byte boundaries.

The following diagram illustrates the format of DHCP messages sent between clients and servers:

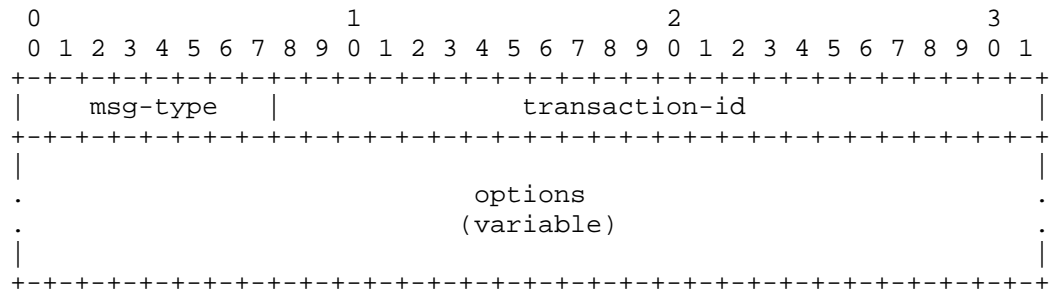


Figure 2: Client/Server message format

msg-type	Identifies the DHCP message type; the available message types are listed in Section 6.3.
transaction-id	The transaction ID for this message exchange.
options	Options carried in this message; options are described in Section 23.

## 8. Relay Agent/Server Message Formats

Relay agents exchange messages with servers to relay messages between clients and servers that are not connected to the same link.

All values in the message header and in options are in network byte order.

Options are stored serially in the options field, with no padding between the options. Options are byte-aligned but are not aligned in any other way such as on 2 or 4 byte boundaries.

There are two relay agent messages, which share the following format:

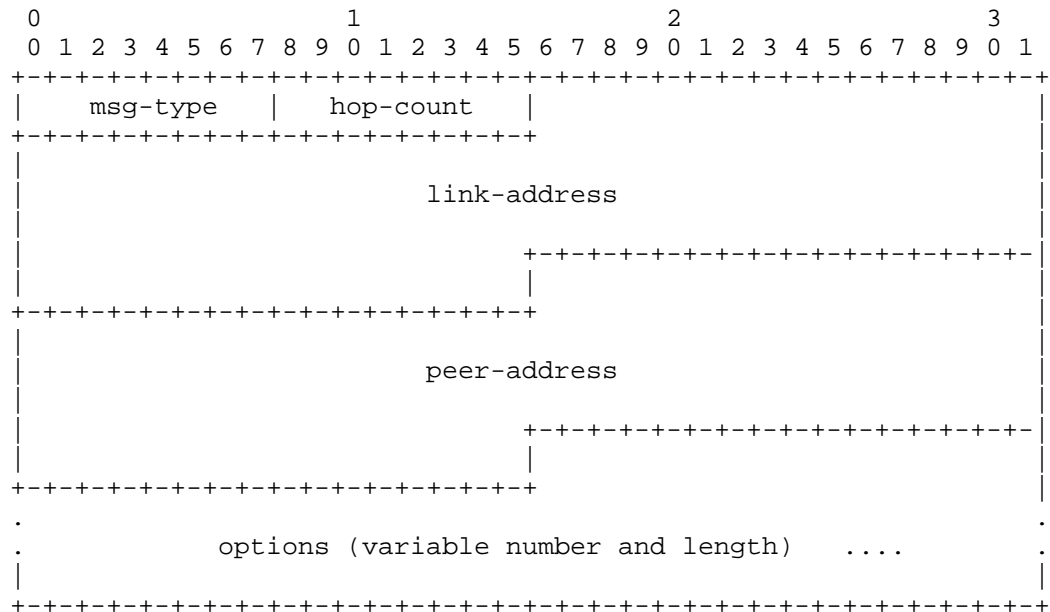


Figure 3: Relay Agent/Server message format

The following sections describe the use of the Relay Agent message header.

#### 8.1. Relay-forward Message

The following table defines the use of message fields in a Relay-forward message.

msg-type	RELAY-FORW
hop-count	Number of relay agents that have relayed this message.
link-address	An address that will be used by the server to identify the link on which the client is located. This is typically global, site-scoped or ULA [RFC4193], but see discussion in Section 21.1.1.
peer-address	The address of the client or relay agent from which the message to be relayed was received.



options	MUST include a "Relay Message option" (see Section 23.10); MAY include other options added by the relay agent.
---------	--

## 8.2. Relay-reply Message

The following table defines the use of message fields in a Relay-reply message.

msg-type	RELAY-REPL
hop-count	Copied from the Relay-forward message
link-address	Copied from the Relay-forward message
peer-address	Copied from the Relay-forward message
options	MUST include a "Relay Message option"; see Section 23.10; MAY include other options

## 9. Representation and Use of Domain Names

So that domain names may be encoded uniformly, a domain name or a list of domain names is encoded using the technique described in section 3.1 of [RFC1035]. A domain name, or list of domain names, in DHCP MUST NOT be stored in compressed form, as described in section 4.1.4 of [RFC1035].

## 10. DHCP Unique Identifier (DUID)

Each DHCP client and server has a DUID. DHCP servers use DUIDs to identify clients for the selection of configuration parameters and in the association of IAs with clients. DHCP clients use DUIDs to identify a server in messages where a server needs to be identified. See Section 23.2 and Section 23.3 for the representation of a DUID in a DHCP message.

Clients and servers MUST treat DUIDs as opaque values and MUST only compare DUIDs for equality. Clients and servers MUST NOT in any other way interpret DUIDs. Clients and servers MUST NOT restrict DUIDs to the types defined in this document, as additional DUID types may be defined in the future.

The DUID is carried in an option because it may be variable length and because it is not required in all DHCP messages. The DUID is designed to be unique across all DHCP clients and servers, and stable for any specific client or server - that is, the DUID used by a client or server SHOULD NOT change over time if at all possible; for

example, a device's DUID should not change as a result of a change in the device's network hardware.

The motivation for having more than one type of DUID is that the DUID must be globally unique, and must also be easy to generate. The sort of globally-unique identifier that is easy to generate for any given device can differ quite widely. Also, some devices may not contain any persistent storage. Retaining a generated DUID in such a device is not possible, so the DUID scheme must accommodate such devices.

#### 10.1. DUID Contents

A DUID consists of a two-octet type code represented in network byte order, followed by a variable number of octets that make up the actual identifier. The length of the DUID (not including the type code) is at least 1 octet and at most 128 octets. The following types are currently defined:

Type	Description
1	Link-layer address plus time
2	Vendor-assigned unique ID based on Enterprise Number
3	Link-layer address
4	Universally Unique Identifier (UUID) - see [RFC6355]

Formats for the variable field of the DUID for the first 3 of the above types are shown below. The fourth type, DUID-UUID [RFC6355], can be used in situations where there is a UUID stored in a device's firmware settings.

#### 10.2. DUID Based on Link-layer Address Plus Time, DUID-LLT

This type of DUID consists of a two octet type field containing the value 1, a two octet hardware type code, four octets containing a time value, followed by link-layer address of any one network interface that is connected to the DHCP device at the time that the DUID is generated. The time value is the time that the DUID is generated represented in seconds since midnight (UTC), January 1, 2000, modulo  $2^{32}$ . The hardware type MUST be a valid hardware type assigned by the IANA as described in [RFC0826]. Both the time and the hardware type are stored in network byte order. The link-layer address is stored in canonical form, as described in [RFC2464].

The following diagram illustrates the format of a DUID-LLT:

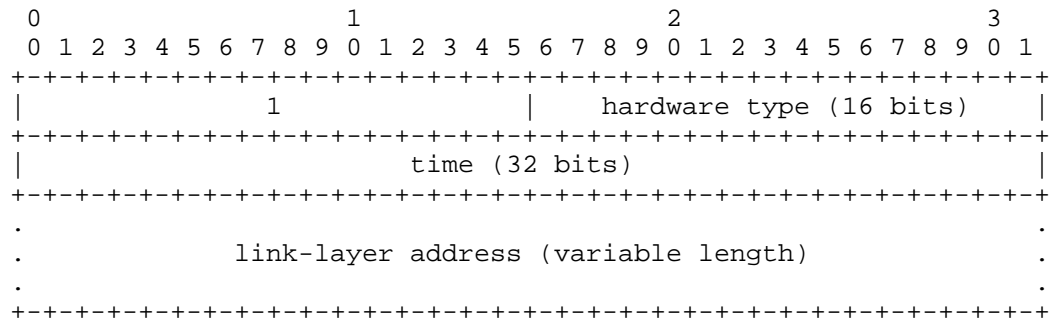


Figure 4: DUID-LLT format

The choice of network interface can be completely arbitrary, as long as that interface provides a globally unique link-layer address for the link type, and the same DUID-LLT SHOULD be used in configuring all network interfaces connected to the device, regardless of which interface's link-layer address was used to generate the DUID-LLT.

Clients and servers using this type of DUID MUST store the DUID-LLT in stable storage, and MUST continue to use this DUID-LLT even if the network interface used to generate the DUID-LLT is removed. Clients and servers that do not have any stable storage MUST NOT use this type of DUID.

Clients and servers that use this DUID SHOULD attempt to configure the time prior to generating the DUID, if that is possible, and MUST use some sort of time source (for example, a real-time clock) in generating the DUID, even if that time source could not be configured prior to generating the DUID. The use of a time source makes it unlikely that two identical DUID-LLTs will be generated if the network interface is removed from the client and another client then uses the same network interface to generate a DUID-LLT. A collision between two DUID-LLTs is very unlikely even if the clocks have not been configured prior to generating the DUID.

This method of DUID generation is recommended for all general purpose computing devices such as desktop computers and laptop computers, and also for devices such as printers, routers, and so on, that contain some form of writable non-volatile storage.

Despite our best efforts, it is possible that this algorithm for generating a DUID could result in a client identifier collision. A DHCP client that generates a DUID-LLT using this mechanism MUST provide an administrative interface that replaces the existing DUID with a newly-generated DUID-LLT.

### 10.3. DUID Assigned by Vendor Based on Enterprise Number, DUID-EN

This form of DUID is assigned by the vendor to the device. It consists of the vendor's registered Private Enterprise Number as maintained by IANA [IANA-PEN] followed by a unique identifier assigned by the vendor. The following diagram summarizes the structure of a DUID-EN:

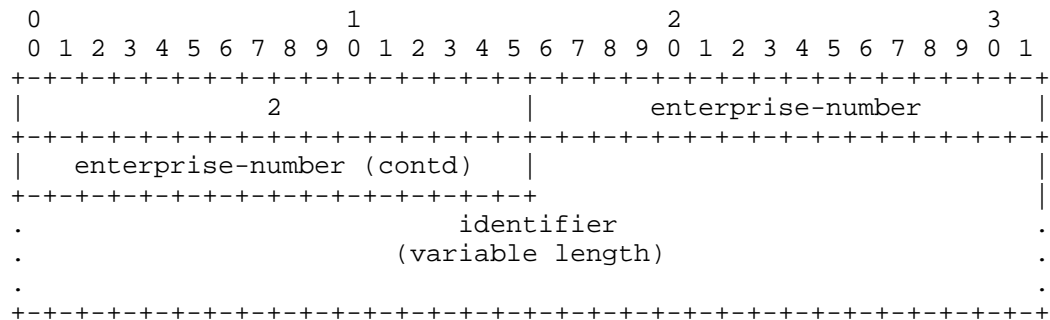


Figure 5: DUID-EN format

The source of the identifier is left up to the vendor defining it, but each identifier part of each DUID-EN MUST be unique to the device that is using it, and MUST be assigned to the device no later than at the first usage and stored in some form of non-volatile storage. This typically means being assigned during manufacture process in case of physical devices or when the image is created or booted for the first time in case of virtual machines. The generated DUID SHOULD be recorded in non-erasable storage. The enterprise-number is the vendor's registered Private Enterprise Number as maintained by IANA [IANA-PEN]. The enterprise-number is stored as an unsigned 32 bit number.

An example DUID of this type might look like this:

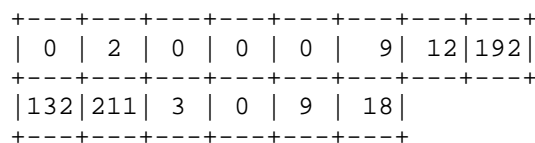


Figure 6: DUID-EN example

This example includes the two-octet type of 2, the Enterprise Number (9), followed by eight octets of identifier data (0x0CC084D303000912).

#### 10.4. DUID Based on Link-layer Address, DUID-LL

This type of DUID consists of two octets containing the DUID type 3, a two octet network hardware type code, followed by the link-layer address of any one network interface that is permanently connected to the client or server device. For example, a host that has a network interface implemented in a chip that is unlikely to be removed and used elsewhere could use a DUID-LL. The hardware type **MUST** be a valid hardware type assigned by the IANA, as described in [RFC0826]. The hardware type is stored in network byte order. The link-layer address is stored in canonical form, as described in [RFC2464]. The following diagram illustrates the format of a DUID-LL:

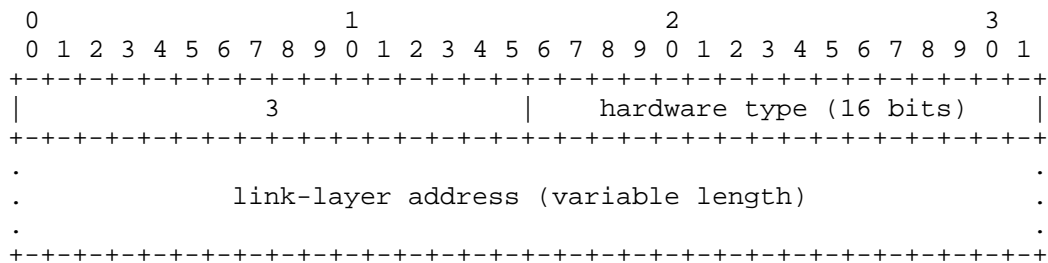


Figure 7: DUID-LL format

The choice of network interface can be completely arbitrary, as long as that interface provides a unique link-layer address and is permanently attached to the device on which the DUID-LL is being generated. The same DUID-LL **SHOULD** be used in configuring all network interfaces connected to the device, regardless of which interface's link-layer address was used to generate the DUID.

DUID-LL is recommended for devices that have a permanently-connected network interface with a link-layer address, and do not have nonvolatile, writable stable storage. DUID-LL **MUST NOT** be used by DHCP clients or servers that cannot tell whether or not a network interface is permanently attached to the device on which the DHCP client is running.

## 11. Identity Association

An "identity-association" (IA) is a construct through which a server and a client can identify, group, and manage a set of related IPv6 addresses or delegated prefixes. Each IA consists of an IAID and associated configuration information.

The IAID uniquely identifies the IA and must be chosen to be unique among the IAIDs for that IA type on the client. The IAID is chosen by the client. For any given use of an IA by the client, the IAID for that IA MUST be consistent across restarts of the DHCP client. The client may maintain consistency either by storing the IAID in non-volatile storage or by using an algorithm that will consistently produce the same IAID as long as the configuration of the client has not changed. There may be no way for a client to maintain consistency of the IAIDs if it does not have non-volatile storage and the client's hardware configuration changes. If the client uses only one IAID, it can use a well-known value, e.g., zero.

### 11.1. Identity Associations for Address Assignment

A client must associate at least one distinct IA with each of its network interfaces for which it is to request the assignment of IPv6 addresses from a DHCP server. The client uses the IAs assigned to an interface to obtain configuration information from a server for that interface. Each IA must be associated with exactly one interface.

The configuration information in an IA consists of one or more IPv6 addresses along with the times T1 and T2 for the IA. See Section 22.4 for the representation of an IA in a DHCP message.

Each address in an IA has a preferred lifetime and a valid lifetime, as defined in [RFC4862]. The lifetimes are transmitted from the DHCP server to the client in the IA option. The lifetimes apply to the use of IPv6 addresses, as described in section 5.5.4 of [RFC4862].

### 11.2. Identity Associations for Prefix Delegation

An IA\_PD is different from an IA for address assignment, in that it does not need to be associated with exactly one interface. One IA\_PD can be associated with the requesting router, with a set of interfaces or with exactly one interface. A requesting router must create at least one distinct IA\_PD. It may associate a distinct IA\_PD with each of its downstream network interfaces and use that IA\_PD to obtain a prefix for that interface from the delegating router.

The configuration information in an IA\_PD consists of one or more IPv6 prefixes along with the times T1 and T2 for the IA\_PD. See Section 23.21 for the representation of an IA\_PD in a DHCP message.

## 12. Selecting Addresses for Assignment to an IA

A server selects addresses to be assigned to an IA according to the address assignment policies determined by the server administrator and the specific information the server determines about the client from some combination of the following sources:

- The link to which the client is attached. The server determines the link as follows:
  - \* If the server receives the message directly from the client and the source address in the IP datagram in which the message was received is a link-local address, then the client is on the same link to which the interface over which the message was received is attached.
  - \* If the server receives the message from a forwarding relay agent, then the client is on the same link as the one to which the interface, identified by the link-address field in the message from the relay agent, is attached. According to [RFC6221], the server MUST ignore any link-address field whose value is zero. The link address field refers to the link-address field of the Relay-Forward message, and the link-address fields in any Relay-Forward messages that may be nested within the Relay-Forward message.
  - \* If the server receives the message directly from the client and the source address in the IP datagram in which the message was received is not a link-local address, then the client is on the link identified by the source address in the IP datagram (note that this situation can occur only if the server has enabled the use of unicast message delivery by the client and the client has sent a message for which unicast delivery is allowed).
- The DUID supplied by the client.
- Other information in options supplied by the client, e.g. IA Address options that include the client's requests for specific addresses.
- Other information in options supplied by the relay agent.

Any address assigned by a server that is based on an EUI-64 identifier MUST include an interface identifier with the "u" (universal/local) and "g" (individual/group) bits of the interface identifier set appropriately, as indicated in section 2.5.1 of [RFC4291].

A server MUST NOT assign an address that is otherwise reserved for some other purpose. For example, a server MUST NOT assign reserved anycast addresses, as defined in [RFC2526], from any subnet.

### 13. Management of Temporary Addresses

A client may request the assignment of temporary addresses (see [RFC4941] for the definition of temporary addresses). DHCPv6 handling of address assignment is no different for temporary addresses.

Clients ask for temporary addresses and servers assign them. Temporary addresses are carried in the Identity Association for Temporary Addresses (IA\_TA) option (see Section 23.5). Each IA\_TA option contains at most one temporary address for each of the prefixes on the link to which the client is attached.

The lifetime of the assigned temporary address is set in the IA Address Option (see Section 23.6) within the IA\_TA option. It is RECOMMENDED to set short lifetimes, typically shorter than TEMP\_VALID\_LIFETIME and TEMP\_PREFERRED\_LIFETIME (see Section 5, [RFC4941]).

The IAID number space for the IA\_TA option IAID number space is separate from the IA\_NA option IAID number space.

A DHCPv6 server implementation MAY generate temporary addresses referring to the algorithm defined in Section 3.2.1, [RFC4941], with additional condition that the new address is not duplicated with any assigned addresses.

The server MAY update the DNS for a temporary address, as described in section 4 of [RFC4941].

On the clients, by default, temporary addresses are preferred in source address selection, according to Rule 7, [RFC6724]. However, this policy is overridable.

One of the most important properties of temporary address is unlinkability of different actions over time. So, it is NOT RECOMMENDED for a client to renew expired temporary addresses, though DHCPv6 provides such possibility (see Section 23.5).



## 14. Transmission of Messages by a Client

Unless otherwise specified in this document, or in a document that describes how IPv6 is carried over a specific type of link (for link types that do not support multicast), a client sends DHCP messages to the All\_DHCP\_Relay\_Agents\_and\_Servers.

A client uses multicast to reach all servers or an individual server. An individual server is indicated by specifying that server's DUID in a Server Identifier option (see Section 23.3) in the client's message (all servers will receive this message but only the indicated server will respond). All servers are indicated by not supplying this option.

A client may send some messages directly to a server using unicast, as described in Section 23.12.

### 14.1. Rate Limiting

In order to avoid prolonged message bursts that may be caused by possible logic loops, a DHCPv6 client MUST limit the rate of DHCPv6 messages it transmits. One example is that a client obtains an address, but does not like the response; it reverts back to Solicit procedure, discovers the same (sole) server, requests an address and gets the same address as before (the server still has the lease that was requested just previously). This loops can repeat infinitely if there is not a quit/stop mechanism. Therefore, a client must not initiate transmissions too frequently.

A recommended method for implementing the rate limiting function is a token bucket, limiting the average rate of transmission to a certain number in a certain time. This method of bounding burstiness also guarantees that the long-term transmission rate will not exceed.

TRT        Transmission Rate Limit

The Transmission Rate Limit parameter (TRT) SHOULD be configurable. A possible default could be 20 packets in 20 seconds.

For a device that has multiple interfaces, the limit MUST be enforced on a per interface basis.

Rate limiting of forwarded DHCPv6 messages and server-side messages are out of scope of this specification.

## 15. Reliability of Client Initiated Message Exchanges

DHCP clients are responsible for reliable delivery of messages in the client-initiated message exchanges described in Section 18 and Section 19. If a DHCP client fails to receive an expected response from a server, the client must retransmit its message. This section describes the retransmission strategy to be used by clients in client-initiated message exchanges.

Note that the procedure described in this section is slightly modified when used with the Solicit message. The modified procedure is described in Section 18.1.2.

The client begins the message exchange by transmitting a message to the server. The message exchange terminates when either the client successfully receives the appropriate response or responses from a server or servers, or when the message exchange is considered to have failed according to the retransmission mechanism described below.

The client retransmission behavior is controlled and described by the following variables:

RT	Retransmission timeout
IRT	Initial retransmission time
MRC	Maximum retransmission count
MRT	Maximum retransmission time
MRD	Maximum retransmission duration
RAND	Randomization factor

With each message transmission or retransmission, the client sets RT according to the rules given below. If RT expires before the message exchange terminates, the client recomputes RT and retransmits the message.

Each of the computations of a new RT include a randomization factor (RAND), which is a random number chosen with a uniform distribution between -0.1 and +0.1. The randomization factor is included to minimize synchronization of messages transmitted by DHCP clients.

The algorithm for choosing a random number does not need to be cryptographically sound. The algorithm SHOULD produce a different sequence of random numbers from each invocation of the DHCP client.

RT for the first message transmission is based on IRT:

$$RT = IRT + RAND * IRT$$

RT for each subsequent message transmission is based on the previous value of RT:

$$RT = 2 * RT_{prev} + RAND * RT_{prev}$$

MRT specifies an upper bound on the value of RT (disregarding the randomization added by the use of RAND). If MRT has a value of 0, there is no upper limit on the value of RT. Otherwise:

$$\begin{aligned} &\text{if } (RT > MRT) \\ &\quad RT = MRT + RAND * MRT \end{aligned}$$

MRC specifies an upper bound on the number of times a client may retransmit a message. Unless MRC is zero, the message exchange fails once the client has transmitted the message MRC times.

MRD specifies an upper bound on the length of time a client may retransmit a message. Unless MRD is zero, the message exchange fails once MRD seconds have elapsed since the client first transmitted the message.

If both MRC and MRD are non-zero, the message exchange fails whenever either of the conditions specified in the previous two paragraphs are met.

If both MRC and MRD are zero, the client continues to transmit the message until it receives a response.

A client is not expected to listen for a response during the entire period between transmission of Solicit or Information-request messages.

## 16. Message Validation

Clients and servers might get messages that contain options not allowed to appear in the received message. For example, an IA option is not allowed to appear in an Information-request message. Clients and servers MAY choose either to extract information from such a message if the information is of use to the recipient, or to ignore such message completely and just drop it.

A server MUST discard any Solicit, Confirm, Rebind or Information-request messages it receives with a unicast destination address.

Message validation based on DHCP authentication is discussed in Section 22.4.2.

If a server receives a message that contains options it should not contain (such as an Information-request message with an IA option), is missing options that it should contain, or is otherwise not valid, it MAY send a Reply (or Advertise as appropriate) with a Server Identifier option, a Client Identifier option if one was included in the message and a Status Code option with status UnSpecFail.

A client or server MUST silently discard and ignore received DHCPv6 messages with an unknown message type.

#### 16.1. Use of Transaction IDs

The "transaction-id" field holds a value used by clients and servers to synchronize server responses to client messages. A client SHOULD generate a random number that cannot easily be guessed or predicted to use as the transaction ID for each new message it sends. Note that if a client generates easily predictable transaction identifiers, it may become more vulnerable to certain kinds of attacks from off-path intruders. A client MUST leave the transaction ID unchanged in retransmissions of a message.

#### 16.2. Solicit Message

Clients MUST discard any received Solicit messages.

Servers MUST discard any Solicit messages that do not include a Client Identifier option or that do include a Server Identifier option.

#### 16.3. Advertise Message

Clients MUST discard any received Advertise message that meets any of the following conditions:

- the message does not include a Server Identifier option.
- the message does not include a Client Identifier option.
- the contents of the Client Identifier option does not match the client's DUID.
- the "transaction-id" field value does not match the value the client used in its Solicit message.

Servers and relay agents MUST discard any received Advertise messages.

#### 16.4. Request Message

Clients MUST discard any received Request messages.

Servers MUST discard any received Request message that meets any of the following conditions:

- the message does not include a Server Identifier option.
- the contents of the Server Identifier option do not match the server's DUID.
- the message does not include a Client Identifier option.

#### 16.5. Confirm Message

Clients MUST discard any received Confirm messages.

Servers MUST discard any received Confirm messages that do not include a Client Identifier option or that do include a Server Identifier option.

#### 16.6. Renew Message

Clients MUST discard any received Renew messages.

Servers MUST discard any received Renew message that meets any of the following conditions:

- the message does not include a Server Identifier option.
- the contents of the Server Identifier option does not match the server's identifier.
- the message does not include a Client Identifier option.

#### 16.7. Rebind Message

Clients MUST discard any received Rebind messages.

Servers MUST discard any received Rebind messages that do not include a Client Identifier option or that do include a Server Identifier option.

#### 16.8. Decline Messages

Clients MUST discard any received Decline messages.

Servers MUST discard any received Decline message that meets any of the following conditions:

- the message does not include a Server Identifier option.
- the contents of the Server Identifier option does not match the server's identifier.
- the message does not include a Client Identifier option.

#### 16.9. Release Message

Clients MUST discard any received Release messages.

Servers MUST discard any received Release message that meets any of the following conditions:

- the message does not include a Server Identifier option.
- the contents of the Server Identifier option does not match the server's identifier.
- the message does not include a Client Identifier option.

#### 16.10. Reply Message

Clients MUST discard any received Reply message that meets any of the following conditions:

- the message does not include a Server Identifier option.
- the "transaction-id" field in the message does not match the value used in the original message.

If the client included a Client Identifier option in the original message, the Reply message MUST include a Client Identifier option and the contents of the Client Identifier option MUST match the DUID of the client; OR, if the client did not include a Client Identifier option in the original message, the Reply message MUST NOT include a Client Identifier option.

Servers and relay agents MUST discard any received Reply messages.

#### 16.11. Reconfigure Message

Servers and relay agents MUST discard any received Reconfigure messages.

Clients MUST discard any Reconfigure message that meets any of the following conditions:

- the message was not unicast to the client.
- the message does not include a Server Identifier option.
- the message does not include a Client Identifier option that contains the client's DUID.
- the message does not contain a Reconfigure Message option.
- the Reconfigure Message option msg-type is not a valid value.
- the message includes any IA options and the msg-type in the Reconfigure Message option is INFORMATION-REQUEST.
- the message does not include DHCP authentication:
  - \* the message does not contain an authentication option.
  - \* the message does not pass the authentication validation performed by the client.

#### 16.12. Information-request Message

Clients MUST discard any received Information-request messages.

Servers MUST discard any received Information-request message that meets any of the following conditions:

- The message includes a Server Identifier option and the DUID in the option does not match the server's DUID.
- The message includes an IA option.

#### 16.13. Relay-forward Message

Clients MUST discard any received Relay-forward messages.

#### 16.14. Relay-reply Message

Clients and servers MUST discard any received Relay-reply messages.

### 17. Client Source Address and Interface Selection

Client's behavior is different depending on the purpose of the configuration.

#### 17.1. Address Assignment

When a client sends a DHCP message to the `All_DHCP_Relay_Agents_and_Servers` address, it SHOULD send the message through the interface for which configuration information is being requested. However, the client MAY send the message through another interface if the interface is a logical interface without direct link attachment or the client is certain that two interfaces are attached to the same link.

When a client sends a DHCP message directly to a server using unicast (after receiving the Server Unicast option from that server), the source address in the header of the IPv6 datagram MUST be an address assigned to the interface for which the client is interested in obtaining configuration and which is suitable for use by the server in responding to the client.

#### 17.2. Prefix Delegation

Delegated prefixes are not associated with a particular interface in the same way as addresses are for address assignment, and mentioned above.

When a client (acting as requesting router) sends a DHCP message for the purpose of prefix delegation, it SHOULD be sent on the interface associated with the upstream router (ISP network). The upstream interface is typically determined by configuration. This rule applies even in the case where a separate `IA_PD` is used for each downstream interface.

When a requesting router sends a DHCP message directly to a delegating router using unicast (after receiving the Server Unicast option from that delegating router), the source address SHOULD be an address from the upstream interface and which is suitable for use by the delegating router in responding to the requesting router.



## 18. DHCP Server Solicitation

This section describes how a client locates servers that will assign addresses and delegated prefixes to IAs belonging to the client.

The client is responsible for creating IAs and requesting that a server assign IPv6 addresses and delegated prefixes to the IAs. The client first creates the IAs and assigns IAIDs to them. The client then transmits a Solicit message containing the IA options describing the IAs. The client **MUST NOT** be using any of the addresses or delegated prefixes for which it tries to obtain the bindings by sending the Solicit message. In particular, if the client had some valid bindings and has chosen to start the server solicitation process to obtain the bindings from a different server, the client **MUST** stop using the addresses and delegated prefixes for the bindings it had obtained from the previous server, and which it is now trying to obtain from a new server.

Servers that can assign addresses or delegated prefixes to the IAs respond to the client with an Advertise message. The client then initiates a configuration exchange as described in Section 19.

If the client will accept a Reply message with committed address assignments and other resources in response to the Solicit message, the client includes a Rapid Commit option (see Section 23.14) in the Solicit message.

### 18.1. Client Behavior

A client uses the Solicit message to discover DHCP servers configured to assign addresses or return other configuration parameters on the link to which the client is attached.

#### 18.1.1. Creation of Solicit Messages

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

The client **MUST** include a Client Identifier option to identify itself to the server. The client includes IA options for any IAs to which it wants the server to assign addresses. The client **MAY** include addresses in the IAs as a hint to the server about addresses for which the client has a preference. The client **MUST NOT** include any other options in the Solicit message, except as specifically allowed in the definition of individual options.

The client uses IA\_NA options to request the assignment of non-temporary addresses and uses IA\_TA options to request the assignment of temporary addresses. Either IA\_NA or IA\_TA options, or a combination of both, can be included in DHCP messages.

The client MUST include an Option Request option (see Section 23.7) to request the SOL\_MAX\_RT option (see Section 23.23) and any other options the client is interested in receiving. The client MAY additionally include instances of those options that are identified in the Option Request option, with data values as hints to the server about parameter values the client would like to have returned.

The client includes a Reconfigure Accept option (see Section 23.20) if the client is willing to accept Reconfigure messages from the server.

#### 18.1.2. Transmission of Solicit Messages

The first Solicit message from the client on the interface MUST be delayed by a random amount of time between 0 and SOL\_MAX\_DELAY. In the case of a Solicit message transmitted when DHCP is initiated by IPv6 Neighbor Discovery, the delay gives the amount of time to wait after IPv6 Neighbor Discovery causes the client to invoke the stateful address autoconfiguration protocol (see section 5.5.3 of [RFC4862]). This random delay desynchronizes clients which start at the same time (for example, after a power outage).

The client transmits the message according to Section 15, using the following parameters:

IRT	SOL_TIMEOUT
MRT	SOL_MAX_RT
MRC	0
MRD	0

If the client has included a Rapid Commit option in its Solicit message, the client terminates the waiting process as soon as a Reply message with a Rapid Commit option is received.

If the client is waiting for an Advertise message, the mechanism in Section 15 is modified as follows for use in the transmission of Solicit messages. The message exchange is not terminated by the receipt of an Advertise before the first RT has elapsed. Rather, the client collects Advertise messages until the first RT has elapsed.

Also, the first RT MUST be selected to be strictly greater than IRT by choosing RAND to be strictly greater than 0.

A client MUST collect Advertise messages for the first RT seconds, unless it receives an Advertise message with a preference value of 255. The preference value is carried in the Preference option (Section 23.8). Any Advertise that does not include a Preference option is considered to have a preference value of 0. If the client receives an Advertise message that includes a Preference option with a preference value of 255, the client immediately begins a client-initiated message exchange (as described in Section 19) by sending a Request message to the server from which the Advertise message was received. If the client receives an Advertise message that does not include a Preference option with a preference value of 255, the client continues to wait until the first RT elapses. If the first RT elapses and the client has received an Advertise message, the client SHOULD continue with a client-initiated message exchange by sending a Request message.

If the client does not receive any Advertise messages before the first RT has elapsed, it begins the retransmission mechanism described in Section 15. The client terminates the retransmission process as soon as it receives any Advertise message, and the client acts on the received Advertise message without waiting for any additional Advertise messages.

A DHCP client SHOULD choose MRC and MRD to be 0. If the DHCP client is configured with either MRC or MRD set to a value other than 0, it MUST stop trying to configure the interface if the message exchange fails. After the DHCP client stops trying to configure the interface, it SHOULD restart the reconfiguration process after some external event, such as user input, system restart, or when the client is attached to a new link.

#### 18.1.3. Receipt of Advertise Messages

The client MUST process SOL\_MAX\_RT and INF\_MAX\_RT options in an Advertise message, even if the message contains a Status Code option indicating a failure, and the Advertise message will be discarded by the client.

The client MUST ignore any IAs in an Advertise message that include a Status Code option containing the value NoAddrsAvail, with the exception that the client MAY display the associated status message to the user.

Upon receipt of one or more valid Advertise messages, the client selects one or more Advertise messages based upon the following criteria.

- Those Advertise messages with the highest server preference value are preferred over all other Advertise messages.
- Within a group of Advertise messages with the same server preference value, a client MAY select those servers whose Advertise messages advertise information of interest to the client.
- 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.

Once a client has selected Advertise message(s), the client will typically store information about each server, such as server preference value, addresses advertised, when the advertisement was received, and so on.

In practice, this means that the client will maintain independent per-IA state machines per each selected server.

If the client needs to select an alternate server in the case that a chosen server does not respond, the client chooses the next server according to the criteria given above.

#### 18.1.4. Receipt of Reply Message

If the client includes a Rapid Commit option in the Solicit message, it will expect a Reply message that includes a Rapid Commit option in response. The client discards any Reply messages it receives that do not include a Rapid Commit option. If the client receives a valid Reply message that includes a Rapid Commit option, it processes the message as described in Section 19.1.8. If it does not receive such a Reply message and does receive a valid Advertise message, the client processes the Advertise message as described in Section 18.1.3.

If the client subsequently receives a valid Reply message that includes a Rapid Commit option, it either:

- processes the Reply message as described in Section 19.1.8, and discards any Reply messages received in response to the Request message, or

- processes any Reply messages received in response to the Request message and discards the Reply message that includes the Rapid Commit option.

## 18.2. Server Behavior

A server sends an Advertise message in response to valid Solicit messages it receives to announce the availability of the server to the client.

### 18.2.1. Receipt of Solicit Messages

The server determines the information about the client and its location as described in Section 12 and checks its administrative policy about responding to the client. If the server is not permitted to respond to the client, the server discards the Solicit message. For example, if the administrative policy for the server is that it may only respond to a client that is willing to accept a Reconfigure message, if the client does not include a Reconfigure Accept option (see Section 23.20) in the Solicit message, the servers discard the Solicit message.

If the client has included a Rapid Commit option in the Solicit message and the server has been configured to respond with committed address assignments and other resources, the server responds to the Solicit with a Reply message as described in Section 18.2.3. Otherwise, the server ignores the Rapid Commit option and processes the remainder of the message as if no Rapid Commit option were present.

### 18.2.2. Creation and Transmission of Advertise Messages

The server sets the "msg-type" field to ADVERTISE and copies the contents of the transaction-id field from the Solicit message received from the client to the Advertise message. The server includes its server identifier in a Server Identifier option and copies the Client Identifier from the Solicit message into the Advertise message.

The server MAY add a Preference option to carry the preference value for the Advertise message. The server implementation SHOULD allow the setting of a server preference value by the administrator. The server preference value MUST default to zero unless otherwise configured by the server administrator.

The server includes a Reconfigure Accept option if the server wants to require that the client accept Reconfigure messages.

The server includes options the server will return to the client in a subsequent Reply message. The information in these options may be used by the client in the selection of a server if the client receives more than one Advertise message. If the client has included an Option Request option in the Solicit message, the server includes options in the Advertise message containing configuration parameters for all of the options identified in the Option Request option that the server has been configured to return to the client. The server MAY return additional options to the client if it has been configured to do so. The server must be aware of the recommendations on packet sizes and the use of fragmentation in section 5 of [RFC2460].

If the Solicit message from the client included one or more IA options, the server MUST include IA options in the Advertise message containing any addresses that would be assigned to IAs contained in the Solicit message from the client. If the client has included addresses in the IAs in the Solicit message, the server uses those addresses as hints about the addresses the client would like to receive.

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, a Client Identifier option with the client's DUID, and (optionally) SOL\_MAX\_RT and/or INF\_MAX\_RT options. The server SHOULD include other stateful IA options (like IA\_PD) and other configuration options in the Advertise message.

If the Solicit message was received directly by the server, the server unicasts the Advertise message directly to the client using the address in the source address field from the IP datagram in which the Solicit message was received. The Advertise message MUST be unicast on the link from which the Solicit message was received.

If the Solicit message was received in a Relay-forward message, the server constructs a Relay-reply message with the Advertise message in the payload of a "relay-message" option. If the Relay-forward messages included an Interface-id option, the server copies that option to the Relay-reply message. The server unicasts the Relay-reply message directly to the relay agent using the address in the source address field from the IP datagram in which the Relay-forward message was received.

### 18.2.3. Creation and Transmission of Reply Messages

The server **MUST** commit the assignment of any addresses or other configuration information message before sending a Reply message to a client in response to a Solicit message.

#### DISCUSSION:

When using the Solicit-Reply message exchange, the server commits the assignment of any addresses before sending the Reply message. The client can assume it has been assigned the addresses in the Reply message and does not need to send a Request message for those addresses.

Typically, servers that are configured to use the Solicit-Reply message exchange will be deployed so that only one server will respond to a Solicit message. If more than one server responds, the client will only use the addresses from one of the servers, while the addresses from the other servers will be committed to the client but not used by the client.

The server includes a Rapid Commit option in the Reply message to indicate that the Reply is in response to a Solicit message.

The server includes a Reconfigure Accept option if the server wants to require that the client accept Reconfigure messages.

The server produces the Reply message as though it had received a Request message, as described in Section 19.2.1. The server transmits the Reply message as described in Section 19.2.8.

### 18.3. Client behavior for Prefix Delegation

The requesting router creates and transmits a Solicit message as described in Section 18.1.1 and Section 18.1.2. The client creates an IA\_PD and assigns it an IAID. The client **MUST** include the IA\_PD option in the Solicit message.

The client processes any received Advertise messages as described in Section 18.1.3. The client **MAY** choose to consider the presence of advertised prefixes in its decision about which delegating router to respond to.

The client **MUST** ignore any IA\_PDs in an Advertise message that include a Status Code option containing the value NoPrefixAvail, with the exception that the client **MAY** display the associated status message to the user and **SHOULD** process SOL\_MAX\_RT and INF\_MAX\_RT options.

#### 18.4. Server Behavior for Prefix Delegation

The server sends an Advertise message to the requesting router in the same way as described in Section 18.2.2. If the message contains an IA\_PD option and the delegating router is configured to delegate prefix(es) to the requesting router, the delegating router selects the prefix(es) to be delegated to the requesting router. The mechanism through which the delegating router selects prefix(es) for delegation is not specified in this document. Examples of ways in which the server might select prefix(es) for a client include: static assignment based on subscription to an ISP; dynamic assignment from a pool of available prefixes; selection based on an external authority such as a RADIUS server using the Framed-IPv6-Prefix option as described in [RFC3162].

If the client includes an IA\_PD Prefix option in the IA\_PD option in its Solicit message, the server MAY choose to use the information in that option to select the prefix(es) or prefix size to be delegated to the client.

The server sends an Advertise message to the requesting router in the same way as described in Section 18.2.2. The server MUST include an IA\_PD option, identifying any prefix(es) that the server will delegate to the client.

If the server will not assign any prefixes to an IA\_PD in a subsequent Request from the requesting router, the server MUST send an Advertise message to the client that includes the IA\_PD with no prefixes in the IA\_PD and a Status Code option in the IA\_PD containing status code NoPrefixAvail 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. The server SHOULD include other stateful IA options (like IA\_NA) and other configuration options in the Advertise message.

#### 19. DHCP Client-Initiated Configuration Exchange

A client initiates a message exchange with a server or servers to acquire or update configuration information of interest. The client may initiate the configuration exchange as part of the operating system configuration process, when requested to do so by the application layer, when required by Stateless Address Autoconfiguration or as required to extend the lifetime of address(es) or/and delegated prefix(es), using Renew and Rebind messages.

According to a terminology for the prefix delegation, a client requesting a delegation of a prefix is referred to as a requesting



router and a server delegating the prefix is referred to as a delegating router. The requesting router and the delegating router use the IA\_PD Prefix option to exchange information about prefix(es) in much the same way as IA Address options are used for assigned addresses. Typically, a single DHCP session is used to exchange information about addresses and prefixes, i.e. IA\_NA and IA\_PD options are carried in the same message.

#### 19.1. Client Behavior

A client uses Request, Renew, Rebind, Release and Decline messages during the normal life cycle of addresses. It uses Confirm to validate addresses when it may have moved to a new link. It uses Information-Request messages when it needs configuration information but no addresses.

If the client has a source address of sufficient scope that can be used by the server as a return address, and the client has received a Server Unicast option (Section 23.12) from the server, the client SHOULD unicast any Request, Renew, Release and Decline messages to the server.

##### DISCUSSION:

Use of unicast may avoid delays due to the relaying of messages by relay agents, as well as avoid overhead and duplicate responses by servers due to the delivery of client messages to multiple servers. Requiring the client to relay all DHCP messages through a relay agent enables the inclusion of relay agent options in all messages sent by the client. The server should enable the use of unicast only when relay agent options will not be used.

##### 19.1.1. Creation and Transmission of Request Messages

The client uses a Request message to populate IAs with addresses and obtain other configuration information. The client includes one or more IA options in the Request message. The server then returns addresses and other information about the IAs to the client in IA options in a Reply message.

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 other appropriate options,

including one or more IA options (if the client is requesting that the server assign it some network addresses).

The client **MUST** include an Option Request option (see Section 23.7) 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 includes a Reconfigure Accept option (see Section 23.20) indicating whether or not the client is willing to accept Reconfigure messages from the server.

The client transmits the message according to Section 15, using the following parameters:

IRT	REQ_TIMEOUT
MRT	REQ_MAX_RT
MRC	REQ_MAX_RC
MRD	0

If the message exchange fails, the client takes an action based on the client's local policy. Examples of actions the client might take include:

- Select another server from a list of servers known to the client; for example, servers that responded with an Advertise message.
- Initiate the server discovery process described in Section 18.
- Terminate the configuration process and report failure.

#### 19.1.1.2. Creation and Transmission of Confirm Messages

Whenever a client may have moved to a new link, the prefixes/addresses assigned to the interfaces on that link may no longer be appropriate for the link to which the client is attached. Examples of times when a client may have moved to a new link include:

- o The client reboots.
- o The client is physically connected to a wired connection.
- o The client returns from sleep mode.
- o The client using a wireless technology changes access points.

In any situation when a client may have moved to a new link, the client SHOULD initiate a Confirm/Reply message exchange. The client includes any IAs assigned to the interface that may have moved to a new link, along with the addresses associated with those IAs, in its Confirm message. Any responding servers will indicate whether those addresses are appropriate for the link to which the client is attached with the status in the Reply message it returns to the client.

One example when this rule may not be followed is when the client does not store its leases in stable storage and experiences a reboot. It may simply not retain any information, so it does not know what to confirm. In such case client MUST restart server discovery process as described in Section 18.1.1.

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

The client MUST include a Client Identifier option to identify itself to the server. The client includes IA options for all of the IAs assigned to the interface for which the Confirm message is being sent. The IA options include all of the addresses the client currently has associated with those IAs. The client SHOULD set the T1 and T2 fields in any IA\_NA options, and the preferred-lifetime and valid-lifetime fields in the IA Address options to 0, as the server will ignore these fields.

The first Confirm message from the client on the interface MUST be delayed by a random amount of time between 0 and CNF\_MAX\_DELAY. The client transmits the message according to Section 15, using the following parameters:

IRT	CNF_TIMEOUT
MRT	CNF_MAX_RT
MRC	0
MRD	CNF_MAX_RD

If the client receives no responses before the message transmission process terminates, as described in Section 15, the client SHOULD continue to use any IP addresses, using the last known lifetimes for those addresses, and SHOULD continue to use any other previously obtained configuration parameters.

### 19.1.1.3. Creation and Transmission of Renew Messages

To extend the valid and preferred lifetimes for the addresses associated with an IA, the client sends a Renew message to the server from which the client obtained the addresses in the IA containing an IA option for the IA. The client includes IA Address options in the IA option for the addresses associated with the IA. The server determines new lifetimes for the addresses in the IA according to the administrative configuration of the server. The server may also add new addresses to the IA. The server may remove addresses from the IA by setting the preferred and valid lifetimes of those addresses 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.

At time T1 for an IA, the client initiates a Renew/Reply message exchange to extend the lifetimes on any addresses in the IA. The client includes an IA option with all addresses currently assigned to the IA in its Renew message.

If T1 or T2 is set to 0 by the server (for an IA\_NA) or there are no T1 or T2 times (for an IA\_TA), 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. The client MUST include the list of addresses the client currently has associated with the IAs in the Renew message.

The client MUST include an Option Request option (see Section 23.7) 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 15, 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 19.1.4), at which time the client begins a Rebind message exchange.

#### 19.1.4. Creation and Transmission of Rebind Messages

At time T2 for an IA (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 includes an IA option with all addresses currently assigned to the IA in its Rebind message.

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

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. The client MUST include the list of addresses the client currently has associated with the IAs in the Rebind message.

The client MUST include an Option Request option (see Section 23.7) 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 15, using the following parameters:

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

The message exchange is terminated when the valid lifetimes of all the addresses assigned to the IA expire (see Section 11), at which

time the client has several alternative actions to choose from; for example:

- The client may choose to use a Solicit message to locate a new DHCP server and send a Request for the expired IA to the new server.
- The client may have other addresses in other IAs, so the client may choose to discard the expired IA and use the addresses in the other IAs.

#### 19.1.1.5. Creation and Transmission of Information-request Messages

The client uses an Information-request message to obtain configuration information without having addresses assigned to it.

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

The client SHOULD include a Client Identifier option to identify itself to the server. If the client does not include a Client Identifier option, the server will not be able to return any client-specific options to the client, or the server may choose not to respond to the message at all. The client MUST include a Client Identifier option if the Information-Request message will be authenticated.

The client MUST include an Option Request option (see Section 23.7) to request the INF\_MAX\_RT option (see Section 23.24) and any other 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 first Information-request message from the client on the interface MUST be delayed by a random amount of time between 0 and INF\_MAX\_DELAY. The client transmits the message according to Section 15, using the following parameters:

IRT	INF_TIMEOUT
MRT	INF_MAX_RT
MRC	0
MRD	0

#### 19.1.1.6. Creation and Transmission of Release Messages

To release one or more addresses, a client sends a Release message to the server.

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

The client places the identifier of the server that allocated the address(es) in a Server Identifier option.

The client MUST include a Client Identifier option to identify itself to the server. The client includes options containing the IAs for the addresses it is releasing in the "options" field. The addresses to be released MUST be included in the IAs. Any addresses for the IAs the client wishes to continue to use MUST NOT be added to the IAs.

The client MUST NOT use any of the addresses it is releasing as the source address in the Release message or in any subsequently transmitted message.

Because Release messages may be lost, the client should retransmit the Release if no Reply is received. However, there are scenarios where the client may not wish to wait for the normal retransmission timeout before giving up (e.g., on power down). Implementations SHOULD retransmit one or more times, but MAY choose to terminate the retransmission procedure early.

The client transmits the message according to Section 15, using the following parameters:

IRT	REL_TIMEOUT
MRT	0
MRC	REL_MAX_RC
MRD	0

The client MUST stop using all of the addresses being released as soon as the client begins the Release message exchange process. If addresses are released but the Reply from a DHCP server is lost, the client will retransmit the Release message, and the server may respond with a Reply indicating a status of NoBinding. Therefore, the client does not treat a Reply message with a status of NoBinding in a Release message exchange as if it indicates an error.

Note that if the client fails to release the addresses, each address assigned to the IA will be reclaimed by the server when the valid lifetime of that address expires.

#### 19.1.7. Creation and Transmission of Decline Messages

If a client detects that one or more addresses assigned to it by a server are already in use by another node, the client sends a Decline message to the server to inform it that the address is suspect.

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

The client places the identifier of the server that allocated the address(es) in a Server Identifier option.

The client MUST include a Client Identifier option to identify itself to the server. The client includes options containing the IAs for the addresses it is declining in the "options" field. The addresses to be declined MUST be included in the IAs. Any addresses for the IAs the client wishes to continue to use should not be included to the IAs.

The client MUST NOT use any of the addresses it is declining as the source address in the Decline message or in any subsequently transmitted message.

The client transmits the message according to Section 15, using the following parameters:

IRT	DEC_TIMEOUT
MRT	0
MRC	DEC_MAX_RC
MRD	0

If addresses are declined but the Reply from a DHCP server is lost, the client will retransmit the Decline message, and the server may respond with a Reply indicating a status of NoBinding. Therefore, the client does not treat a Reply message with a status of NoBinding in a Decline message exchange as if it indicates an error.



#### 19.1.1.8. Receipt of Reply Messages

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.

The client SHOULD perform duplicate address detection [RFC4862] on each of the addresses in any IAs it receives in the Reply message before using that address for traffic. If any of the addresses are found to be in use on the link, the client sends a Decline message to the server as described in Section 19.1.7.

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 23.

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 (see Section 14.1).

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 18, and client-initiated configuration as described in Section 19. 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 Solicit (with a Rapid Commit option) or a Request, the client can either re-issue the Request without specifying any addresses or restart the DHCP server discovery process (see Section 18).

The client examines the status code in each IA individually. If the status code is NoAddrsAvail, the client has received no usable addresses in the IA and may choose to try obtaining addresses for the IA from another server. The client uses addresses and other information from any IAs that do not contain a Status Code option with the NoAddrsAvail code. If the client receives no 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.

Whenever a client restarts the DHCP server discovery process or selects an alternate server, as described in Section 18.1.3, the client SHOULD stop using all the addresses and delegated prefixes for which it has the bindings and try to obtain all required addresses and prefixes from the new server. This facilitates the client using a single state machine for all bindings.

When the client receives a Reply message in response to a Renew or Rebind message, the client examines each IA independently. For each IA in the original Renew or Rebind message, the client:

- sends a Request message if the IA contained a Status Code option with the NoBinding status (and does not send any additional Renew/Rebind messages)
- sends a Renew/Rebind if the IA is not in the Reply message
- 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.

## 19.2. Server Behavior

For this discussion, the Server is assumed to have been configured in an implementation specific manner with configuration of interest to clients.

In most instances, the server will send a Reply in response to a client message. This Reply message **MUST** always contain the Server Identifier option containing the server's DUID and the Client Identifier option from the client message if one was present.

In most Reply messages, the server includes options containing configuration information for the client. The server must be aware of the recommendations on packet sizes and the use of fragmentation in section 5 of [RFC2460]. If the client included an Option Request option in its message, the server includes options in the Reply message containing configuration parameters for all of the options identified in the Option Request option that the server has been configured to return to the client. The server **MAY** return additional options to the client if it has been configured to do so.

### 19.2.1. Receipt of Request Messages

When the server receives a Request message via unicast from a client to which the server has not sent a unicast option, the server discards the Request 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.

When the server receives a valid Request message, the server creates the bindings for that client according to the server's policy and configuration information and records the IAs and other information requested by the client.

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

If the server finds that the prefix on one or more IP addresses in any IA in the message from the client is not appropriate for the link to which the client is connected, the server MUST return the IA to the client with a Status Code option with the value NotOnLink.

If the server cannot assign any addresses to an IA in the message from the client, the server MUST include the IA in the Reply message with no addresses in the IA and a Status Code option in the IA containing status code NoAddrsAvail.

For any IAs to which the server can assign addresses, the server includes the IA with addresses and other configuration parameters, and records the IA as a new client binding.

The server includes a Reconfigure Accept option if the server wants to require that the client accept Reconfigure messages.

The server includes other options containing configuration information to be returned to the client as described in Section 19.2.

If the server finds that the client has included an IA in the Request message for which the server already has a binding that associates the IA with the client, the client has resent a Request message for which it did not receive a Reply message. The server either resends a previously cached Reply message or sends a new Reply message.

#### 19.2.2. Receipt of Confirm Messages

When the server receives a Confirm message, the server determines whether the addresses in the Confirm message are appropriate for the link to which the client is attached. If all of the addresses in the Confirm message pass this test, the server returns a status of Success. If any of the addresses do not pass this test, the server returns a status of NotOnLink. If the server is unable to perform this test (for example, the server does not have information about prefixes on the link to which the client is connected), or there were no addresses in any of the IAs sent by the client, the server MUST NOT send a reply to the client.

The server ignores the T1 and T2 fields in the IA options and the preferred-lifetime and valid-lifetime fields in the IA Address options.

The server constructs a Reply message by setting the "msg-type" field to REPLY, and copying the transaction ID from the Confirm 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 Confirm message in the Reply message. The server includes a Status Code option indicating the status of the Confirm message.

#### 19.2.3. Receipt of Renew Messages

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.

When the server receives a Renew 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 cannot find a client entry for the IA the server returns the IA containing no addresses with a Status Code option set to NoBinding in the Reply message.

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

If the server finds the addresses in the IA for the client then the server sends back the IA to the client with new lifetimes and T1/T2 times. The server may choose to change the list of addresses and the lifetimes of addresses in IAs that are returned to the client.

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 19.2.

#### 19.2.4. Receipt of Rebind Messages

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 cannot find a client entry for the IA 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 zero. 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 discards the Rebind message.

If the server finds that any of the addresses are no longer appropriate for the link to which the client is attached, the server returns the address to the client with lifetimes of 0.

If the server finds the addresses in the IA for the client then the server SHOULD send back the IA to the client with new lifetimes and T1/T2 times.

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 19.2.

#### 19.2.5. Receipt of Information-request Messages

When the server receives an Information-request message, the client is requesting configuration information that does not include the assignment of any addresses. The server determines all configuration parameters appropriate to the client, based on the server configuration policies known to the server.

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

The server MUST include a Server Identifier option containing the server's DUID in the Reply message. If the client included a Client Identification option in the Information-request message, the server copies that option to the Reply message.

The server includes options containing configuration information to be returned to the client as described in Section 19.2.

If the Information-request message received from the client did not include a Client Identifier option, the server SHOULD respond with a Reply message containing any configuration parameters that are not determined by the client's identity. If the server chooses not to respond, the client may continue to retransmit the Information-request message indefinitely.

#### 19.2.6. Receipt of Release Messages

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

Upon the receipt of a valid Release message, the server examines the IAs and the addresses in the IAs for validity. If the IAs in the message are in a binding for the client, and the addresses in the IAs have been assigned by the server to those IAs, the server deletes the addresses from the IAs and makes the addresses available for assignment to other clients. The server ignores addresses not assigned to the IA, although it may choose to log an error.

After all the addresses have been processed, the server generates a Reply message and includes a Status Code option with value Success, a Server Identifier option with the server's DUID, and a Client Identifier option with the client's DUID. For each IA in the Release message for which the server has no binding information, the server adds an IA option using the IAID from the Release message, and includes a Status Code option with the value NoBinding in the IA option. No other options are included in the IA option.

A server may choose to retain a record of assigned addresses and IAs after the lifetimes on the addresses have expired to allow the server to reassign the previously assigned addresses to a client.

#### 19.2.7. Receipt of Decline Messages

When the server receives a Decline message via unicast from a client to which the server has not sent a unicast option, the server discards the Decline 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.

Upon the receipt of a valid Decline message, the server examines the IAs and the addresses in the IAs for validity. If the IAs in the message are in a binding for the client, and the addresses in the IAs have been assigned by the server to those IAs, the server deletes the addresses from the IAs. The server ignores addresses not assigned to the IA (though it may choose to log an error if it finds such an address).

The client has found any addresses in the Decline messages to be already in use on its link. Therefore, the server SHOULD mark the addresses declined by the client so that those addresses are not assigned to other clients, and MAY choose to make a notification that addresses were declined. Local policy on the server determines when the addresses identified in a Decline message may be made available for assignment.

After all the addresses have been processed, the server generates a Reply message and includes a Status Code option with the value Success, a Server Identifier option with the server's DUID, and a Client Identifier option with the client's DUID. For each IA in the Decline message for which the server has no binding information, the server adds an IA option using the IAID from the Decline message and includes a Status Code option with the value NoBinding in the IA option. No other options are included in the IA option.

#### 19.2.8. Transmission of Reply Messages

If the original message was received directly by the server, the server unicasts the Reply message directly to the client using the address in the source address field from the IP datagram in which the original message was received. The Reply message MUST be unicast through the interface on which the original message was received.

If the original message was received in a Relay-forward message, the server constructs a Relay-reply message with the Reply message in the payload of a Relay Message option (see Section 23.10). If the Relay-forward messages included an Interface-id option, the server copies that option to the Relay-reply message. The server unicasts the Relay-reply message directly to the relay agent using the address in



the source address field from the IP datagram in which the Relay-forward message was received.

### 19.3. Requesting Router Behavior for Prefix Delegation

The requesting router uses a Request message to populate IA\_PDs with prefixes. The requesting router includes one or more IA\_PD options in the Request message. The delegating router then returns the prefixes for the IA\_PDs to the requesting router in IA\_PD options in a Reply message.

The requesting router includes IA\_PD options in any Renew, or Rebind messages sent by the requesting router. The IA\_PD option includes all of the prefixes the requesting router currently has associated with that IA\_PD.

In some circumstances the requesting router may need verification that the delegating router still has a valid binding for the requesting router. Examples of times when a requesting router may ask for such verification include:

- o The requesting router reboots.
- o The requesting router's upstream link flaps.
- o The requesting router is physically disconnected from a wired connection.

If such verification is needed the requesting router MUST initiate a Rebind/Reply message exchange as described in section Section 19.1.4, with the exception that the retransmission parameters should be set as for the Confirm message, described in Section 19.1.2. The requesting router includes any IA\_PDs, along with prefixes associated with those IA\_PDs in its Rebind message.

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 uses a Release message to return a delegated prefix to a delegating router. The prefixes to be released MUST be included in the IA\_PDs.

The Confirm and Decline message types are not used with Prefix Delegation.

Upon the receipt of a valid Reply message, for each IA\_PD the requesting router assigns a subnet from each of the delegated prefixes to each of the links to which the associated interfaces are attached.

When the Delegating Router delegates prefixes to a Requesting Router, the Requesting Router has sole authority for assignment of those prefixes, and the Delegating Router MUST NOT assign any prefixes from that delegated prefix to any of its own links.

When a requesting router subnets a delegated prefix, it must assign additional bits to the prefix to generate unique, longer prefixes. For example, if the requesting router in Figure 1 were delegated 3FFE:FFFF:0::/48, it might generate 3FFE:FFFF:0:1::/64 and 3FFE:FFFF:0:2::/64 for assignment to the two links in the subscriber network. If the requesting router were delegated 3FFE:FFFF:0::/48 and 3FFE:FFFF:5::/48, it might assign 3FFE:FFFF:0:1::/64 and 3FFE:FFFF:5:1::/64 to one of the links, and 3FFE:FFFF:0:2::/64 and 3FFE:FFFF:5:2::/64 for assignment to the other link.

If the requesting router assigns a delegated prefix to a link to which the router is attached, and begins to send router advertisements for the prefix on the link, the requesting router MUST set the valid lifetime in those advertisements to be no later than the valid lifetime specified in the IA\_PD Prefix option. A requesting router MAY use the preferred lifetime specified in the IA\_PD Prefix option.

Handling of Status Codes options in received Reply messages is described in section Section 19.1.8. The NoPrefixAvail Status Code is handled in the same manner as the NoAddrsAvail Status Code.

#### 19.4. Delegating Router Behavior for Prefix Delegation

When a delegating router receives a Request message from a requesting router that contains an IA\_PD option, and the delegating router is authorized to delegate prefix(es) to the requesting router, the delegating router selects the prefix(es) to be delegated to the requesting router. The mechanism through which the delegating router selects prefix(es) for delegation is not specified in this document. Section 18.4 gives examples of ways in which a delegating router might select the prefix(es) to be delegated to a requesting router.

A delegating router examines the prefix(es) identified in IA\_PD Prefix options (in an IA\_PD option) in Renew and Rebind messages and responds according to the current status of the prefix(es). The delegating router returns IA\_PD Prefix options (within an IA\_PD option) with updated lifetimes for each valid prefix in the message

from the requesting router. If the delegating router finds that any of the prefixes are not in the requesting router's binding entry, the delegating router returns the prefix to the requesting router with lifetimes of 0.

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

Renew message: If the delegating router cannot find a binding for the requesting router's IA\_PD the delegating router returns the IA\_PD containing no prefixes with a Status Code option set to NoBinding in the Reply message.

Rebind message: If the delegating router cannot find a binding for the requesting router's IA\_PD and the delegating router determines that the prefixes in the IA\_PD are not appropriate for the link to which the requesting router's interface is attached according to the delegating routers explicit configuration, the delegating router MAY send a Reply message to the requesting router containing the IA\_PD with the lifetimes of the prefixes in the IA\_PD set to zero. This Reply constitutes an explicit notification to the requesting router that the prefixes in the IA\_PD are no longer valid. If the delegating router is unable to determine if the prefix is not appropriate for the link, the Rebind message is discarded.

A delegating router may mark any prefix(es) in IA\_PD Prefix options in a Release message from a requesting router as "available", dependent on the mechanism used to acquire the prefix, e.g., in the case of a dynamic pool.

The delegating router MUST include an IA\_PD Prefix option or options (in an IA\_PD option) in Reply messages sent to a requesting router.

## 20. DHCP Server-Initiated Configuration Exchange

A server initiates a configuration exchange to cause DHCP clients to obtain new addresses and other configuration information. For example, an administrator may use a server-initiated configuration exchange when links in the DHCP domain are to be renumbered. Other examples include changes in the location of directory servers, addition of new services such as printing, and availability of new software.

## 20.1. Server Behavior

A server sends a Reconfigure message to cause a client to initiate immediately a Renew/Reply or Information-request/Reply message exchange with the server.

### 20.1.1. Creation and Transmission of Reconfigure Messages

The server sets the "msg-type" field to RECONFIGURE. The server sets the transaction-id field to 0. The server includes a Server Identifier option containing its DUID and a Client Identifier option containing the client's DUID in the Reconfigure message.

The server MAY include an Option Request option to inform the client of what information has been changed or new information that has been added. In particular, the server specifies the IA option in the Option Request option if the server wants the client to obtain new address information. If the server identifies the IA option in the Option Request option, the server MUST include an IA option to identify each IA that is to be reconfigured on the client. The IA options included by the server MUST NOT contain any options.

Because of the risk of denial of service attacks against DHCP clients, the use of a security mechanism is mandated in Reconfigure messages. The server MUST use DHCP authentication in the Reconfigure message.

The server MUST include a Reconfigure Message option (defined in Section 23.19) to select whether the client responds with a Renew message, a Rebind message, or an Information-Request message.

The server MUST NOT include any other options in the Reconfigure except as specifically allowed in the definition of individual options.

A server sends each Reconfigure message to a single DHCP client, using an IPv6 unicast address of sufficient scope belonging to the DHCP client. If the server does not have an address to which it can send the Reconfigure message directly to the client, the server uses a Relay-reply message (as described in Section 21.3) to send the Reconfigure message to a relay agent that will relay the message to the client. The server may obtain the address of the client (and the appropriate relay agent, if required) through the information the server has about clients that have been in contact with the server, or through some external agent.

To reconfigure more than one client, the server unicasts a separate message to each client. The server may initiate the reconfiguration

of multiple clients concurrently; for example, a server may send a Reconfigure message to additional clients while previous reconfiguration message exchanges are still in progress.

The Reconfigure message causes the client to initiate a Renew/Reply, a Rebind/Reply, or Information-request/Reply message exchange with the server. The server interprets the receipt of a Renew, a Rebind, or Information-request message (whichever was specified in the original Reconfigure message) from the client as satisfying the Reconfigure message request.

#### 20.1.2. Time Out and Retransmission of Reconfigure Messages

If the server does not receive a Renew, Rebind, or Information-request message from the client in REC\_TIMEOUT milliseconds, the server retransmits the Reconfigure message, doubles the REC\_TIMEOUT value and waits again. The server continues this process until REC\_MAX\_RC unsuccessful attempts have been made, at which point the server SHOULD abort the reconfigure process for that client.

Default and initial values for REC\_TIMEOUT and REC\_MAX\_RC are documented in Section 6.5.

#### 20.2. Receipt of Renew or Rebind Messages

In response to a Renew message, the server generates and sends a Reply message to the client as described in Section 19.2.3 and Section 19.2.8, including options for configuration parameters.

In response to a Rebind message, the server generates and sends a Reply message to the client as described in Section 19.2.4 and Section 19.2.8, including options for configuration parameters.

The server MAY include options containing the IAs and new values for other configuration parameters in the Reply message, even if those IAs and parameters were not requested in the Renew or Rebind message from the client.

#### 20.3. Receipt of Information-request Messages

The server generates and sends a Reply message to the client as described in Section 19.2.5 and Section 19.2.8, including options for configuration parameters.

The server MAY include options containing new values for other configuration parameters in the Reply message, even if those parameters were not requested in the Information-request message from the client.

## 20.4. Client Behavior

A client receives Reconfigure messages sent to the UDP port 546 on interfaces for which it has acquired configuration information through DHCP. These messages may be sent at any time. Since the results of a reconfiguration event may affect application layer programs, the client **SHOULD** log these events, and **MAY** notify these programs of the change through an implementation-specific interface.

### 20.4.1. Receipt of Reconfigure Messages

Upon receipt of a valid Reconfigure message, the client responds with either a Renew message, a Rebind message, or an Information-request message as indicated by the Reconfigure Message option (as defined in Section 23.19). The client ignores the transaction-id field in the received Reconfigure message. While the transaction is in progress, the client discards any Reconfigure messages it receives.

#### DISCUSSION:

The Reconfigure message acts as a trigger that signals the client to complete a successful message exchange. Once the client has received a Reconfigure, the client proceeds with the message exchange (retransmitting the Renew or Information-request message if necessary); the client ignores any additional Reconfigure messages until the exchange is complete. Subsequent Reconfigure messages cause the client to initiate a new exchange.

How does this mechanism work in the face of duplicated or retransmitted Reconfigure messages? Duplicate messages will be ignored because the client will begin the exchange after the receipt of the first Reconfigure. Retransmitted messages will either trigger the exchange (if the first Reconfigure was not received by the client) or will be ignored. The server can discontinue retransmission of Reconfigure messages to the client once the server receives the Renew or Information-request message from the client.

It might be possible for a duplicate or retransmitted Reconfigure to be sufficiently delayed (and delivered out of order) to arrive at the client after the exchange (initiated by the original Reconfigure) has been completed. In this case, the client would initiate a redundant exchange. The likelihood of delayed and out of order delivery is small enough to be ignored. The consequence of the redundant exchange is inefficiency rather than incorrect operation.

#### 20.4.2. Creation and Transmission of Renew or Rebind Messages

When responding to a Reconfigure, the client creates and sends the Renew message in exactly the same manner as outlined in Section 19.1.3, with the exception that the client copies the Option Request option and any IA options from the Reconfigure message into the Renew message. The client **MUST** include a Server Identifier option in the Renew message, identifying the server with which the client most recently communicated.

When responding to a Reconfigure, the client creates and sends the Rebind message in exactly the same manner as outlined in Section 19.1.4, with the exception that the client copies the Option Request option and any IA options from the Reconfigure message into the Rebind message.

If a client is currently sending Rebind messages, as described in Section 19.1.3, the client ignores any received Reconfigure messages.

#### 20.4.3. Creation and Transmission of Information-request Messages

When responding to a Reconfigure, the client creates and sends the Information-request message in exactly the same manner as outlined in Section 19.1.5, with the exception that the client includes a Server Identifier option with the identifier from the Reconfigure message to which the client is responding.

#### 20.4.4. Time Out and Retransmission of Renew, Rebind or Information-request Messages

The client uses the same variables and retransmission algorithm as it does with Renew, Rebind, or Information-request messages generated as part of a client-initiated configuration exchange. See Section 19.1.3, Section 19.1.4, and Section 19.1.5 for details. If the client does not receive a response from the server by the end of the retransmission process, the client ignores and discards the Reconfigure message.

#### 20.4.5. Receipt of Reply Messages

Upon the receipt of a valid Reply message, the client processes the options and sets (or resets) configuration parameters appropriately. The client records and updates the lifetimes for any addresses specified in IAs in the Reply message.

## 20.5. Prefix Delegation Reconfiguration

This section describes prefix delegation in Reconfigure message exchanges.

### 20.5.1. Delegating Router Behavior

The delegating router initiates a configuration message exchange with a requesting router, as described in Section 20, by sending a Reconfigure message (acting as a DHCP server) to the requesting router, as described in Section 20.1. The delegating router specifies the IA\_PD option in the Option Request option to cause the requesting router to include an IA\_PD option to obtain new information about delegated prefix(es).

### 20.5.2. Requesting Router Behavior

The requesting router responds to a Reconfigure message, acting as a DHCP client, received from a delegating router as described in Section 20.4. The requesting router MUST include the IA\_PD Prefix option(s) (in an IA\_PD option) for prefix(es) that have been delegated to the requesting router by the delegating router from which the Reconfigure message was received.

## 21. Relay Agent Behavior

The relay agent MAY be configured to use a list of destination addresses, which MAY include unicast addresses, the All\_DHCP\_Servers multicast address, or other addresses selected by the network administrator. If the relay agent has not been explicitly configured, it MUST use the All\_DHCP\_Servers multicast address as the default.

If the relay agent relays messages to the All\_DHCP\_Servers multicast address or other multicast addresses, it sets the Hop Limit field to 32.

If the relay agent receives a message other than Relay-forward and Relay-reply and the relay agent does not recognize its message type, it MUST forward them as described in Section 21.1.1.

### 21.1. Relaying a Client Message or a Relay-forward Message

A relay agent relays both messages from clients and Relay-forward messages from other relay agents. When a relay agent receives a valid message (for a definition of a valid message, see Section 4.1 of [RFC7283]) to be relayed, it constructs a new Relay-forward message. The relay agent copies the source address from the header



of the IP datagram in which the message was received to the peer-address field of the Relay-forward message. The relay agent copies the received DHCP message (excluding any IP or UDP headers) into a Relay Message option in the new message. The relay agent adds to the Relay-forward message any other options it is configured to include.

[RFC6221] defines a Lightweight DHCPv6 Relay Agent (LDRA) that allows Relay Agent Information to be inserted by an access node that performs a link-layer bridging (i.e., non-routing) function.

#### 21.1.1. Relaying a Message from a Client

If the relay agent received the message to be relayed from a client, the relay agent places a global, ULA [RFC4193] or site-scoped address with a prefix assigned to the link on which the client should be assigned an address in the link-address field. (It is possible for the relay to use link local address instead, but that is not recommended as it would require additional information to be provided in the server configuration. See Section 3.2 of [I-D.ietf-dhc-topo-conf] for detailed discussion.) This address will be used by the server to determine the link from which the client should be assigned an address and other configuration information. The hop-count in the Relay-forward message is set to 0.

If the relay agent cannot use the address in the link-address field to identify the interface through which the response to the client will be relayed, the relay agent MUST include an Interface-id option (see Section 23.18) in the Relay-forward message. The server will include the Interface-id option in its Relay-reply message. The relay agent fills in the link-address field as described in the previous paragraph regardless of whether the relay agent includes an Interface-id option in the Relay-forward message.

#### 21.1.2. Relaying a Message from a Relay Agent

If the message received by the relay agent is a Relay-forward message and the hop-count in the message is greater than or equal to HOP\_COUNT\_LIMIT, the relay agent discards the received message.

The relay agent copies the source address from the IP datagram in which the message was received from the relay agent into the peer-address field in the Relay-forward message and sets the hop-count field to the value of the hop-count field in the received message incremented by 1.

If the source address from the IP datagram header of the received message is a global or site-scoped address (and the device on which the relay agent is running belongs to only one site), the relay agent

sets the link-address field to 0; otherwise the relay agent sets the link-address field to a global or site-scoped address assigned to the interface on which the message was received, or includes an Interface-ID option to identify the interface on which the message was received.

#### 21.1.3. Relay Agent Behavior with Prefix Delegation

A relay agent forwards messages containing Prefix Delegation options in the same way as described earlier in this section.

If a delegating router communicates with a requesting router through a relay agent, the delegating router may need a protocol or other out-of-band communication to configure routing information for delegated prefixes on any router through which the requesting router may forward traffic.

#### 21.2. Relaying a Relay-reply Message

The relay agent processes any options included in the Relay-reply message in addition to the Relay Message option, and then discards those options.

The relay agent extracts the message from the Relay Message option and relays it to the address contained in the peer-address field of the Relay-reply message. Relay agents **MUST NOT** modify the message.

If the Relay-reply message includes an Interface-id option, the relay agent relays the message from the server to the client on the link identified by the Interface-id option. Otherwise, if the link-address field is not set to zero, the relay agent relays the message on the link identified by the link-address field.

If the relay agent receives a Relay-reply message, it **MUST** process the message as defined above, regardless of the type of message encapsulated in the Relay Message option.

#### 21.3. Construction of Relay-reply Messages

A server uses a Relay-reply message to return a response to a client if the original message from the client was relayed to the server in a Relay-forward message or to send a Reconfigure message to a client if the server does not have an address it can use to send the message directly to the client.

A response to the client **MUST** be relayed through the same relay agents as the original client message. The server causes this to happen by creating a Relay-reply message that includes a Relay

Message option containing the message for the next relay agent in the return path to the client. The contained Relay-reply message contains another Relay Message option to be sent to the next relay agent, and so on. The server must record the contents of the peer-address fields in the received message so it can construct the appropriate Relay-reply message carrying the response from the server.

For example, if client C sent a message that was relayed by relay agent A to relay agent B and then to the server, the server would send the following Relay-Reply message to relay agent B:

```
msg-type:      RELAY-REPLY
hop-count:     1
link-address:   0
peer-address:   A
Relay Message option, containing:
  msg-type:     RELAY-REPLY
  hop-count:    0
  link-address: address from link to which C is attached
  peer-address: C
  Relay Message option: <response from server>
```

Figure 8: Relay-reply Example

When sending a Reconfigure message to a client through a relay agent, the server creates a Relay-reply message that includes a Relay Message option containing the Reconfigure message for the next relay agent in the return path to the client. The server sets the peer-address field in the Relay-reply message header to the address of the client, and sets the link-address field as required by the relay agent to relay the Reconfigure message to the client. The server obtains the addresses of the client and the relay agent through prior interaction with the client or through some external mechanism.

## 22. Authentication of DHCP Messages

Some network administrators may wish to provide authentication of the source and contents of DHCP messages. For example, clients may be subject to denial of service attacks through the use of bogus DHCP servers, or may simply be misconfigured due to unintentionally instantiated DHCP servers. Network administrators may wish to constrain the allocation of addresses to authorized hosts to avoid denial of service attacks in "hostile" environments where the network medium is not physically secured, such as wireless networks or college residence halls.

The DHCP authentication mechanism is based on the design of authentication for DHCPv4 [RFC3118].

#### 22.1. Security of Messages Sent Between Servers and Relay Agents

Relay agents and servers that exchange messages securely use the IPsec mechanisms for IPv6 [RFC4301]. If a client message is relayed through multiple relay agents, each of the relay agents must have established independent, pairwise trust relationships. That is, if messages from client C will be relayed by relay agent A to relay agent B and then to the server, relay agents A and B must be configured to use IPsec for the messages they exchange, and relay agent B and the server must be configured to use IPsec for the messages they exchange.

Relay agents and servers that support secure relay agent to server or relay agent to relay agent communication use IPsec under the following conditions:

Selectors	Relay agents are manually configured with the addresses of the relay agent or server to which DHCP messages are to be forwarded. Each relay agent and server that will be using IPsec for securing DHCP messages must also be configured with a list of the relay agents to which messages will be returned. The selectors for the relay agents and servers will be the pairs of addresses defining relay agents and servers that exchange DHCP messages on DHCPv6 UDP port 547.
Mode	Relay agents and servers use transport mode and ESP. The information in DHCP messages is not generally considered confidential, so encryption need not be used (i.e., NULL encryption can be used).
Key management	Because the relay agents and servers are used within an organization, public key schemes are not necessary. Because the relay agents and servers must be manually configured, manually configured key management may suffice, but does not provide defense against replayed messages. Accordingly, IKE with preshared secrets SHOULD be supported. IKE with public keys MAY be supported.

Security policy	DHCP messages between relay agents and servers should only be accepted from DHCP peers as identified in the local configuration.
Authentication	Shared keys, indexed to the source IP address of the received DHCP message, are adequate in this application.
Availability	Appropriate IPsec implementations are likely to be available for servers and for relay agents in more featureful devices used in enterprise and core ISP networks. IPsec is less likely to be available for relay agents in low end devices primarily used in the home or small office markets.

## 22.2. Summary of DHCP Authentication

Authentication of DHCP messages is accomplished through the use of the Authentication option (see Section 23.11). The authentication information carried in the Authentication option can be used to reliably identify the source of a DHCP message and to confirm that the contents of the DHCP message have not been tampered with.

The Authentication option provides a framework for multiple authentication protocols. Two such protocols are defined here. Other protocols defined in the future will be specified in separate documents.

Any DHCP message MUST NOT include more than one Authentication option.

The protocol field in the Authentication option identifies the specific protocol used to generate the authentication information carried in the option. The algorithm field identifies a specific algorithm within the authentication protocol; for example, the algorithm field specifies the hash algorithm used to generate the message authentication code (MAC) in the authentication option. The replay detection method (RDM) field specifies the type of replay detection used in the replay detection field.

## 22.3. Replay Detection

The Replay Detection Method (RDM) field determines the type of replay detection used in the Replay Detection field.

If the RDM field contains 0x00, the replay detection field MUST be set to the value of a strictly monotonically increasing counter. Using a counter value, such as the current time of day (for example, an NTP-format timestamp [RFC5905]), can reduce the danger of replay attacks. This method MUST be supported by all protocols.

#### 22.4. Delayed Authentication Protocol

If the protocol field is 2, the message is using the "delayed authentication" mechanism. In delayed authentication, the client requests authentication in its Solicit message, and the server replies with an Advertise message that includes authentication information. This authentication information contains a nonce value generated by the source as a message authentication code (MAC) to provide message authentication and entity authentication.

Note that the delayed authentication protocol cannot work with 2-message exchange model. This protocol uses Solicit/Advertise exchange as the key and server selection process. So, real DHCPv6 procedures can only be made in the follow-up messages.

The use of a particular technique based on the HMAC protocol [RFC2104] using the MD5 hash [RFC1321] is defined here.

##### 22.4.1. Use of the Authentication Option in the Delayed Authentication Protocol

In a Solicit message, the client fills in the protocol, algorithm and RDM fields in the Authentication option with the client's preferences. The client sets the replay detection field to zero and omits the authentication information field. The client sets the option-len field to 11.

In all other messages, the protocol and algorithm fields identify the method used to construct the contents of the authentication information field. The RDM field identifies the method used to construct the contents of the replay detection field.

The format of the Authentication information is:

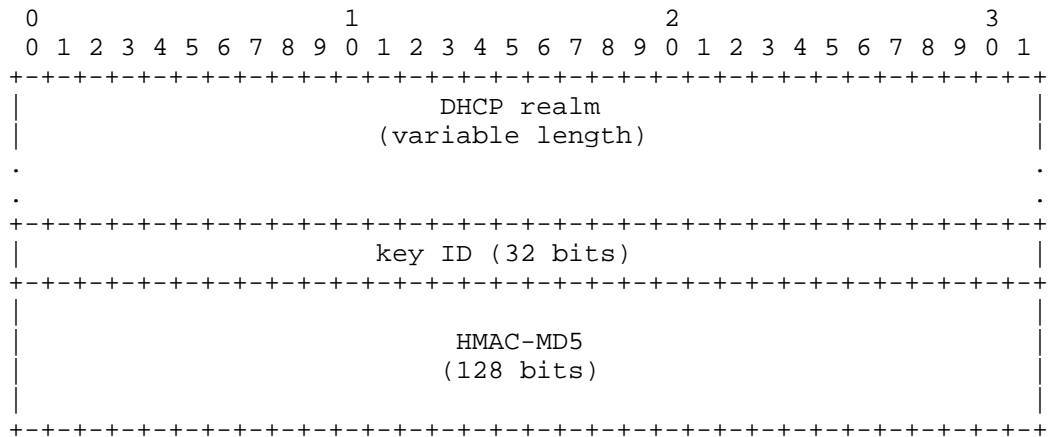


Figure 9: Authentication information format

DHCP realm	The DHCP realm that identifies the key used to generate the HMAC-MD5 value. This is a domain name encoded as described in Section 9.
key ID	The key identifier that identified the key used to generate the HMAC-MD5 value.
HMAC-MD5	The message authentication code generated by applying MD5 to the DHCP message using the key identified by the DHCP realm, client DUID, and key ID.

The sender computes the MAC using the HMAC generation algorithm [RFC2104] and the MD5 hash function [RFC1321]. The entire DHCP message (setting the MAC field of the authentication option to zero), including the DHCP message header and the options field, is used as input to the HMAC-MD5 computation function.

#### DISCUSSION:

Algorithm 1 specifies the use of HMAC-MD5. Use of a different technique, such as HMAC-SHA, will be specified as a separate protocol.

The DHCP realm used to identify authentication keys is chosen to be unique among administrative domains. Use of the DHCP realm allows DHCP administrators to avoid conflict in the use of key

identifiers, and allows a host using DHCP to use authenticated DHCP while roaming among DHCP administrative domains.

#### 22.4.2. Message Validation

Any DHCP message that includes more than one authentication option MUST be discarded.

To validate an incoming message, the receiver first checks that the value in the replay detection field is acceptable according to the replay detection method specified by the RDM field. If no replay is detected, then the receiver computes the MAC as described in [RFC2104]. The entire DHCP message (setting the MAC field of the authentication option to 0) is used as input to the HMAC-MD5 computation function. If the MAC computed by the receiver does not match the MAC contained in the authentication option, the receiver MUST discard the DHCP message.

#### 22.4.3. Key Utilization

Each DHCP client has a set of keys. Each key is identified by <DHCP realm, client DUID, key id>. Each key also has a lifetime. The key may not be used past the end of its lifetime. The client's keys are initially distributed to the client through some out-of-band mechanism. The lifetime for each key is distributed with the key. Mechanisms for key distribution and lifetime specification are beyond the scope of this document.

The client and server use one of the client's keys to authenticate DHCP messages during a session (until the next Solicit message sent by the client).

#### 22.4.4. Client Considerations for Delayed Authentication Protocol

The client announces its intention to use DHCP authentication by including an Authentication option in its Solicit message. The server selects a key for the client based on the client's DUID. The client and server use that key to authenticate all DHCP messages exchanged during the session.

##### 22.4.4.1. Sending Solicit Messages

When the client sends a Solicit message and wishes to use authentication, it includes an Authentication option with the desired protocol, algorithm and RDM as described in Section 22.4. The client does not include any replay detection or authentication information in the Authentication option.



#### 22.4.4.2. Receiving Advertise Messages

The client validates any Advertise messages containing an Authentication option specifying the delayed authentication protocol using the validation test described in Section 22.4.2.

The Client behavior is defined by local policy, as detailed below.

If the client requires that Advertise messages be authenticated, then it **MUST** ignore Advertise messages that do not include authentication information, or for which the client has no matching key, or that do not pass the validation test.

Local policy **MAY** also prefer authenticated Advertise messages, in which case the client **SHOULD** attempt to validate all Advertise messages for which the client has a matching key. Messages for which the client has a key, but which do not pass the validation test **MUST** be rejected, even if the client would otherwise accept the same message without the Authentication option.

In all cases, messages for which the client does not have a matching key should be treated as if they have no Authentication option.

When the decision to accept unauthenticated message is made, it should be made with care. Accepting an unauthenticated Advertise message can make the client vulnerable to spoofing and other attacks. Policies and actions which were depending upon Authentication **MUST** NOT be executed. Local users **SHOULD** be informed that the client has accepted an unauthenticated Advertise message.

A client **MUST** be configurable to discard unauthenticated messages, and **SHOULD** be configured by default to discard unauthenticated messages if the client has been configured with an authentication key or other authentication information.

A client **MAY** choose to differentiate between Advertise messages with no authentication information and Advertise messages that do not pass the validation test; for example, a client might accept the former and discard the latter. If a client does accept an unauthenticated message, the client **SHOULD** inform any local users and **SHOULD** log the event.

#### 22.4.4.3. Sending Request, Confirm, Renew, Rebind, Decline or Release Messages

If the client authenticated the Advertise message through which the client selected the server, the client **MUST** generate authentication information for subsequent Request, Confirm, Renew, Rebind or Release

messages sent to the server, as described in Section 22.4. When the client sends a subsequent message, it MUST use the same key used by the server to generate the authentication information.

#### 22.4.4.4. Sending Information-request Messages

If the server has selected a key for the client in a previous message exchange (see Section 22.4.5.1), the client MUST use the same key to generate the authentication information throughout the session.

#### 22.4.4.5. Receiving Reply Messages

If the client authenticated the Advertise it accepted, the client MUST validate the associated Reply message from the server. The client MUST ignore and discard the Reply if the message fails to pass the validation test and MAY log the validation failure.

If the client accepted an Advertise message that did not include authentication information or did not pass the validation test, the client MAY accept an unauthenticated Reply message from the server.

#### 22.4.4.6. Receiving Reconfigure Messages

The client MUST discard the Reconfigure if the message fails to pass the validation test and MAY log the validation failure.

### 22.4.5. Server Considerations for Delayed Authentication Protocol

After receiving a Solicit message that contains an Authentication option, the server selects a key for the client, based on the client's DUID and key selection policies with which the server has been configured. The server identifies the selected key in the Advertise message and uses the key to validate subsequent messages between the client and the server.

#### 22.4.5.1. Receiving Solicit Messages and Sending Advertise Messages

The server selects a key for the client and includes authentication information in the Advertise message returned to the client as specified in Section 22.4. The server MUST record the identifier of the key selected for the client and use that same key for validating subsequent messages with the client.

#### 22.4.5.2. Receiving Request, Confirm, Renew, Rebind or Release Messages and Sending Reply Messages

The server uses the key identified in the message and validates the message as specified in Section 22.4.2. If the message fails to pass the validation test or the server does not know the key identified by the 'key ID' field, the server MUST discard the message and MAY choose to log the validation failure. If the server receives a client message without an authentication option while the server has previously sent authentication information in the same session, it MUST discard the message and MAY choose to log the validation failure, because the client violates the definition in Section 22.4.4.3.

If the message passes the validation test, the server responds to the specific message as described in Section 19.2. The server MUST include authentication information generated using the key identified in the received message, as specified in Section 22.4.

#### 22.5. Reconfigure Key Authentication Protocol

The Reconfigure key authentication protocol provides protection against misconfiguration of a client caused by a Reconfigure message sent by a malicious DHCP server. In this protocol, a DHCP server sends a Reconfigure Key to the client in the initial exchange of DHCP messages. The client records the Reconfigure Key for use in authenticating subsequent Reconfigure messages from that server. The server then includes an HMAC computed from the Reconfigure Key in subsequent Reconfigure messages.

Both the Reconfigure Key sent from the server to the client and the HMAC in subsequent Reconfigure messages are carried as the Authentication information in an Authentication option. The format of the Authentication information is defined in the following section.

The Reconfigure Key protocol is used (initiated by the server) only if the client and server are not using any other authentication protocol and the client and server have negotiated to use Reconfigure messages.

##### 22.5.1. Use of the Authentication Option in the Reconfigure Key Authentication Protocol

The following fields are set in an Authentication option for the Reconfigure Key Authentication Protocol:

protocol    3



Authentication option; the HMAC-MD5 field in the Authentication option is set to zero for the HMAC-MD5 computation. The server includes the HMAC-MD5 in the authentication information field in an Authentication option included in the Reconfigure message sent to the client.

#### 22.5.3. Client considerations for Reconfigure Key protocol

The client will receive a Reconfigure Key from the server in the initial Reply message from the server. The client records the Reconfigure Key for use in authenticating subsequent Reconfigure messages.

To authenticate a Reconfigure message, the client computes an HMAC-MD5 over the DHCP Reconfigure message, using the Reconfigure Key received from the server. If this computed HMAC-MD5 matches the value in the Authentication option, the client accepts the Reconfigure message.

### 23. DHCP Options

Options are used to carry additional information and parameters in DHCP messages. Every option shares a common base format, as described in Section 23.1. All values in options are represented in network byte order.

This document describes the DHCP options defined as part of the base DHCP specification. Other options may be defined in the future in separate documents. See [RFC7227] for guidelines regarding new options definition.

Unless otherwise noted, each option may appear only in the options area of a DHCP message and may appear only once. If an option does appear multiple times, each instance is considered separate and the data areas of the options MUST NOT be concatenated or otherwise combined.

Options that are allowed to appear only once are called singleton options. The only non-singleton options defined in this document are IA\_NA (see Section 23.4), IA\_TA (see Section 23.5), and IA\_PD (see Section 23.21) options. Also, IAAddress (see Section 23.6) and IAPrefix (see Section 23.22) may appear in their respective IA options more than once.

### 23.1. Format of DHCP Options

The format of DHCP options is:

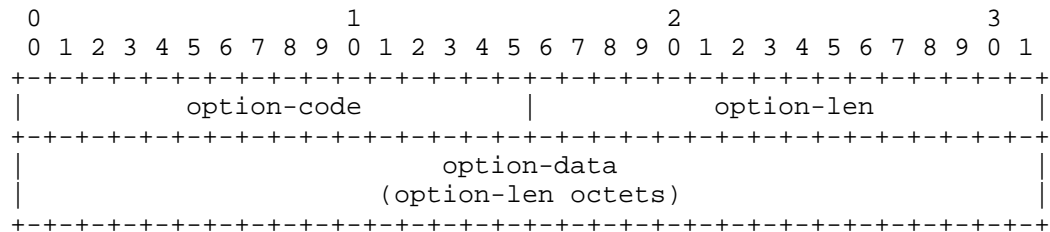


Figure 11: Option Format

option-code	An unsigned integer identifying the specific option type carried in this option.
option-len	An unsigned integer giving the length of the option-data field in this option in octets.
option-data	The data for the option; the format of this data depends on the definition of the option.

DHCPv6 options are scoped by using encapsulation. Some options apply generally to the client, some are specific to an IA, and some are specific to the addresses within an IA. These latter two cases are discussed in Section 23.4 and Section 23.6.

### 23.2. Client Identifier Option

The Client Identifier option is used to carry a DUID (see Section 10) identifying a client between a client and a server. The format of the Client Identifier option is:

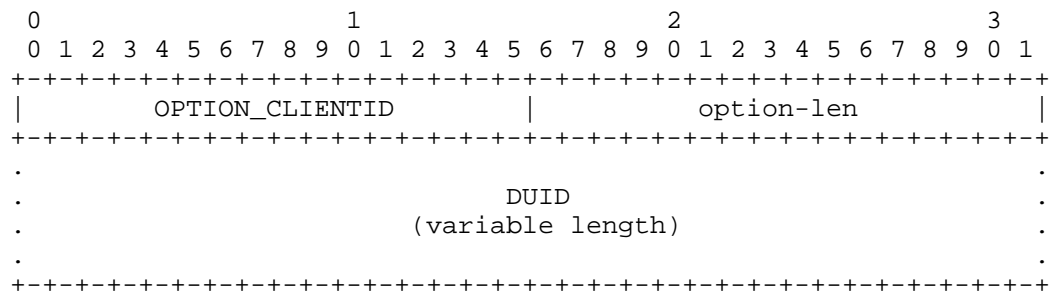


Figure 12: Client Identifier Option Format

option-code            OPTION\_CLIENTID (1).

option-len            Length of DUID in octets.

DUID                  The DUID for the client.

### 23.3. Server Identifier Option

The Server Identifier option is used to carry a DUID (see Section 10) identifying a server between a client and a server. The format of the Server Identifier option is:

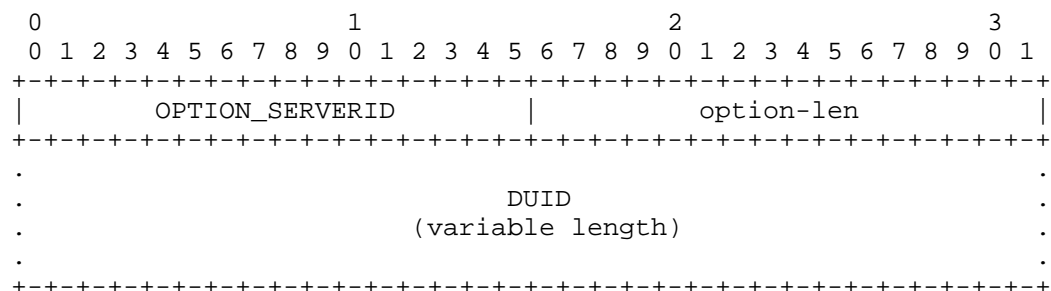


Figure 13: Server Identifier Option Format

option-code            OPTION\_SERVERID (2).

option-len            Length of DUID in octets.

DUID                  The DUID for the server.

#### 23.4. Identity Association for Non-temporary Addresses Option

The Identity Association for Non-temporary Addresses option (IA\_NA option) is used to carry an IA\_NA, the parameters associated with the IA\_NA, and the non-temporary addresses associated with the IA\_NA.

Addresses appearing in an IA\_NA option are not temporary addresses (see Section 23.5).

The format of the IA\_NA option is:

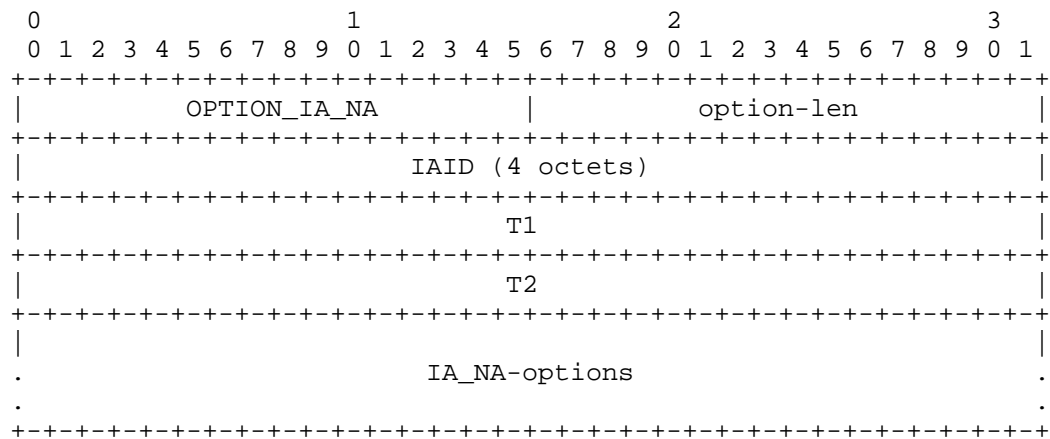


Figure 14: Identity Association for Non-temporary Addresses Option Format

option-code	OPTION_IA_NA (3).
option-len	12 + length of IA_NA-options field.
IAID	The unique identifier for this IA_NA; the IAID must be unique among the identifiers for all of this client's IA_NAs. The number space for IA_NA IAIDs is separate from the number space for IA_TA IAIDs.
T1	The time at which the client contacts the server from which the addresses in the IA_NA were obtained to extend the lifetimes of the addresses assigned to the IA_NA; T1 is a time duration relative to the current time expressed in units of seconds.



T2                   The time at which the client contacts any available server to extend the lifetimes of the addresses assigned to the IA\_NA; T2 is a time duration relative to the current time expressed in units of seconds.

IA\_NA-options       Options associated with this IA\_NA.

The IA\_NA-options field encapsulates those options that are specific to this IA\_NA. For example, all of the IA Address Options carrying the addresses associated with this IA\_NA are in the IA\_NA-options field.

Each IA\_NA carries one "set" of non-temporary addresses; that is, at most one address from each prefix assigned to the link to which the client is attached.

An IA\_NA option may only appear in the options area of a DHCP message. A DHCP message may contain multiple IA\_NA options.

The status of any operations involving this IA\_NA is indicated in a Status Code option in the IA\_NA-options field.

Note that an IA\_NA has no explicit "lifetime" or "lease length" of its own. When the valid lifetimes of all of the addresses in an IA\_NA have expired, the IA\_NA can be considered as having expired. T1 and T2 are included to give servers explicit control over when a client recontacts the server about a specific IA\_NA.

In a message sent by a client to a server, values in the T1 and T2 fields indicate the client's preference for those parameters. The client sets T1 and T2 to 0 if it has no preference for those values. In a message sent by a server to a client, the client MUST use the values in the T1 and T2 fields for the T1 and T2 parameters, unless those values in those fields are 0. The values in the T1 and T2 fields are the number of seconds until T1 and T2.

The server selects the T1 and T2 times to allow the client to extend the lifetimes of any addresses in the IA\_NA before the lifetimes expire, even if the server is unavailable for some short period of time. Recommended values for T1 and T2 are .5 and .8 times the shortest preferred lifetime of the addresses in the IA that the server is willing to extend, respectively. If the "shortest" preferred lifetime is 0xffffffff ("infinity"), the recommended T1 and T2 values are also 0xffffffff. If the time at which the addresses in an IA\_NA are to be renewed is to be left to the discretion of the client, the server sets T1 and T2 to 0.

If a server receives an IA\_NA with T1 greater than T2, and both T1 and T2 are greater than 0, the server ignores the invalid values of T1 and T2 and processes the IA\_NA as though the client had set T1 and T2 to 0.

If a client receives an IA\_NA with T1 greater than T2, and both T1 and T2 are greater than 0, the client discards the IA\_NA option and processes the remainder of the message as though the server had not included the invalid IA\_NA option.

Care should be taken in setting T1 or T2 to 0xffffffff ("infinity"). A client will never attempt to extend the lifetimes of any addresses in an IA with T1 set to 0xffffffff. A client will never attempt to use a Rebind message to locate a different server to extend the lifetimes of any addresses in an IA with T2 set to 0xffffffff.

This option MAY appear in a Confirm message if the lifetimes on the non-temporary addresses in the associated IA have not expired.

### 23.5. Identity Association for Temporary Addresses Option

The Identity Association for the Temporary Addresses (IA\_TA) option is used to carry an IA\_TA, the parameters associated with the IA\_TA and the addresses associated with the IA\_TA. All of the addresses in this option are used by the client as temporary addresses, as defined in [RFC4941]. The format of the IA\_TA option is:

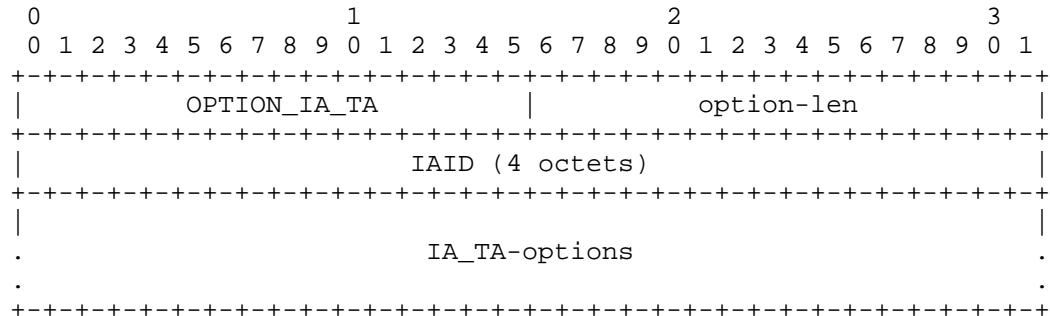


Figure 15: Identity Association for Temporary Addresses Option Format

option-code	OPTION_IA_TA (4).
option-len	4 + length of IA_TA-options field.
IAID	The unique identifier for this IA_TA; the IAID must be unique among the identifiers for

all of this client's IA\_TAs. The number space for IA\_TA IAIDs is separate from the number space for IA\_NA IAIDs.

IA\_TA-options                      Options associated with this IA\_TA.

The IA\_TA-Options field encapsulates those options that are specific to this IA\_TA. For example, all of the IA Address Options carrying the addresses associated with this IA\_TA are in the IA\_TA-options field.

Each IA\_TA carries one "set" of temporary addresses.

An IA\_TA option may only appear in the options area of a DHCP message. A DHCP message may contain multiple IA\_TA options.

The status of any operations involving this IA\_TA is indicated in a Status Code option in the IA\_TA-options field.

Note that an IA has no explicit "lifetime" or "lease length" of its own. When the valid lifetimes of all of the addresses in an IA\_TA have expired, the IA can be considered as having expired.

An IA\_TA option does not include values for T1 and T2. A client MAY request that the lifetimes on temporary addresses be extended by including the addresses in a IA\_TA option sent in a Renew or Rebind message to a server. For example, a client would request an extension on the lifetime of a temporary address to allow an application to continue to use an established TCP connection.

The client obtains new temporary addresses by sending an IA\_TA option with a new IAID to a server. Requesting new temporary addresses from the server is the equivalent of generating new temporary addresses as described in [RFC4941]. The server will generate new temporary addresses and return them to the client. The client should request new temporary addresses before the lifetimes on the previously assigned addresses expire.

A server MUST return the same set of temporary address for the same IA\_TA (as identified by the IAID) as long as those addresses are still valid. After the lifetimes of the addresses in an IA\_TA have expired, the IAID may be reused to identify a new IA\_TA with new temporary addresses.

This option MAY appear in a Confirm message if the lifetimes on the temporary addresses in the associated IA have not expired.

## 23.6. IA Address Option

The IA Address option is used to specify IPv6 addresses associated with an IA\_NA or an IA\_TA. The IA Address option must be encapsulated in the Options field of an IA\_NA or IA\_TA option. The Options fields of the IA\_NA or IA\_TA option encapsulates those options that are specific to this address.

The format of the IA Address option is:

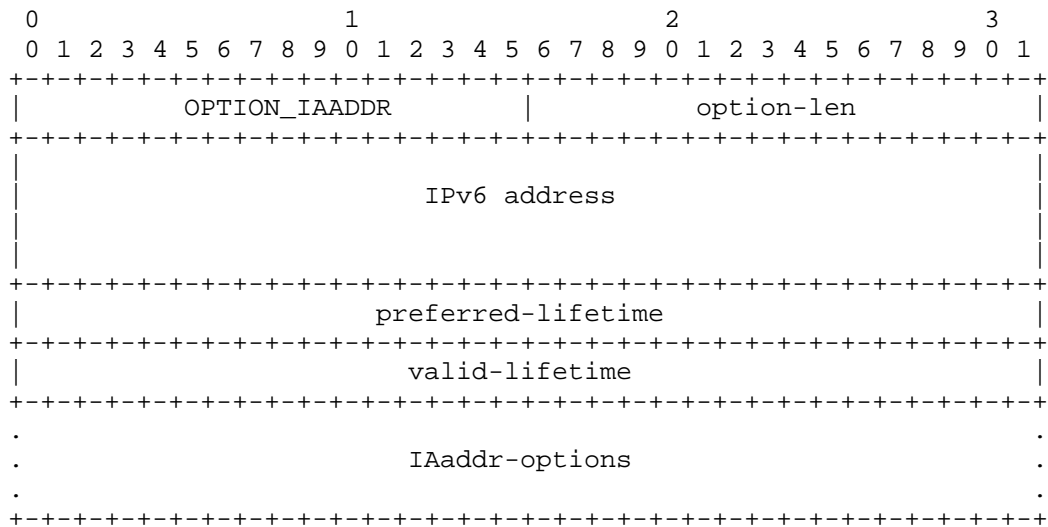


Figure 16: IA Address Option Format

option-code	OPTION_IAADDR (5).
option-len	24 + length of IAaddr-options field.
IPv6 address	An IPv6 address.
preferred-lifetime	The preferred lifetime for the IPv6 address in the option, expressed in units of seconds.
valid-lifetime	The valid lifetime for the IPv6 address in the option, expressed in units of seconds.
IAaddr-options	Options associated with this address.

In a message sent by a client to a server, values in the preferred and valid lifetime fields indicate the client's preference for those

parameters. The client may send 0 if it has no preference for the preferred and valid lifetimes. If a client wishes to express its lifetimes preferences and does not have the knowledge to populate the IPv6 address field, it can use unspecified address (::). It is up to a server to honor or ignore these preferences.

In a message sent by a server to a client, the client **MUST** use the values in the preferred and valid lifetime fields for the preferred and valid lifetimes. The values in the preferred and valid lifetimes are the number of seconds remaining in each lifetime.

A client discards any addresses for which the preferred lifetime is greater than the valid lifetime. A server ignores the lifetimes set by the client if the preferred lifetime is greater than the valid lifetime and ignores the values for T1 and T2 set by the client if those values are greater than the preferred lifetime.

Care should be taken in setting the valid lifetime of an address to 0xffffffff ("infinity"), which amounts to a permanent assignment of an address to a client.

More than one IA Address Option can appear in an IA\_NA option or an IA\_TA option.

The status of any operations involving this IA Address is indicated in a Status Code option in the IAAddr-options field, as specified in Section 23.13.

### 23.7. Option Request Option

The Option Request option is used to identify a list of options in a message between a client and a server. The format of the Option Request option is:

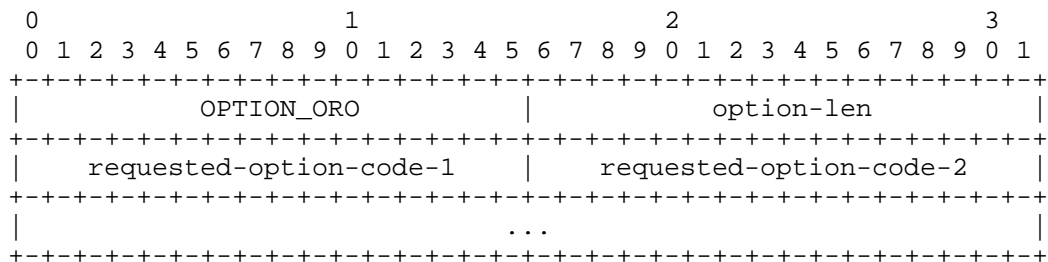


Figure 17: Option Request Option Format

option-code                      OPTION\_ORO (6).

option-len                    2 \* number of requested options.

requested-option-code-n    The option code for an option requested  
by the client.

A client MAY include an Option Request option in a Solicit, Request, Renew, Rebind, Confirm or Information-request message to inform the server about options the client wants the server to send to the client. A server MAY include an Option Request option in a Reconfigure message to indicate which options the client should request from the server. If there is a need to request encapsulated options, top-level Option Request option MUST be used for that purpose. There is no need request IAADDR or IAPREFIX.

### 23.8. Preference Option

The Preference option is sent by a server to a client to affect the selection of a server by the client.

The format of the Preference option is:

```

      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_PREFERENCE          |          option-len          |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|  pref-value  |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Figure 18: Preference Option Format

option-code                    OPTION\_PREFERENCE (7).

option-len                    1.

pref-value                    The preference value for the server in this  
message.

A server MAY include a Preference option in an Advertise message to control the selection of a server by the client. See Section 18.1.3 for the use of the Preference option by the client and the interpretation of Preference option data value.

## 23.9. Elapsed Time Option

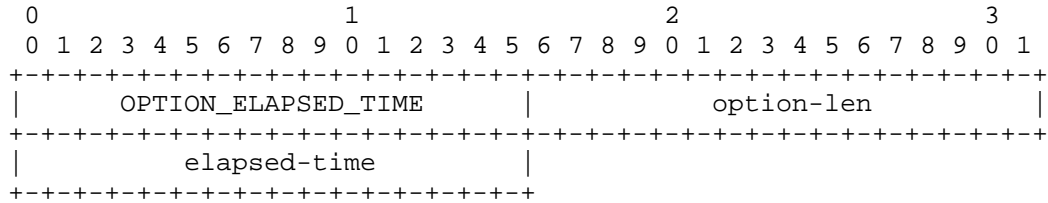


Figure 19: Elapsed Time Option Format

option-code	OPTION_ELAPSED_TIME (8).
option-len	2.
elapsed-time	The amount of time since the client began its current DHCP transaction. This time is expressed in hundredths of a second ( $10^{-2}$ seconds).

A client MUST include an Elapsed Time option in messages to indicate how long the client has been trying to complete a DHCP message exchange. The elapsed time is measured from the time at which the client sent the first message in the message exchange, and the elapsed-time field is set to 0 in the first message in the message exchange. Servers and Relay Agents use the data value in this option as input to policy controlling how a server responds to a client message. For example, the elapsed time option allows a secondary DHCP server to respond to a request when a primary server has not answered in a reasonable time. The elapsed time value is an unsigned, 16 bit integer. The client uses the value 0xffff to represent any elapsed time values greater than the largest time value that can be represented in the Elapsed Time option.

## 23.10. Relay Message Option

The Relay Message option carries a DHCP message in a Relay-forward or Relay-reply message.

The format of the Relay Message option is:

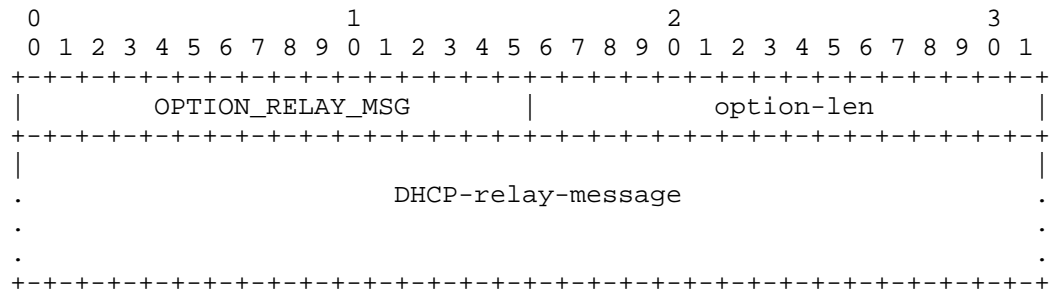


Figure 20: Relay Message Option Format

option-code	OPTION_RELAY_MSG (9)
option-len	Length of DHCP-relay-message
DHCP-relay-message	In a Relay-forward message, the received message, relayed verbatim to the next relay agent or server; in a Relay-reply message, the message to be copied and relayed to the relay agent or client whose address is in the peer-address field of the Relay-reply message

### 23.11. Authentication Option

The Authentication option carries authentication information to authenticate the identity and contents of DHCP messages. The use of the Authentication option is described in Section 22. The format of the Authentication option is:



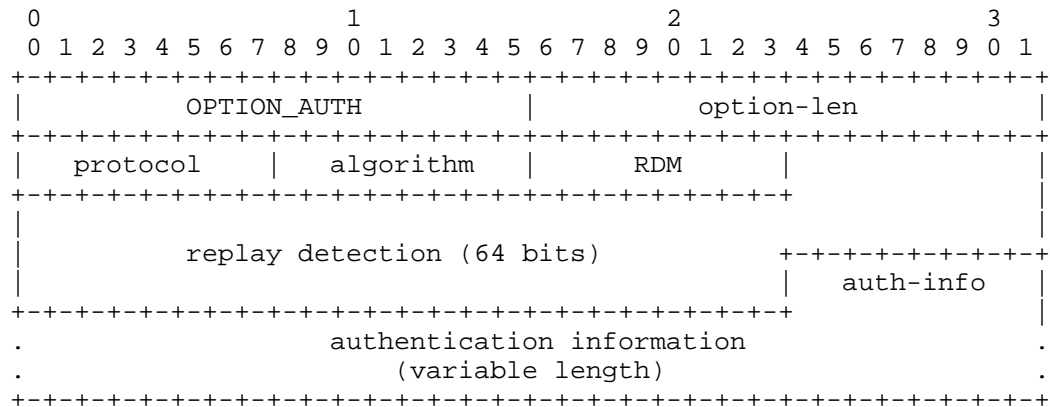


Figure 21: Authentication Option Format

option-code	OPTION_AUTH (11).
option-len	11 + length of authentication information field.
protocol	The authentication protocol used in this authentication option.
algorithm	The algorithm used in the authentication protocol.
RDM	The replay detection method used in this authentication option.
Replay detection	The replay detection information for the RDM.
authentication information	The authentication information, as specified by the protocol and algorithm used in this authentication option.

### 23.12. Server Unicast Option

The server sends this option to a client to indicate to the client that it is allowed to unicast messages to the server. The format of the Server Unicast option is:

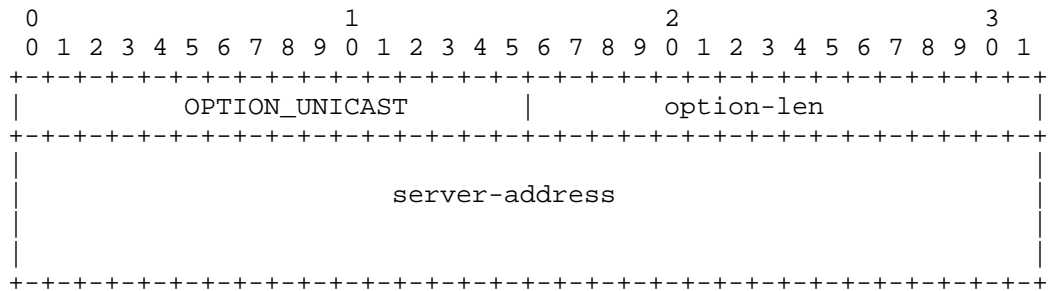


Figure 22: Server Unicast Option Format

option-code	OPTION_UNICAST (12).
option-len	16.
server-address	The IP address to which the client should send messages delivered using unicast.

The server specifies the IPv6 address to which the client is to send unicast messages in the server-address field. When a client receives this option, where permissible and appropriate, the client sends messages directly to the server using the IPv6 address specified in the server-address field of the option.

When the server sends a Unicast option to the client, some messages from the client will not be relayed by Relay Agents, and will not include Relay Agent options from the Relay Agents. Therefore, a server should only send a Unicast option to a client when Relay Agents are not sending Relay Agent options. A DHCP server rejects any messages sent inappropriately using unicast to ensure that messages are relayed by Relay Agents when Relay Agent options are in use.

Details about when the client may send messages to the server using unicast are in Section 19.

### 23.13. Status Code Option

This option returns a status indication related to the DHCP message or option in which it appears. The format of the Status Code option is:

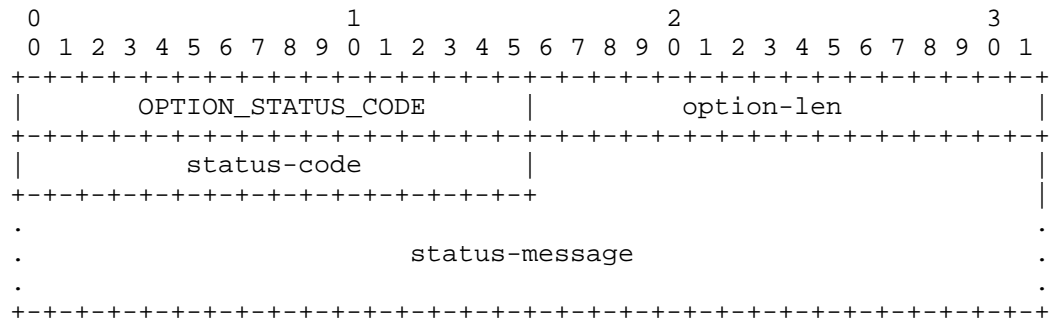


Figure 23: Status Code Option Format

option-code	OPTION_STATUS_CODE (13).
option-len	2 + length of status-message.
status-code	The numeric code for the status encoded in this option.
status-message	A UTF-8 encoded text string suitable for display to an end user, which MUST NOT be null-terminated.

A Status Code option may appear in the options field of a DHCP message and/or in the options field of another option. If the Status Code option does not appear in a message in which the option could appear, the status of the message is assumed to be Success.

The status-codes values previously defined by [RFC3315] and [RFC3633] are:

Name	Code	Description
Success	0	Success.
UnspecFail	1	Failure, reason unspecified; this status code is sent by either a client or a server to indicate a failure not explicitly specified in this document.
NoAddrsAvail	2	Server has no addresses available to assign to the IA(s).
NoBinding	3	Client record (binding) unavailable.
NotOnLink	4	The prefix for the address is not appropriate for the link to which the client is attached.
UseMulticast	5	Sent by a server to a client to force the client to send messages to the server using the All_DHCP_Relay_Agents_and_Servers address.
NoPrefixAvail	6	Delegating router has no prefixes available to assign to the IAPD(s).

#### 23.14. Rapid Commit Option

The Rapid Commit option is used to signal the use of the two message exchange for address assignment. The format of the Rapid Commit option is:

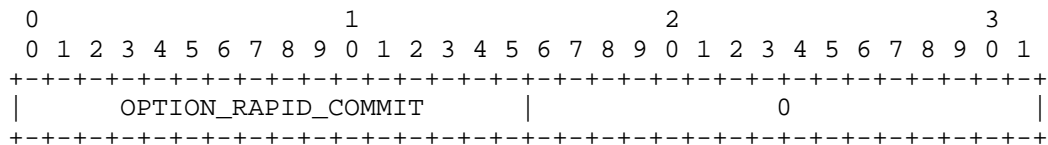


Figure 24: Rapid Commit Option Format

option-code           OPTION\_RAPID\_COMMIT (14).

option-len            0.

A client MAY include this option in a Solicit message if the client is prepared to perform the Solicit-Reply message exchange described in Section 18.1.1.

A server MUST include this option in a Reply message sent in response to a Solicit message when completing the Solicit-Reply message exchange.

## DISCUSSION:

Each server that responds with a Reply to a Solicit that includes a Rapid Commit option will commit the assigned addresses in the Reply message to the client, and will not receive any confirmation that the client has received the Reply message. Therefore, if more than one server responds to a Solicit that includes a Rapid Commit option, some servers will commit addresses that are not actually used by the client.

The problem of unused addresses can be minimized, for example, by designing the DHCP service so that only one server responds to the Solicit or by using relatively short lifetimes for assigned addresses, or the DHCP client initiatively releases unused addresses using the Release message.

## 23.15. User Class Option

The User Class option is used by a client to identify the type or category of user or applications it represents.

The format of the User Class option is:

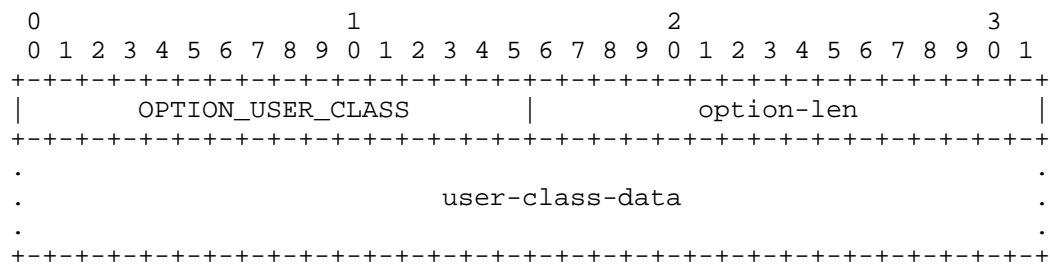


Figure 25: User Class Option Format

option-code            OPTION\_USER\_CLASS (15).  
option-len            Length of user class data field.  
user-class-data        The user classes carried by the client.

The information contained in the data area of this option is contained in one or more opaque fields that represent the user class or classes of which the client is a member. A server selects configuration information for the client based on the classes identified in this option. For example, the User Class option can be used to configure all clients of people in the accounting department

with a different printer than clients of people in the marketing department. The user class information carried in this option MUST be configurable on the client.

The data area of the user class option MUST contain one or more instances of user class data. Each instance of the user class data is formatted as follows:

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+-----+
|          user-class-len          |          opaque-data          |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+...+-----+-----+

```

Figure 26: User Class Data Format

The user-class-len is two octets long and specifies the length of the opaque user class data in network byte order.

A server interprets the classes identified in this option according to its configuration to select the appropriate configuration information for the client. A server may use only those user classes that it is configured to interpret in selecting configuration information for a client and ignore any other user classes. In response to a message containing a User Class option, a server includes a User Class option containing those classes that were successfully interpreted by the server, so that the client can be informed of the classes interpreted by the server.

#### 23.16. Vendor Class Option

This option is used by a client to identify the vendor that manufactured the hardware on which the client is running. The information contained in the data area of this option is contained in one or more opaque fields that identify details of the hardware configuration. The format of the Vendor Class option is:

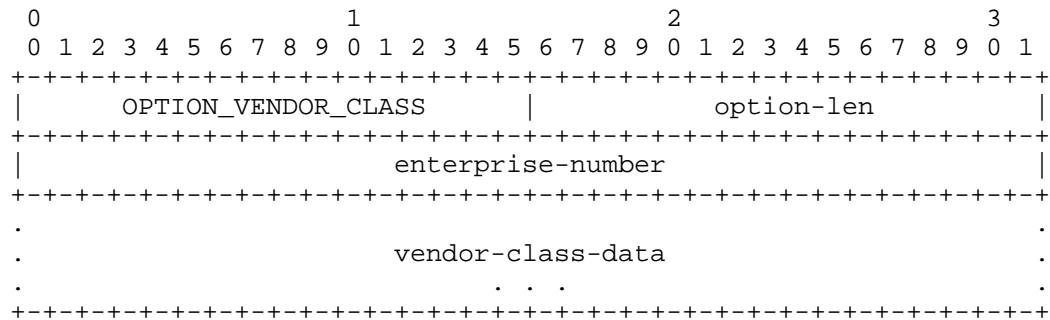


Figure 27: Vendor Class Option Format

option-code	OPTION_VENDOR_CLASS (16).
option-len	4 + length of vendor class data field.
enterprise-number	The vendor's registered Enterprise Number as registered with IANA [IANA-PEN].
vendor-class-data	The hardware configuration of the host on which the client is running.

The vendor-class-data is composed of a series of separate items, each of which describes some characteristic of the client's hardware configuration. Examples of vendor-class-data instances might include the version of the operating system the client is running or the amount of memory installed on the client.

Each instance of the vendor-class-data is formatted as follows:

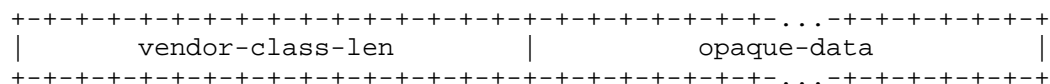


Figure 28: Vendor Class Data Format

The vendor-class-len is two octets long and specifies the length of the opaque vendor class data in network byte order.

Servers and clients MUST NOT include more than one instance of OPTION\_VENDOR\_CLASS with the same Enterprise Number. Each instance of OPTION\_VENDOR\_CLASS can carry multiple sub-options.

## 23.17. Vendor-specific Information Option

This option is used by clients and servers to exchange vendor-specific information.

The format of the Vendor-specific Information option is:

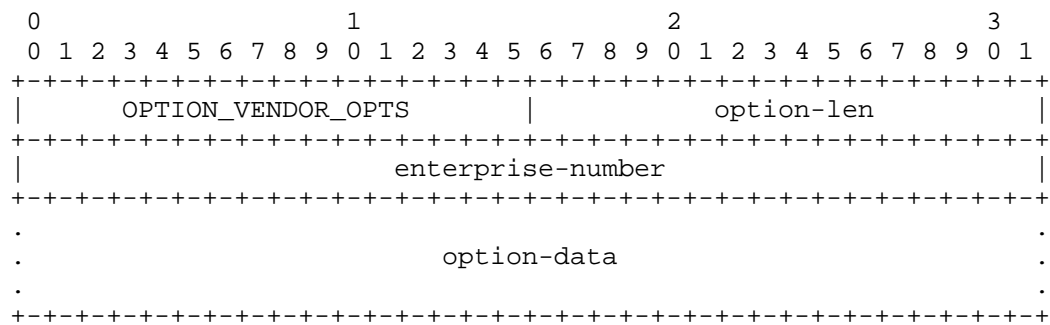


Figure 29: Vendor-specific Information Option Format

option-code	OPTION_VENDOR_OPTS (17).
option-len	4 + length of option-data field.
enterprise-number	The vendor's registered Enterprise Number as registered with IANA [IANA-PEN].
option-data	An opaque object, interpreted by vendor-specific code on the clients and servers.

The definition of the information carried in this option is vendor specific. The vendor is indicated in the enterprise-number field. Use of vendor-specific information allows enhanced operation, utilizing additional features in a vendor's DHCP implementation. A DHCP client that does not receive requested vendor-specific information will still configure the host device's IPv6 stack to be functional.

The encapsulated vendor-specific options field MUST be encoded as a sequence of code/length/value fields of identical format to the DHCP options field. The option codes are defined by the vendor identified in the enterprise-number field and are not managed by IANA. Each of the encapsulated options is formatted as follows:



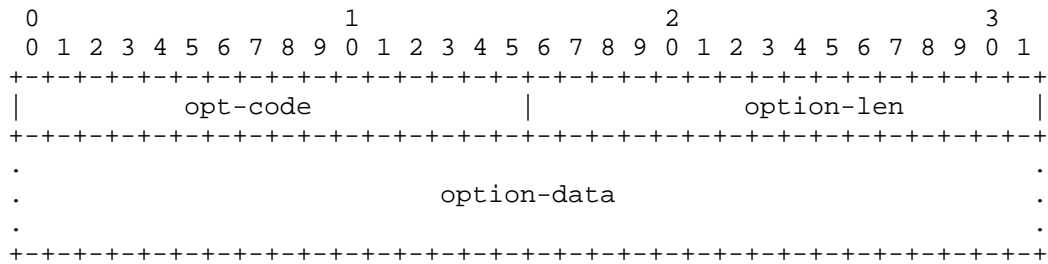


Figure 30: Vendor-specific Options Format

opt-code	The code for the encapsulated option.
option-len	An unsigned integer giving the length of the option-data field in this encapsulated option in octets.
option-data	The data area for the encapsulated option.

Multiple instances of the Vendor-specific Information option may appear in a DHCP message. Each instance of the option is interpreted according to the option codes defined by the vendor identified by the Enterprise Number in that option. Servers and clients MUST NOT send more than one instance of Vendor-specific Information option with the same Enterprise Number. Each instance of Vendor-specific Information option MAY contain multiple encapsulated options.

A client that is interested in receiving a Vendor-specific Information Option:

- MUST specify the Vendor-specific Information Option in an Option Request Option.
- MAY specify an associated Vendor Class Option.
- MAY specify the Vendor-specific Information Option with any data.

Servers only return the Vendor-specific Information Options if specified in Option Request Options from clients and:

- MAY use the Enterprise Numbers in the associated Vendor Class Options to restrict the set of Enterprise Numbers in the Vendor-specific Information Options returned.
- MAY return all configured Vendor-specific Information Options.

- MAY use other information in the packet or in its configuration to determine which set of Enterprise Numbers in the Vendor-specific Information Options to return.

### 23.18. Interface-Id Option

The relay agent MAY send the Interface-id option to identify the interface on which the client message was received. If a relay agent receives a Relay-reply message with an Interface-id option, the relay agent relays the message to the client through the interface identified by the option.

The format of the Interface ID option is:

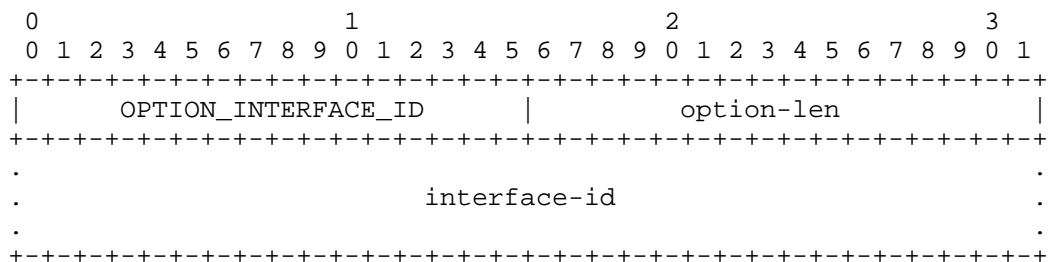


Figure 31: Interface-ID Option Format

option-code	OPTION_INTERFACE_ID (18).
option-len	Length of interface-id field.
interface-id	An opaque value of arbitrary length generated by the relay agent to identify one of the relay agent's interfaces.

The server MUST copy the Interface-Id option from the Relay-forward message into the Relay-reply message the server sends to the relay agent in response to the Relay-forward message. This option MUST NOT appear in any message except a Relay-forward or Relay-reply message.

Servers MAY use the Interface-ID for parameter assignment policies. The Interface-ID SHOULD be considered an opaque value, with policies based on exact match only; that is, the Interface-ID SHOULD NOT be internally parsed by the server. The Interface-ID value for an interface SHOULD be stable and remain unchanged, for example, after the relay agent is restarted; if the Interface-ID changes, a server will not be able to use it reliably in parameter assignment policies.

## 23.19. Reconfigure Message Option

A server includes a Reconfigure Message option in a Reconfigure message to indicate to the client whether the client responds with a Renew message, a Rebind message, or an Information-request message. The format of this option is:

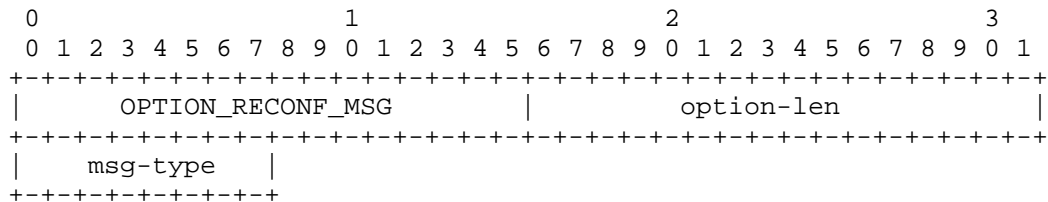


Figure 32: Reconfigure Message Option Format

option-code	OPTION_RECONF_MSG (19).
option-len	1.
msg-type	5 for Renew message, 6 for Rebind, 11 for Information-request message.

The Reconfigure Message option can only appear in a Reconfigure message.

## 23.20. Reconfigure Accept Option

A client uses the Reconfigure Accept option to announce to the server whether the client is willing to accept Reconfigure messages, and a server uses this option to tell the client whether or not to accept Reconfigure messages. The default behavior, in the absence of this option, means unwillingness to accept Reconfigure messages, or instruction not to accept Reconfigure messages, for the client and server messages, respectively. The following figure gives the format of the Reconfigure Accept option:

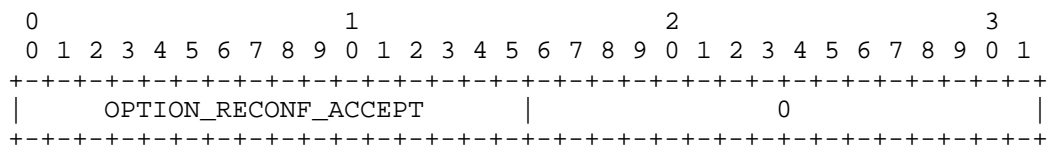


Figure 33: Reconfigure Accept Option Format

option-code           OPTION\_RECONF\_ACCEPT (20).

option-len            0.

### 23.21. Identity Association for Prefix Delegation Option

The IA\_PD option is used to carry a prefix delegation identity association, the parameters associated with the IA\_PD and the prefixes associated with it.

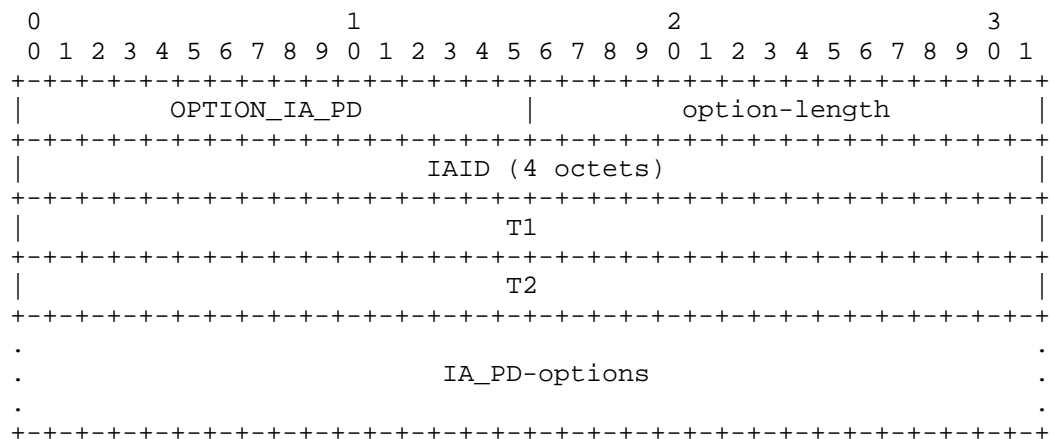


Figure 34: Identity Association for Prefix Delegation Option Format

option-code           OPTION\_IA\_PD (25).

option-length        12 + length of IA\_PD-options field.

IAID                The unique identifier for this IA\_PD; the IAID must be unique among the identifiers for all of this requesting router's IA\_PD's.

T1                 The time at which the requesting router should contact the delegating router from which the prefixes in the IA\_PD were obtained to extend the lifetimes of the prefixes delegated to the IA\_PD; T1 is a time duration relative to the current time expressed in units of seconds.

T2                 The time at which the requesting router should contact any available delegating router to extend the lifetimes of the

prefixes assigned to the IA\_PD; T2 is a time duration relative to the current time expressed in units of seconds.

IA\_PD-options            Options associated with this IA\_PD.

The IA\_PD-options field encapsulates those options that are specific to this IA\_PD. For example, all of the IA\_PD Prefix Options carrying the prefixes associated with this IA\_PD are in the IA\_PD-options field.

An IA\_PD option may only appear in the options area of a DHCP message. A DHCP message may contain multiple IA\_PD options.

The status of any operations involving this IA\_PD is indicated in a Status Code option in the IA\_PD-options field.

Note that an IA\_PD has no explicit "lifetime" or "lease length" of its own. When the valid lifetimes of all of the prefixes in a IA\_PD have expired, the IA\_PD can be considered as having expired. T1 and T2 are included to give delegating routers explicit control over when a requesting router should contact the delegating router about a specific IA\_PD.

In a message sent by a requesting router to a delegating router, values in the T1 and T2 fields indicate the requesting router's preference for those parameters. The requesting router sets T1 and T2 to zero if it has no preference for those values. In a message sent by a delegating router to a requesting router, the requesting router MUST use the values in the T1 and T2 fields for the T1 and T2 parameters. The values in the T1 and T2 fields are the number of seconds until T1 and T2.

The delegating router selects the T1 and T2 times to allow the requesting router to extend the lifetimes of any prefixes in the IA\_PD before the lifetimes expire, even if the delegating router is unavailable for some short period of time. Recommended values for T1 and T2 are .5 and .8 times the shortest preferred lifetime of the prefixes in the IA\_PD that the delegating router is willing to extend, respectively. If the time at which the prefixes in an IA\_PD are to be renewed is to be left to the discretion of the requesting router, the delegating router sets T1 and T2 to 0.

If a delegating router receives an IA\_PD with T1 greater than T2, and both T1 and T2 are greater than 0, the delegating router ignores the invalid values of T1 and T2 and processes the IA\_PD as though the requesting router had set T1 and T2 to 0.

## 23.22. IA Prefix Option

[illegible]

Figure 35: IA Prefix Option Format

prefix-length	Length for this prefix in bits.
IPv6-prefix	An IPv6 prefix.
IAprefix-options	Options associated with this prefix.

In a message sent by a requesting router to a delegating router, the values in the fields can be used to indicate the requesting router's preference for those values. The requesting router may send a value of zero to indicate no preference. A requesting router may set the IPv6 prefix field to zero and a given value in the prefix-length field to indicate a preference for the size of the prefix to be delegated.

In a message sent by a delegating router the preferred and valid lifetimes should be set to the values of AdvPreferredLifetime and AdvValidLifetime as specified in section 6.2.1, "Router Configuration Variables" of [RFC2461], unless administratively configured.

A requesting router discards any prefixes for which the preferred lifetime is greater than the valid lifetime. A delegating router ignores the lifetimes set by the requesting router if the preferred lifetime is greater than the valid lifetime and ignores the values for T1 and T2 set by the requesting router if those values are greater than the preferred lifetime.

The values in the preferred and valid lifetimes are the number of seconds remaining for each lifetime.

An IA\_PD Prefix option may appear only in an IA\_PD option. More than one IA\_PD Prefix Option can appear in a single IA\_PD option.

The status of any operations involving this IA\_PD Prefix option is indicated in a Status Code option in the IAprefix-options field.

### 23.23. SOL\_MAX\_RT Option

A DHCP server sends the SOL\_MAX\_RT option to a client to override the default value of SOL\_MAX\_RT. The value of SOL\_MAX\_RT in the option replaces the default value defined in Section 6.5. One use for the SOL\_MAX\_RT option is to set a longer value for SOL\_MAX\_RT, which reduces the Solicit traffic from a client that has not received a response to its Solicit messages.

The format of the SOL\_MAX\_RT option is:

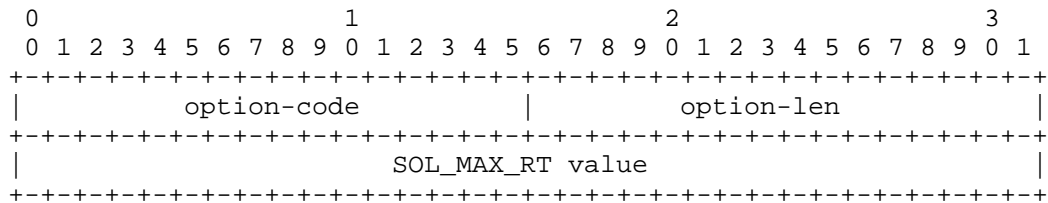


Figure 36: SOL\_MAX\_RT Option Format

option-code	OPTION_SOL_MAX_RT (82).
option-len	4.
SOL_MAX_RT value	Overriding value for SOL_MAX_RT in seconds; MUST be in range: 60 <= "value" <= 86400 (1 day).

A DHCP client MUST include the SOL\_MAX\_RT option code in any Option Request option (see Section 23.7) it sends.

The DHCP server MAY include the SOL\_MAX\_RT option in any response it sends to a client that has included the SOL\_MAX\_RT option code in an Option Request option. The SOL\_MAX\_RT option is sent in the main body of the message to client, not as an encapsulated option in, e.g., an IA\_NA, IA\_TA, or IA\_PD option.

A DHCP client MUST ignore any SOL\_MAX\_RT option values that are less than 60 or more than 86400.

If a DHCP client receives a message containing a SOL\_MAX\_RT option that has a valid value for SOL\_MAX\_RT, the client MUST set its internal SOL\_MAX\_RT parameter to the value contained in the SOL\_MAX\_RT option. This value of SOL\_MAX\_RT is then used by the retransmission mechanism defined in Section 15 and Section 18.1.2.

Updated SOL\_MAX\_RT value applies only to the network interface on which the client received SOL\_MAX\_RT option.

#### 23.24. INF\_MAX\_RT Option

A DHCP server sends the INF\_MAX\_RT option to a client to override the default value of INF\_MAX\_RT. The value of INF\_MAX\_RT in the option replaces the default value defined in Section 6.5. One use for the INF\_MAX\_RT option is to set a longer value for INF\_MAX\_RT, which reduces the Information-request traffic from a client that has not received a response to its Information-request messages.



The format of the INF\_MAX\_RT option is:

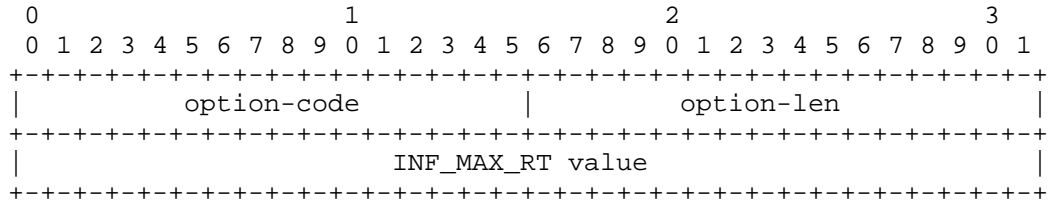


Figure 37: INF\_MAX\_RT Option Format

option-code	OPTION_INF_MAX_RT (83).
option-len	4.
SOL_MAX_RT value	Overriding value for INF_MAX_RT in seconds; MUST be in range: 60 <= "value" <= 86400 (1 day).

A DHCP client MUST include the INF\_MAX\_RT option code in any Option Request option (see Section 23.7) it sends.

The DHCP server MAY include the INF\_MAX\_RT option in any response it sends to a client that has included the INF\_MAX\_RT option code in an Option Request option. The INF\_MAX\_RT option is sent in the main body of the message to client, not as an encapsulated option in, e.g., an IA\_NA, IA\_TA, or IA\_PD option.

A DHCP client MUST ignore any INF\_MAX\_RT option values that are less than 60 or more than 86400.

If a DHCP client receives a message containing an INF\_MAX\_RT option that has a valid value for INF\_MAX\_RT, the client MUST set its internal INF\_MAX\_RT parameter to the value contained in the INF\_MAX\_RT option. This value of INF\_MAX\_RT is then used by the retransmission mechanism defined in Section 15 and Section 19.1.5.

Updated INF\_MAX\_RT value applies only to the network interface on which the client received INF\_MAX\_RT option.

## 24. Security Considerations

The threat to DHCP is inherently an insider threat (assuming a properly configured network where DHCPv6 ports are blocked on the perimeter gateways of the enterprise). Regardless of the gateway

configuration, however, the potential attacks by insiders and outsiders are the same.

Use of manually configured preshared keys for IPsec between relay agents and servers does not defend against replayed DHCP messages. Replayed messages can represent a DOS attack through exhaustion of processing resources, but not through mis-configuration or exhaustion of other resources such as assignable addresses.

One attack specific to a DHCP client is the establishment of a malicious server with the intent of providing incorrect configuration information to the client. The motivation for doing so may be to mount a "man in the middle" attack that causes the client to communicate with a malicious server instead of a valid server for some service such as DNS or NTP. The malicious server may also mount a denial of service attack through misconfiguration of the client that causes all network communication from the client to fail.

A malicious DHCP server might cause a client to set its SOL\_MAX\_RT and INF\_MAX\_RT parameters to an unreasonably high value with the SOL\_MAX\_RT and INF\_MAX\_RT options, which may cause an undue delay in a client completing its DHCP protocol transaction in the case no other valid response is received. Assuming the client also receives a response from a valid DHCP server, large values for SOL\_MAX\_RT and INF\_MAX\_RT will not have any effect.

There is another threat to DHCP clients from mistakenly or accidentally configured DHCP servers that answer DHCP client requests with unintentionally incorrect configuration parameters.

A DHCP client may also be subject to attack through the receipt of a Reconfigure message from a malicious server that causes the client to obtain incorrect configuration information from that server. Note that although a client sends its response (Renew or Information-request message) through a relay agent and, therefore, that response will only be received by servers to which DHCP messages are relayed, a malicious server could send a Reconfigure message to a client, followed (after an appropriate delay) by a Reply message that would be accepted by the client. Thus, a malicious server that is not on the network path between the client and the server may still be able to mount a Reconfigure attack on a client. The use of transaction IDs that are cryptographically sound and cannot easily be predicted will also reduce the probability that such an attack will be successful.

The threat specific to a DHCP server is an invalid client masquerading as a valid client. The motivation for this may be for

theft of service, or to circumvent auditing for any number of nefarious purposes.

The threat common to both the client and the server is the resource "denial of service" (DoS) attack. These attacks typically involve the exhaustion of available addresses, or the exhaustion of CPU or network bandwidth, and are present anytime there is a shared resource.

In the case where relay agents add additional options to Relay Forward messages, the messages exchanged between relay agents and servers may be used to mount a "man in the middle" or denial of service attack.

This threat model does not consider the privacy of the contents of DHCP messages to be important. DHCP is not used to exchange authentication or configuration information that must be kept secret from other networks nodes.

DHCP authentication provides for authentication of the identity of DHCP clients and servers, and for the integrity of messages delivered between DHCP clients and servers. DHCP authentication does not provide any privacy for the contents of DHCP messages.

The Delayed Authentication protocol described in Section 22.4 uses a secret key that is shared between a client and a server. The use of a "DHCP realm" in the shared key allows identification of administrative domains so that a client can select the appropriate key or keys when roaming between administrative domains. However, the Delayed Authentication protocol does not define any mechanism for sharing of keys, so a client may require separate keys for each administrative domain it encounters. The use of shared keys may not scale well and does not provide for repudiation of compromised keys. This protocol is focused on solving the intradomain problem where the out-of-band exchange of a shared key is feasible.

Because of the opportunity for attack through the Reconfigure message, a DHCP client MUST discard any Reconfigure message that does not include authentication or that does not pass the validation process for the authentication protocol.

The Reconfigure Key protocol described in Section 22.5 provides protection against the use of a Reconfigure message by a malicious DHCP server to mount a denial of service or man-in-the-middle attack on a client. This protocol can be compromised by an attacker that can intercept the initial message in which the DHCP server sends the key to the client.

Communication between a server and a relay agent, and communication between relay agents, can be secured through the use of IPsec, as described in Section 22.1. The use of manual configuration and installation of static keys are acceptable in this instance because relay agents and the server will belong to the same administrative domain and the relay agents will require other specific configuration (for example, configuration of the DHCP server address) as well as the IPsec configuration.

A rogue delegating router can issue bogus prefixes to a requesting router. This may cause denial of service due to unreachability.

A malicious requesting router may be able to mount a denial of service attack by repeated requests for delegated prefixes that exhaust the delegating router's available prefixes.

To guard against attacks through prefix delegation, requesting routers and delegating routers SHOULD use DHCP authentication as described in Section 22. For point to point links, where one trusts that there is no man in the middle, or one trusts layer two authentication, DHCP authentication or IPsec may not be necessary. Because a requesting router and delegating routers must each have at least one assigned IPv6 address, the routers may be able to use IPsec for authentication of DHCPv6 messages. The details of using IPsec for DHCPv6 are under development.

Networks configured with delegated prefixes should be configured to preclude intentional or inadvertent inappropriate advertisement of these prefixes.

## 25. IANA Considerations

This document does not define any new DHCPv6 name spaces or definitions.

IANA is requested to update the <http://www.iana.org/assignments/dhcpv6-parameters/dhcpv6-parameters.xhtml> page to add a reference to this document for definitions previously created by [RFC3315], [RFC3633], and [RFC7083].

## 26. Acknowledgments

The following people are authors of the original RFC 3315: Ralph Droms, Jim Bound, Bernie Volz, Ted Lemon, Charles Perkins, and Mike Carney. The following people are authors of the original RFC 3633: Ole Troan and Ralph Droms. This document is merely a refinement of their work and would not be possible without their original work.

A number of additional people have contributed to identifying issues with RFC 3315 and RFC 3633 and proposed resolutions to these issues as reflected in this document (in no particular order): Ole Troan, Robert Marks, Leaf Yeh, Tim Winters, Michelle Cotton, Pablo Armando, John Brzozowski, Suresh Krishnan, Hideshi Enokihara, Alexandru Petrescu, Yukiyo Akisada, Tatuya Jinmei, Fred Templin. With special thanks to Ralph Droms for answering many questions related to the original RFC 3315 work.

The following acknowledgements are from the original RFC 3315 and RFC 3633:

Thanks to the DHC Working Group and the members of the IETF for their time and input into the specification. In particular, thanks also for the consistent input, ideas, and review by (in alphabetical order) Bernard Aboba, Bill Arbaugh, Thirumalesh Bhat, Steve Bellovin, A. K. Vijayabhaskar, Brian Carpenter, Matt Crawford, Steve Deering, Francis Dupont, Dave Forster, Brian Haberman, Richard Hussong, Tatuya Jinmei, Kim Kinnear, Fredrik Lindholm, Tony Lindstrom, Josh Littlefield, Gerald Maguire, Jack McCann, Shin Miyakawa, Thomas Narten, Erik Nordmark, Jarno Rajahalme, Yakov Rekhter, Pekka Savola, Mark Stapp, Matt Thomas, Sue Thomson, Tatuya Jinmei, Bernie Volz, Trevor Warwick, Phil Wells and Toshi Yamasaki.

Thanks to Steve Deering and Bob Hinden, who have consistently taken the time to discuss the more complex parts of the IPv6 specifications.

And, thanks to Steve Deering for pointing out at IETF 51 in London that the DHCPv6 specification has the highest revision number of any Internet Draft.

## 27. References

### 27.1. Normative References

- [RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.
- [RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", STD 37, RFC 826, November 1982.
- [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.
- [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.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.
- [RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.
- [RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.
- [RFC2526] Johnson, D. and S. Deering, "Reserved IPv6 Subnet Anycast Addresses", RFC 2526, March 1999.
- [RFC3118] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.
- [RFC3646] Droms, R., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, December 2003.
- [RFC4075] Kalusivalingam, V., "Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6", RFC 4075, May 2005.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, 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.
- [RFC6221] Miles, D., Ooghe, S., Dec, W., Krishnan, S., and A. Kavanagh, "Lightweight DHCPv6 Relay Agent", RFC 6221, May 2011.
- [RFC6355] Narten, T. and J. Johnson, "Definition of the UUID-Based DHCPv6 Unique Identifier (DUID-UUID)", RFC 6355, August 2011.
- [RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.
- [RFC7083] Droms, R., "Modification to Default Values of SOL\_MAX\_RT and INF\_MAX\_RT", RFC 7083, 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.
- [RFC7283] Cui, Y., Sun, Q., and T. Lemon, "Handling Unknown DHCPv6 Messages", RFC 7283, July 2014.

## 27.2. Informative References

- [I-D.ietf-dhc-topo-conf] Lemon, T. and T. Mrugalski, "Customizing DHCP Configuration on the Basis of Network Topology", draft-ietf-dhc-topo-conf-04 (work in progress), January 2015.
- [IANA-PEN] IANA, "Private Enterprise Numbers registry <http://www.iana.org/assignments/enterprise-numbers>", .
- [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

- [RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.
- [RFC2462] Thomson, S. and T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.
- [RFC3041] Narten, T. and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, January 2001.
- [RFC3162] Aboba, B., Zorn, G., and D. Mitton, "RADIUS and IPv6", RFC 3162, August 2001.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.
- [RFC3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, April 2004.
- [RFC3769] Miyakawa, S. and R. Droms, "Requirements for IPv6 Prefix Delegation", RFC 3769, June 2004.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.
- [RFC7084] Singh, H., Beebee, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, November 2013.
- [RFC7341] Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4-over-DHCPv6 (DHCP 4o6) Transport", RFC 7341, August 2014.

#### Appendix A. Changes since RFC3315

1. Incorporated RFC3315 errata (ids: 294, 1373, 2928, 1815, 3577, 2509, 295).
2. Partially incorporated RFC3315 errata id 2472 (place other IA options if NoAddrsAvail is sent in Advertise).



3. Clarified section 21.4.1 of RFC3315 by defining length of "key ID" field and specifying that 'DHCP realm' is Domain Name encoded as per section 8 of RFC3315. Ticket #43.
4. Added DUID-UUID and reference to RFC6355. Ticket #54.
5. Specified a minimum length for the DUID in section "9.1. DUID Contents". Ticket #39.
6. Removed the use of term "sub-options" from section "19.1.1. Creation and Transmission of Reconfigure Messages". Ticket #40.
7. Added text to section 22.6 "IA Address Option" about the usage of unspecified address to express the client hints for Preferred and Valid lifetimes. Ticket #45.
8. Updated text in 21.4.2 of RFC3315 ("Message Validation") as suggested in section 3.1 of draft-ietf-dhc-dhcpv6-clarify-auth-01. Ticket #87.
9. Merged RFC7083, "Modification to Default Values of SOL\_MAX\_RT and INF\_MAX\_RT", into this document. Ticket #51.
10. Incorporated RFC3315 errata (id 2471), into section 17.1.3. Ticket #25.
11. Added text that relay agents MUST NOT modify the relayed message to section 20.1.2. Ticket #57.
12. Modified the text in section 21.4.4.5, Receiving Reply Messages, to remove special treatment of a Reply validation failure (client ignores message). Ticket #89.
13. Appendix C updated: Authentication option is no longer allowed in Relay-forward and Relay-reply messages, ORO is no longer allowed in Confirm, Release and Decline messages; Preference option is no longer allowed in Reply messages (only in Advertise). Ticket #10.
14. Removed "silently" from several instances of "silently ignores" or "silently" discards. It is up to software vendor if and how to log such events (debug log message, event log, message pop-up etc.). Ticket #50.
15. Clarified that: there should be no more than one instance of Vendor Class option with a given Enterprise Number; that one instance of Vendor Class can contain multiple encapsulated

- options; the same applies to Vendor Specific Information option. Ticket #22.
16. Clarified relay agent definition. Ticket #12.
  17. Changed REL\_MAX\_RC and DEC\_MAX\_RC defaults from 5 to 4 and added retry to parameter description. Ticket #84.
  18. Clarify handling process for Vendor-specific Information Option and Vendor Class Option. Ticket #20.
  19. Replace "monotonic" with "strictly monotonic" in Section 21.3. Ticket #11.
  20. Incorporate everything of RFC 6644, except for Security Considerations Section, which has already covered in a more abstracted way. Ticket #55 & #56.
  21. Clarify the server behavior process when a client violates Delayed Authentication Protocol, in Section 21.4. Ticket #90.
  22. Updated titles of sections 19.4.2. and 19.4.4. to include Rebind messages.
  23. Applied many of the review comments from a review done by Fred Templin in August 2006. Ticket #14.
  24. Reworded the first paragraph of Section 15 to relax the "SHOULD" requirement to drop the messages which contain the options not expected in the current message. Ticket #17.
  25. Changed WG to DHC, added keywords
  26. Loosened requirements for DUID-EN, so that DUID type can be used for virtual machines. Ticket #16.
  27. Clarified that IA may contain other resources than just address. Ticket #93.
  28. Clarified that most options are singletons (i.e. can appear only once). Ticket #83.
  29. Merged sections 1 (Ticket #96), 2 (Ticket #97), 3 (Ticket #98), 4 (Ticket #99), 6 (Ticket #101), 8 (Ticket #103), 9 (Ticket #104), 10 (Ticket #105), 11 (Ticket #106), 13 (Ticket #108), 14 (Ticket #109), 15 (Ticket #110), 16 (Ticket #111), 17 (Ticket #112) and 19 (Ticket #113) from RFC3633 (Prefix Delegation).

30. Clarified that encapsulated options must be requested using top level ORO (ticket #38).
31. Clarified that configuration for interface X should be requested over interface X (ticket #48).
32. CONFIRM is now an optional message (MUST send Confirm eased to SHOULD) (ticket #120).
33. Added reference to RFC7227: DHCPv6 Option Guidelines (ticket #121).
34. Added new section 5 providing an overview of DHCPv6 operational modes and removed two prefix delegation sections from section 1. See tickets #53, #100, and #102.
35. Addressed ticket #115 - don't use DHCPv6 for DHCPv4 configuration.
36. Revised IANA Considerations based on ticket #117.
37. Updated IAID description in the terminology with the clarification that the IAID is unique among IAs of a specific type, rather than globally unique among all IAs (ticket #94).
38. Merged Section 12 from RFC3633 (ticket #107)
39. Clarified behavior for unknown messages (RFC7283), ticket #58.
40. Addressed tickets #123 and #126, and clarified that the client SHOULD abandon its bindings when restarts the server solicitation.
41. Clarified link-address field usage, ticket #73.

#### Appendix B. Changes since RFC3633

1. Incorporated RFC3633 errata (ids: 248, 1880, 2468, 2469, 2470, 3736)
2. ...

#### Appendix C. Appearance of Options in Message Types

The following table indicates with a "\*" the options are allowed in each DHCP message type:

	Client ID	Server ID	IA_NA IA_TA	IA_PD	Option Request	Pref	Elap. Time	Relay Msg.	Auth.	Server Unicast
Solicit	*		*	*	*		*		*	
Advert.	*	*	*	*		*			*	
Request	*	*	*	*	*		*		*	
Confirm	*		*				*		*	
Renew	*	*	*	*	*		*		*	
Rebind	*		*	*	*		*		*	
Decline	*	*	*	*			*		*	
Release	*	*	*	*			*		*	
Reply	*	*	*	*					*	*
Reconf.	*	*			*				*	
Inform.	*	(see note)			*		*		*	
R-forw.								*		
R-repl.								*		

## NOTE:

Only included in Information-request messages that are sent in response to a Reconfigure (see Section 20.4.3).

	Status Code	Rap. Comm.	User Class	Vendor Class	Vendor Spec.	Inter. ID	Recon. Msg.	Recon. Accept	SOL_MAX_RT INF_MAX_RT
Solicit		*	*	*	*			*	
Advert.	*		*	*	*			*	*
Request			*	*	*			*	
Confirm			*	*	*				
Renew			*	*	*			*	
Rebind			*	*	*			*	
Decline			*	*	*				
Release			*	*	*				
Reply	*	*	*	*	*			*	*
Reconf.							*		
Inform.			*	*	*			*	
R-forw.			*	*	*	*			
R-repl.			*	*	*	*			

## Appendix D. Appearance of Options in the Options Field of DHCP Options

The following table indicates with a "\*" where options can appear in the options field of other options:

	Option	IA_NA/				Relay	Relay
	Field	IA_TA	IAADDR	IA_PD	IAPREFIX	Forw.	Reply
Client ID	*						
Server ID	*						
IA_NA/IA_TA	*						
IAADDR		*					
IA_PD	*						
IAPREFIX				*			
ORO	*						
Preference	*						
Elapsed Time	*						
Relay Message						*	*
Authentic.	*						
Server Uni.	*						
Status Code	*	*		*			
Rapid Comm.	*						
User Class	*						
Vendor Class	*						
Vendor Info.	*					*	*
Interf. ID						*	*
Reconf. MSG.	*						
Reconf. Accept	*						

Note: "Relay Forw" / "Relay Reply" options appear in the options field of the message but may only appear in these messages.

#### Authors' Addresses

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

Email: tomasz.mrugalski@gmail.com

Marcin Siodelski  
 Internet Systems Consortium, Inc.  
 950 Charter St.  
 Redwood City, CA 94063  
 USA

Email: msiodelski@gmail.com

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

Email: volz@cisco.com

Andrew Yourtchenko  
Cisco Systems, Inc.  
De Kleetlaan, 7  
Diegem B-1831  
Belgium

Email: ayourtch@cisco.com

Michael C. Richardson  
Sandelman Software Works  
470 Dawson Avenue  
Ottawa, ON K1Z 5V7  
CA

Email: mcr+ietf@sandelman.ca  
URI: <http://www.sandelman.ca/>

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

Ted Lemon  
Nominum, Inc.  
950 Charter St.  
Redwood City, CA 94043  
USA

Email: Ted.Lemon@nominum.com

Dynamic Host Configuration (DHC)  
Internet-Draft  
Intended status: Standards Track  
Expires: March 17, 2014

T. Mrugalski  
ISC  
K. Kinnear  
Cisco  
September 13, 2013

DHCPv6 Failover Design  
draft-ietf-dhc-dhcpv6-failover-design-04

Abstract

DHCPv6 defined in [RFC3315] does not offer server redundancy. This document defines a design for DHCPv6 failover, a mechanism for running two servers on the same network with capability for either server to take over clients' leases in case of server failure or network partition. This is a DHCPv6 Failover design document, it is not a protocol specification document. It is a second document in a planned series of three documents. DHCPv6 failover requirements are specified in [I-D.ietf-dhc-dhcpv6-failover-requirements]. A protocol specification document is planned to follow this document.

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. Requirements Language . . . . .	3
2. Glossary . . . . .	4
3. Introduction . . . . .	5
3.1. Design Requirements . . . . .	6
3.2. Features out of Scope: Load Balancing . . . . .	6
4. Protocol Overview . . . . .	7
4.1. Failover State Machine Overview . . . . .	8
4.2. Messages . . . . .	10
5. Connection Management . . . . .	12
5.1. Creating Connections . . . . .	12
5.2. Endpoint Identification . . . . .	13
6. Resource Allocation . . . . .	14
6.1. Proportional Allocation . . . . .	14
6.2. Independent Allocation . . . . .	17
6.3. Choosing Allocation Algorithm . . . . .	17
7. Information model . . . . .	18
8. Failover Mechanisms . . . . .	23
8.1. Time Skew . . . . .	23
8.2. Lazy updates . . . . .	23
8.3. MCLT concept . . . . .	24
8.3.1. MCLT example . . . . .	25
8.4. Unreachability detection . . . . .	26
8.5. Re-allocating Leases . . . . .	27
8.6. Sending Binding Update . . . . .	28
8.7. Receiving Binding Update . . . . .	29
8.8. Conflict Resolution . . . . .	30
8.9. Acknowledging Reception . . . . .	32
9. Endpoint States . . . . .	32
9.1. State Machine Operation . . . . .	32
9.2. State Machine Initialization . . . . .	35
9.3. STARTUP State . . . . .	35
9.3.1. Operation in STARTUP State . . . . .	36
9.3.2. Transition Out of STARTUP State . . . . .	36
9.4. PARTNER-DOWN State . . . . .	38
9.4.1. Operation in PARTNER-DOWN State . . . . .	38
9.4.2. Transition Out of PARTNER-DOWN State . . . . .	39
9.5. RECOVER State . . . . .	39
9.5.1. Operation in RECOVER State . . . . .	39
9.5.2. Transition Out of RECOVER State . . . . .	40
9.6. RECOVER-WAIT State . . . . .	41



9.6.1. Operation in RECOVER-WAIT State . . . . .	41
9.6.2. Transition Out of RECOVER-WAIT State . . . . .	41
9.7. RECOVER-DONE State . . . . .	42
9.7.1. Operation in RECOVER-DONE State . . . . .	42
9.7.2. Transition Out of RECOVER-DONE State . . . . .	42
9.8. NORMAL State . . . . .	43
9.8.1. Operation in NORMAL State . . . . .	43
9.8.2. Transition Out of NORMAL State . . . . .	44
9.9. COMMUNICATIONS-INTERRUPTED State . . . . .	44
9.9.1. Operation in COMMUNICATIONS-INTERRUPTED State . . . . .	45
9.9.2. Transition Out of COMMUNICATIONS-INTERRUPTED State . . . . .	45
9.10. POTENTIAL-CONFLICT State . . . . .	47
9.10.1. Operation in POTENTIAL-CONFLICT State . . . . .	47
9.10.2. Transition Out of POTENTIAL-CONFLICT State . . . . .	47
9.11. RESOLUTION-INTERRUPTED State . . . . .	48
9.11.1. Operation in RESOLUTION-INTERRUPTED State . . . . .	49
9.11.2. Transition Out of RESOLUTION-INTERRUPTED State . . . . .	49
9.12. CONFLICT-DONE State . . . . .	49
9.12.1. Operation in CONFLICT-DONE State . . . . .	49
9.12.2. Transition Out of CONFLICT-DONE State . . . . .	50
10. Proposed extensions . . . . .	50
10.1. Active-active mode . . . . .	50
11. Dynamic DNS Considerations . . . . .	50
11.1. Relationship between failover and dynamic DNS update . . . . .	51
11.2. Exchanging DDNS Information . . . . .	52
11.3. Adding RRs to the DNS . . . . .	54
11.4. Deleting RRs from the DNS . . . . .	54
11.5. Name Assignment with No Update of DNS . . . . .	55
12. Reservations and failover . . . . .	55
13. Security Considerations . . . . .	57
14. IANA Considerations . . . . .	57
15. Acknowledgements . . . . .	57
16. References . . . . .	58
16.1. Normative References . . . . .	58
16.2. Informative References . . . . .	58
Authors' Addresses . . . . .	59

## 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. Glossary

This is a supplemental glossary that should be combined with definitions in Section 3 of [I-D.ietf-dhc-dhcpv6-failover-requirements].

- o auto-partner-down - a capability where a failover server will move from COMMUNICATIONS-INTERRUPTED state to PARTNER-DOWN state automatically, without operator intervention.
- o DDNS - Dynamic DNS. Typically used as an acronym referring to dynamic update of the DNS.
- o Failover endpoint - The failover protocol allows for there to be a unique failover 'endpoint' for each failover relationship in which a failover server participates. The failover relationship is defined by a relationship name, and includes the failover partner IP address, the role this server takes with respect to that partner (primary or secondary), and the prefixes associated with that relationship. Note that a single prefix can only be associated with a single failover relationship. This failover endpoint can take actions and hold unique states. Typically, there is one failover endpoint per partner (server), although there may be more. 'Server' and 'failover endpoint' are synonymous only if the server participates in only one failover relationship. However, for the sake of simplicity 'Server' is used throughout the document to refer to a failover endpoint unless to do so would be confusing.
- o Failover communication - all messages exchanged between partners.
- o Independent Allocation - an allocation algorithm that splits the available pool of resources between the primary and secondary servers that is particularly well suited for vast pools (i.e. when available resources are not expected to deplete). See Section 6.2 for details.
- o Lease - an association of a DHCPv6 client with an IPv6 address or delegated prefix.
- o Partner - name of the other DHCPv6 server that participates in failover relationship. When the role (primary or secondary) is not important, the other server is referred to as a "failover partner" or simply partner.

- o Primary Server - First out of two DHCPv6 servers that participate in a failover relationship. In active-passive mode this is the server that handles most of the client traffic. Its failover partner is referred to as secondary server.
- o Proportional Allocation - an allocation algorithm that splits the available resources between the primary and secondary servers and maintains proportions between available resources on both. It is particularly well suited for more limited resources. See Section 6.1 for details.
- o Resource - Any type of resource that is managed by DHCPv6. Currently there are three types of such resources defined: a non-temporary IPv6 address, a temporary IPv6 address, and an IPv6 prefix. Other resource types may be defined in the future.
- o Responsive - A server that is responsive, will respond to DHCPv6 client requests.
- o Secondary Server - Second of two DHCPv6 servers that participate in a failover relationship. Its failover partner is referred to as the primary server. In active-passive mode this server (the secondary) typically does not handle client traffic and acts as a backup.
- o Server - A DHCPv6 server that implements DHCPv6 failover. 'Server' and 'failover endpoint' are synonymous only if the server participates in only one failover relationship.
- o Unresponsive - A server that is unresponsive will not respond to DHCPv6 client requests.

### 3. Introduction

The failover protocol design provides a means for cooperating DHCPv6 servers to work together to provide a DHCPv6 service with availability that is increased beyond that which could be provided by a single DHCPv6 server operating alone. It is designed to protect DHCPv6 clients against server unreachability, including server failure and network partition. It is possible to deploy exactly two servers that are able to continue providing a lease on an IPv6 address [RFC3315] or on an IPv6 prefix [RFC3633] without the DHCPv6 client experiencing lease expiration or a reassignment of a lease to a different IPv6 address (or prefix) in the event of failure by one or the other of the two servers.

This protocol defines active-passive mode, sometimes also called a hot standby model. This means that during normal operation one

server is active (i.e. actively responds to clients' requests) while the second is passive (i.e. it does receive clients' requests, but does not respond to them and only maintains a copy of lease database and is ready to take over incoming queries in case of primary server failure). Active-active mode (i.e. both servers actively handling clients' requests) is currently not supported for the sake of simplicity. Such a mode is likely to be defined as an extension at a later time and will probably be based on [I-D.ietf-dhc-dhcpv6-load-balancing].

The failover protocol is designed to provide lease stability for leases with lease times beyond a short period. Due in part to the additional overhead required as well as requirements to handle time skew between failover partners (See Section 8.1), failover is not suitable for leases shorter than 30 seconds. The DHCPv6 Failover protocol MUST NOT be used for leases shorter than 30 seconds.

This design attempts to fulfill all DHCPv6 failover requirements defined in [I-D.ietf-dhc-dhcpv6-failover-requirements].

### 3.1. Design Requirements

The following requirements are not related to failover the mechanism in general, but rather to this particular design.

1. Minimize Asymmetry - while there are two distinct roles in failover (primary and secondary server), the differences between those two roles should be as small as possible. This will yield a simpler design as well as a simpler implementation of that design.

### 3.2. Features out of Scope: Load Balancing

While it is tempting to extend DHCPv6 failover mechanism to also offer load balancing, as DHCPv4 failover did, this design does not do that. Here is the reasoning for this decision. In general case (not related to failover) load balancing solutions are used when each server is not able to handle total incoming traffic. However, by the very definition, DHCPv6 failover is supposed to assume service availability despite failure of one server. That leads to the conclusion that each server must be able to handle all of the traffic. Therefore in properly provisioned setup, load balancing is not needed.

It is likely that active-active mode that is essentially a load balancing will be defined as an extension in the near future.

#### 4. Protocol Overview

The DHCPv6 Failover Protocol is defined as a communication between failover partners with all associated algorithms and mechanisms. Failover communication is conducted over a TCP connection established between the partners. The protocol reuses the framing format specified in Section 5.1 of DHCPv6 Bulk Leasequery [RFC5460], but uses different message types. New failover-specific message types are listed in Section 4.2. All information is sent over the connection as typical DHCPv6 messages that convey DHCPv6 options, following the format defined in Section 22.1 of [RFC3315].

After initialization, the primary server establishes a TCP connection with its partner. The primary server sends a CONNECT message with initial parameters. Secondary server responds with CONNECTACK.

If the primary server cannot immediately establish a connection with its partner, it will continue to attempt to establish a connection. See Section 5.1 for details.

Depending on the failover state of each partner, they MUST initiate one of the binding update procedures. Each server MAY send an UPDREQ message to request its partner to send all updates that have not been sent yet (this case applies when the partner has an existing database and wants to update it). Alternatively, a server MAY choose to send an UPDREQALL message to request a full lease database transmission including all leases (this case applies in case of booting up a new server after installation, corruption or complete loss of database, or other catastrophic failure).

Servers exchange lease information by using BNDUPD messages. Depending on the local and remote state of a lease, a server may either accept or reject the update. Reception of lease update information is confirmed by responding with a BNDACK message with appropriate status. The majority of the messages sent over a failover TCP connection consists of BNDUPD and BNDACK messages.

A subset of available resources (addresses or prefixes) is reserved for secondary server use. This is required for handling a case where both servers are able to communicate with clients, but unable to communicate with each other. After the initial connection is established, the secondary server requests a pool of available addresses or prefixes by sending a POOLREQ message. The primary server assigns addresses or prefixes to the secondary by sending a series of BNDUPD messages. When this process is complete, the primary server sends a POOLRESP message to the secondary server. The secondary server may initiate such pool request at any time when in communication with primary server.

Failover servers use a lazy update mechanism to update their failover partner about changes to their lease state database. After a server performs any modifications to its lease state database (assign a new lease, extend, release or expire existing lease), it sends its response to the client's request first (performing the "regular" DHCPv6 operation) and then informs its failover partner using a BNDUPD message. This BNDUPD message SHOULD be sent soon after the response is sent to the DHCPv6 client, but there is no specific requirement of a minimum time in which to do so.

The major problem with a lazy update mechanism is when the server crashes after sending a response to client, but before sending the lazy update to its partner (or when communication between partners is interrupted). To solve this problem, the concept known as the Maximum Client Lead Time (initially designed for DHCPv4 failover) is used. The MCLT is the maximum amount of time that one server can extend a lease for a client's binding beyond the time known by its failover partner. See Section 8.3 for a detailed description how the MCLT affects assigned lifetimes.

Servers verify each others availability by periodically exchanging CONTACT messages. See Section 8.4 for discussion about detecting a partner's unreachability.

A server that is being shut down transmits a DISCONNECT message, closes the connection with its failover partner and stops operation. A Server SHOULD transmit any pending lease updates before transmitting DISCONNECT message.

#### 4.1. Failover State Machine Overview

The following section provides a simplified description of all states. For the sake of clarity and simplicity, it omits important details. For a complete description, see Section 9. In case of a disagreement between the simplified and complete description, please follow Section 9.

Each server MUST be in one of the well defines states. Depending on its current state a server may be either responsive (responds to clients' queries) or unresponsive (clients' queries are ignored).

A server starts its operation in the short-lived STARTUP state. A server determines its partner reachability and state and sets its own state based on that determination. It typically returns back to the state it was in before shutdown, though the details can be complicated. See Section 9.3.2.

During typical operation when servers maintain communication, both are in NORMAL state. In that state only the primary responds to clients' requests. The secondary server is unresponsive.

If a server discovers that its partner is no longer reachable, it goes to COMMUNICATIONS-INTERRUPTED state. A server must be extra cautious as it can't distinguish if its partner is down or just communication between servers is interrupted. Since communication between partners is not possible, a server must act on the assumption that its partner is up. A failover server must follow a defined procedure, in particular, it MUST NOT extend any lease more than the MCLT beyond its partner's knowledge of the lease expiration time. This imposes an additional burden on the server, in that clients will return to the server for lease renewals more frequently than they would otherwise. Therefore it is not recommended to operate for prolonged periods in this state. Once communication is reestablished, a server may go into NORMAL, POTENTIAL-CONFLICT or PARTNER-DOWN state. It may also stay in COMMUNICATIONS-INTERRUPTED state if certain conditions are met.

Once a server is switched into PARTNER-DOWN (when auto-partner-down is used or as a result of administrative action), it can extend leases, regardless of the original server that initially granted the lease. In that state server handles leases from its own pool, but once its own pool is depleted is also able to serve pool from its downed partner. Some MCLT restrictions no longer apply, but the MCLT still affects whether or not a particular lease can be given to a different client. See Section 9.4.1 for details. Operation in this mode is less demanding for the server that remains operational, than in COMMUNICATIONS-INTERRUPTED state, but PARTNER-DOWN does not offer any kind of redundancy. Even when in PARTNER-DOWN state, a failover server continues to attempt to connect with its failover partner.

A server switches into RECOVER state when any of a variety of conditions are encountered:

- o When a backup server contacts its failover partner for the first time.
- o When either server discovers that its failover partner has contacted it before but it has no local record of this contact. If the record of previous contact is held in the lease-state database, then this situation implies that the server has lost its lease state database.
- o When its failover partner is in PARTNER-DOWN state.

Any of these conditions signal that the server needs to refresh its lease-state database from its partner. Once this operation is complete, it switches to RECOVER-WAIT and later to RECOVER-DONE. See Section 9.6.2.

Once servers reestablish connection, they discover each others' state. Depending on the conditions, they may return to NORMAL or move to POTENTIAL-CONFLICT if the partner is in a state that doesn't allow a simple re-integration of the server's lease state databases. It is a goal of this protocol to minimize the possibility that POTENTIAL-CONFLICT state is ever entered. Servers running in POTENTIAL-CONFLICT do not respond to clients' requests and work only on resolving potential conflicts. Once outstanding lease updates are exchanged, servers move to CONFLICT-DONE or NORMAL states.

Servers that are recovering from potential conflicts and loose communication, switch to RESOLUTION-INTERRUPTED.

A server that is being shut down sends a DISCONNECT message. See Section 4.2. A server that receives a DISCONNECT message moves into COMMUNICATIONS-INTERRUPTED state.

#### 4.2. Messages

The failover protocol is centered around the message exchanges used by one server to update its partner and respond to received updates. It should be noted that no specific formats or message type values are assigned in this document. Appropriate implementation details will be specified in a separate protocol specification document. The following list enumerates these messages:

- o BNDUPD - The binding update message is used to send the binding lease changes to the partner. One message may contain one or more lease updates. The partner is expected to respond with a BNDACK message.
- o BNDACK - The binding acknowledgement is used for confirmation of the received BNDUPD message. It may contain a positive or negative response (e.g. due to detected lease conflict).
- o POOLREQ - The Pool Request message is used by one server (typically secondary) to request allocation of resources (addresses or prefixes) from its partner. The partner responds with POOLRESP.
- o POOLRESP - The Pool Response message is used by one server (typically primary) to indicate that it has responded to its partner's request for resources allocation.



- o UPDREQ - The update request message is used by one server to request that its partner send all binding database changes that have not been sent and confirmed already. Requested partner is expected to respond with zero or more BNDUPD messages, followed by UPDDONE that signals end of updates.
- o UPDREQALL - The update request all is used by one server to request that all binding database information be sent in order to recover from a total loss of its binding database by the requesting server. Requested server responds with zero or more BNDUPD messages, followed by UPDDONE that signal end of updates.
- o UPDDONE - The update done message is used by the server responding to an UPDREQ or UPDREQALL to indicate that all requested updates have been sent by the responding server and acked by the requesting server.
- o CONNECT - The connect message is used by the primary server to establish a high level connection with the other server, and to transmit several important configuration data items between the servers. The partner is expected to confirm by responding with CONNECTACK message.
- o CONNECTACK - The connect acknowledgement message is used by the secondary server to respond to a CONNECT message from the primary server.
- o DISCONNECT - The disconnect message is used by either server when closing a connection and shutting down. No response is required for this message.
- o STATE - The state message is used by either server to inform its partner about a change of failover state. In some cases it may be used to also inform the partner about current state, e.g. after connection is established in COMMUNICATIONS-INTERRUPTED or PARTNER-DOWN states.
- o CONTACT - The contact message is used by either server to ensure that the other server continues to see the connection as operational. It MUST be transmitted periodically over every established connection if other message traffic is not flowing, and it MAY be sent at any time.

## 5. Connection Management

### 5.1. Creating Connections

Every primary server implementing the failover protocol **MUST** attempt to connect to all of its partners periodically, where the period is implementation dependent and **SHOULD** be configurable. In the event that a connection has been rejected by a **CONNECTACK** message with a reject-reason option contained in it or a **DISCONNECT** message, a server **SHOULD** reduce the frequency with which it attempts to connect to that server but it **MUST** continue to attempt to connect periodically.

Every secondary server implementing the failover protocol **MUST** listen for connection attempts from the primary server.

When a connection attempt succeeds, the primary server which has initiated the connection attempt **MUST** send a **CONNECT** message down the connection.

When a connection attempt is received, the only information that the receiving server has is the IP address of the partner initiating a connection. If it has any relationships with the connecting server for which it is a secondary server, it should just await the **CONNECT** message to determine which relationship this connection is to serve.

If it has no secondary relationships with the connecting server, it **MUST** drop the connection. The goal is to limit the resources expended dealing with attempts to create a spurious failover connection.

To summarize -- a primary server **MUST** use a connection that it has initiated in order to send a **CONNECT** message. Every server that is a secondary server in a relationship simply listens for connection attempts from the primary server.

Once a connection is established, the primary server **MUST** send a **CONNECT** message across the connection. A secondary server **MUST** wait for the **CONNECT** message from a primary server. If the secondary server doesn't receive a **CONNECT** message from the primary server in an installation dependent amount of time, it **MAY** drop the connection.

Every **CONNECT** message includes a **TLS-request** option, and if the **CONNECTACK** message does not reject the **CONNECT** message and the **TLS-reply** option says **TLS MUST** be used, then the servers will immediately enter into **TLS** negotiation.

Once TLS negotiation is complete, the primary server MUST resend the CONNECT message on the newly secured TLS connection and then wait for the CONNECTACK message in response. The TLS-request and TLS-reply options MUST NOT appear in either this second CONNECT or its associated CONNECTACK message as they had in the first messages.

The second message sent over a new connection (either a bare TCP connection or a connection utilizing TLS) is a STATE message. Upon the receipt of this message, the receiver can consider communications up.

## 5.2. Endpoint Identification

The proper operation of the failover protocol requires more than the transmission of messages between one server and the other. Each endpoint might seem to be a single DHCPv6 server, but in fact there are situations where additional flexibility in configuration is useful. A failover endpoint is always associated with a set of DHCPv6 prefixes that are configured on the DHCPv6 server where the endpoint appears. A DHCPv6 prefix MUST NOT be associated with more than one failover endpoint.

The failover protocol SHOULD be configured with one failover relationship between each pair of failover servers. In this case there is one failover endpoint for that relationship on each failover partner. This failover relationship MUST have a unique name.

There is typically little need for additional relationships between any two servers but there MAY be more than one failover relationship between two servers -- however each MUST have a unique relationship name.

Any failover endpoint can take actions and hold unique states.

This document frequently describes the behavior of the protocol in terms of primary and secondary servers, not primary and secondary failover endpoints. However, it is important to remember that every 'server' described in this document is in reality a failover endpoint that resides in a particular process, and that several failover endpoints may reside in the same server process.

It is not the case that there is a unique failover endpoint for each prefix that participates in a failover relationship. On one server, there is (typically) one failover endpoint per partner, regardless of how many prefixes are managed by that combination of partner and role. Conversely, on a particular server, any given prefix will be associated with exactly one failover endpoint.

When a connection is received from the partner, the unique failover endpoint to which the message is directed is determined solely by the IP address of the partner, the relationship-name, and the role of the receiving server.

## 6. Resource Allocation

Currently there are two allocation algorithms defined for resources (addresses or prefixes). Additional allocation schemes may be defined as future extensions.

1. Proportional Allocation - This allocation algorithm is a direct application of the algorithm defined in [dhcpv4-failover] to DHCPv6. Remaining available resources are split between the primary and secondary servers in a configured proportion. Released resources are always returned to the primary server. Primary and secondary servers may initiate a rebalancing procedure when disparity between resources available to each server reaches a preconfigured threshold. Only resources that are not leased to any clients are "owned" by one of the servers. This algorithm is particularly well suited for scenarios where amount of available resources is limited, as may be the case with prefix delegation. See Section 6.1 for details.
2. Independent Allocation - This allocation algorithm also assumes that available resources are split between primary and secondary servers. In this case, however, resources are assigned to a specific server for all time, regardless if they are available or currently used. This algorithm is much simpler than proportional allocation, because resource imbalance doesn't have to be checked and there is no rebalancing for independent allocation. This algorithm is particularly well suited for scenarios where there is an abundance of available resources which is typically the case for DHCPv6 address allocation. See Section 6.2 for details.

### 6.1. Proportional Allocation

In this allocation scheme, each server has its own pool of available resources. Remaining available resources are split between the primary and secondary servers in a configured proportion. Note that a resource is not "owned" by a particular server throughout its entire lifetime. Only a resource which is available is "owned" by a particular server -- once it has been leased to a client, it is not owned by either failover partner. When it finally becomes available again, it will be owned initially by the primary server, and it may or may not be allocated to the secondary server by the primary server.

The flow of a resource is as follows: initially a resource is owned by the primary server. It may be allocated to the secondary server if it is available, and then it is owned by the secondary server. Either server can allocate available resources which they own to clients, in which case they cease to own them. When the client releases the resource or the lease on it expires, it will again become available and will be owned by the primary.

A resource will not become owned by the server which allocated it initially when it is released or the lease expires because, in general, that server will have had to replenish its pool of available resources well in advance of any likely lease expirations. Thus, having a particular resource cycle back to the secondary might well put the secondary more out of balance with respect to the primary instead of enhancing the balance of available addresses or prefixes between them.

Pools governed by proportional allocation are used for allocation when the server is in all states, except PARTNER-DOWN. In PARTNER-DOWN state the healthy partner can allocate from either pool (both its own, and its partner's after some time constraints have elapsed). This allocation and maintenance of these address pools is an area of some sensitivity, since the goal is to maintain a more or less constant ratio of available addresses between the two servers.

The initial allocation when the servers first integrate is triggered by the POOLREQ message from the secondary to the primary. This is followed (at some point) by the POOLRESP message where the primary tells the secondary that it received and processed the POOLREQ message. The primary sends the allocated resources to the secondary via BNDUPD messages. The POOLRESP message may be sent before, during, or at the completion of the BNDUPD message exchanges that were triggered by the POOLREQ message. The POOLREQ/POOLRESP message exchange is a trigger to the primary to perform a scan of its database and to ensure that the secondary has enough resources (based on some configured ratio).

The primary server SHOULD examine some or all of its database from time to time to determine if resources should be shifted between the primary and secondary (in either direction). The POOLREQ/POOLRESP message exchange allows the secondary server to explicitly request that the primary server examine the entirety of its database to ensure that the secondary has the appropriate resources available.

Servers frequently have several kinds of resources available on a particular network segment. The failover protocol assumes that both primary and secondary servers are configured in such a way that each knows the type and number of resources on every network segment

participating in the failover protocol. The primary server is responsible for allocating the secondary server the correct proportion of available resources of each kind.

The resources are delegated to the secondary using the BNDUPD message with a state of FREE\_BACKUP, which indicates the resource is now available for allocation by the secondary. Once the message is sent, the primary MUST NOT use these resources for allocation to DHCPv6 clients.

Available resources can be delegated back to the primary server in certain cases. BNDUPD will contain state FREE for leases that were previously in FREE\_BACKUP state.

The POOLREQ/POOLRESP message exchange initiated by the secondary is valid at any time both partners remain in contact, and the primary server SHOULD, whenever it receives the POOLREQ message, scan its database of prefixes and determine if the secondary needs more resources from any of the prefixes.

In order to support a reasonably dynamic balance of the resources between the failover partners, the primary server needs to do additional work to ensure that the secondary server has as many resources as it needs (but that it doesn't have more than it needs).

The primary server SHOULD examine the balance of available resources between the primary and secondary for a particular prefix whenever the number of available resources for either the primary or secondary changes by more than a configured limit. The primary server SHOULD adjust the available resource balance as required to ensure the configured resource balance, excepting that the primary server SHOULD employ some threshold mechanism to such a balance adjustment in order to minimize the overhead of maintaining this balance.

An example of a threshold approach is: do not attempt to re-balance the prefixes on the primary and secondary until the out of balance value exceeds a configured value.

The primary server can, at any time, send an available resource to the secondary using a BNDUPD with the state FREE\_BACKUP. The primary server can attempt to take an available resource away from the secondary by sending a BNDUPD with the state FREE. If the secondary accepts the BNDUPD, then the resource is now available to the primary and not available to the secondary. Of course, the secondary MUST reject that BNDUPD if it has already used that resource for a DHCP client.

## 6.2. Independent Allocation

In this allocation scheme, available resources are permanently (until server configuration changes) split between servers. Available resources are split between the primary and secondary servers as part of initial connection establishment. Once resources are allocated to each server, there is no need to reassign them. The resource allocation is algorithmic in nature, and does not require a message exchange for each resource allocated. This algorithm is simpler than proportional allocation since it does not require a rebalancing mechanism. It assumes that the pool assigned to each server will never deplete. That is often a reasonable assumption for IPv6 addresses (e.g. servers are often assigned a /64 pool that contains many more addresses than existing electronic devices on Earth). This allocation mechanism SHOULD be used for IPv6 addresses, unless the configured address pool is small or is otherwise administratively limited.

Once each server is assigned a resource pool during initial connection establishment, it may allocate assigned resources to clients. Once a client releases a resource or its lease is expired, the returned resource returns to the pool for the server that leased it. Resources never changes servers.

Resources using the independent allocation approach are ignored when a server processes a POOLREQ message.

During COMMUNICATION-INTERRUPTED events, a partner MAY continue extending existing leases when requested by clients. A healthy partner MUST NOT lease resources that were assigned to its downed partner and later released by a client unless it is in PARTNER-DOWN state. When it is in PARTNER-DOWN state, a server SHOULD use its own pool first and then it MAY start making new assignments from its downed partner's pool. As the assumption is that independent allocation should be used only when available resources are vast and not expected to be fully used at any given time, it is very unlikely that the server will ever need to use its downed partner pools. This makes a recovery even after prolonged down-time much easier.

## 6.3. Choosing Allocation Algorithm

All implementations SHOULD support both the proportional allocation algorithm and the independent allocation algorithm. The specific requirements for support (i.e., which algorithm(s) MUST be supported), and the assignment of a specific algorithm to a specific allocation domain, would be documented in any protocol specifications that follow from this document.

The proportional allocation mechanism is more flexible as it can dynamically rebalance available resources between servers. That balance creates an additional burden for the servers and generates more traffic between servers. The proportional algorithm can be considered more efficient at managing available resources, compared to the independent algorithm. That is an important aspect when working in a network that is nearing address and/or prefix depletion.

Independent allocation can be used when the number of available resources are large and there is no realistic danger of running out of resources. Use of the independent allocation makes communication between partners simpler. It also makes recovery easier and potential conflict less likely to appear.

Typically independent allocation is used for IPv6 addresses, because even for /64 pools a server will never run out of addresses to assign, so there is no need to rebalance. For the prefix delegation mechanism, available resources are typically much smaller, so there is a danger of running out of prefixes. Therefore typically proportional allocation will be used for prefix delegations. Independent allocation still may be used, but the implication must be well understood. For example in a network that delegates /64 prefixes out of a /48 prefix (so there can be up to 65536 prefixes delegated) and a 1000 requesting routers, it is safe to use independent allocation.

It should be stressed that the independent allocation algorithm SHOULD NOT be used when the number of resources is limited and there is a realistic danger of depleting resources. If this recommendation is violated, it may lead to a case when one server denies clients due to pool depletion despite the fact that the other partner still has many resources available.

With independent allocation it is very unlikely for a remaining healthy server to allocate resources from its unavailable partner's pool. That makes recovery easier and any potential conflicts are less likely to appear.

## 7. Information model

In most DHCP servers a resource (an IP address or a prefix) can take on several different binding-status values, sometimes also called lease states. While no two DHCP server implementations probably have exactly the same possible binding-status values, [RFC3315] enforces some commonality among the general semantics of the binding-status values used by various DHCP server implementations.



In order to transmit binding database updates between one server and another using the failover protocol, some common denominator binding-status values must be defined. It is not expected that these values correspond with any actual implementation of the DHCP protocol in a DHCP server, but rather that the binding-status values defined in this document should be a common denominator of those in use by many DHCP server implementations.

The lease binding-status values defined for the failover protocol are listed below. Unless otherwise noted below, there MAY be client information associated with each of these binding-status value.

ACTIVE -- The lease is assigned to a client. Client identification data MUST appear.

EXPIRED -- indicates that a client's binding on a given lease has expired. When the partner acks the BNDUPD of an expired lease, the server sets its internal state to FREE\*. Client identification SHOULD appear.

RELEASED -- indicates that a client sent in RELEASE message. When the partner acks the BNDUPD of a released lease, the server sets its internal state to FREE\*. Client identification SHOULD appear.

FREE\* -- Once a lease is expired or released, its state becomes FREE\*. Depending on which algorithm and which pool was used to allocate a given lease, FREE\* may either mean FREE or FREE\_BACKUP. Implementations do not have to implement this FREE\* state, but may choose to switch to the destination state directly. For a clarity of representation, this transitional FREE\* state is treated as a separate state.

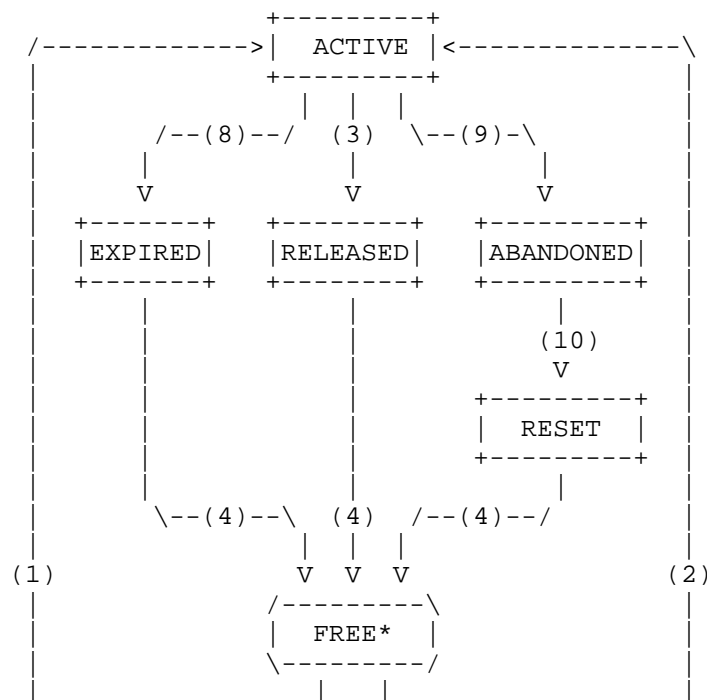
FREE -- Is used when a DHCP server needs to communicate that a resource is unused by any client, but it was not just released, expired or reset by a network administrator. When the partner acks the BNDUPD of a FREE lease, the server marks the lease as available for assignment by the primary server. Note that on a secondary server running in PARTNER-DOWN state, after waiting the MCLT, the resource MAY be allocated to a client by the secondary server. Client identification MAY appear and indicates the last client to have used this resource as a hint.

**FREE\_BACKUP** -- indicates that this resource can be allocated by the secondary server to a client at any time. Note that the primary server running in PARTNER-DOWN state, after waiting the MCLT, the resource MAY be allocated to a client by the primary server if proportional algorithm was used. Client identification MAY appear and indicates the last client to have used this resource as a hint.

**ABANDONED** -- indicates that a lease is considered unusable by the DHCP system. The primary reason for entering such state is reception of DECLINE message for said lease. Client identification MAY appear.

**RESET** -- indicates that this resource was made available by operator command. This is a distinct state so that the reason that the resource became FREE can be determined. Client identification MAY appear.

The lease state machine has been presented in Figure 1. Most states are stationary, i.e. the lease stays in a given state until external event triggers transition to another state. The only transitive state is FREE\*. Once it is reached, the state machine immediately transitions to either FREE or FREE\_BACKUP state.



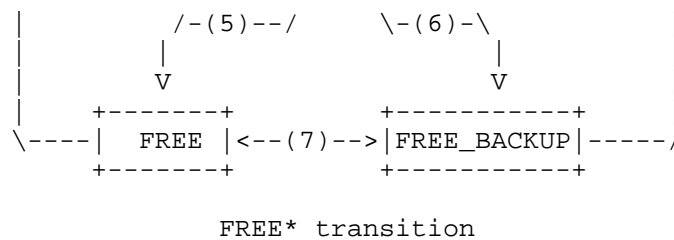


Figure 1: Lease State Machine

Transitions between states are results of the following events:

1. Primary server allocates a lease.
2. Secondary server allocates a lease.
3. Client sends RELEASE and the lease is released.
4. Partner acknowledges state change. This transition MAY also occur if the server is in PARTNER-DOWN state and the MCLT has passed since the entry in RELEASED, EXPIRED, or RESET states.
5. The lease belongs to a pool that is governed by the proportional allocation, or independent allocation is used and this lease belongs to primary server pool.
6. The lease belongs to a pool that is governed by the independent allocation and the lease belongs to the secondary server.
7. Pool rebalance event occurs (POOLREQ/POOLRESP messages are exchanged). Addresses (or prefixes) belonging to the primary server can be assigned to the secondary server pool (transition from FREE to FREE\_BACKUP) or vice versa.
8. The lease has expired.
9. DECLINE message is received or a lease is deemed unusable for other reasons.
10. An administrative action is taken to recover an abandoned lease back to usable state. This transition MAY occur due to an implementation specific handling on ABANDONED resource. One possible example of such use is a Neighbor Discovery or ICMPv6 Echo check if the address is still in use.

The resource that is no longer in use (due to expiration or release), becomes FREE\*. Depending of what allocation algorithm is used, the resource that is no longer is use, returns to primary (FREE) or secondary pool (FREE\_BACKUP). The conditions for specific transitions are depicted in Figure 2.

+-----+-----+-----+		
\Resource owner		
\-----\		
Algorithm \		
+-----+-----+-----+		
	Primary	Secondary
+-----+-----+-----+		
Proportional	FREE	FREE
Independent	FREE	FREE_BACKUP
+-----+-----+-----+		

Figure 2: FREE\* State Transitions

In case of servers operating in active-passive mode, while a majority of the resources are owned by the primary server, the secondary server will need a portion of the resources to serve new clients while operating in COMMUNICATION-INTERRUPTED state and also in PARTNER-DOWN state before it can take over the entire address pool (after the expiry of MCLT).

The secondary server cannot simply take over the entire resource pool immediately, since it could also be that both servers are able to communicate with DHCP clients, but unable to communicate with each other.

The size of the resource pool allocated to the secondary is specified as a percentage of the currently available resources. Thus, as the number of available resources changes on the primary server, the number of resources available to the secondary server MUST also change, although the frequency of the changes made to the secondary server's pool of address resources SHOULD be low enough to not use significant processing power or network bandwidth.

The required size of this private pool allocated to the secondary server is based only on the arrival rate of new DHCP clients and the length of expected downtime of the primary server, and is not directly influenced by the total number of DHCP clients supported by the server pair.

## 8. Failover Mechanisms

This section lays out an overview of the communication between partners and other mechanisms required for failover operation. As this is a design document, not a protocol specification, high level ideas are presented without implementation specific details (e.g. on-wire protocol formats).

### 8.1. Time Skew

Partners exchange information about known lease states. To reliably compare a known lease state with an update received from a partner, servers must be able to reliably compare the times stored in the known lease state with the times received in the update. Although a simple approach would be to require both partners to use synchronized time, e.g. by using NTP, such a service may not always be available in some scenarios that failover expects to cover. Therefore a mechanism to measure and track relative time differences between servers is necessary. To do so, each message **MUST** contain information about the time of the transmission in the time context of the transmitter. The transmitting server **MUST** set this as close to the actual transmission as possible. Transmission here is when data is added to the send queue of the socket (or the equivalent), as the application may not know about the time of the actual transmission of the "wire". The receiving partner **MUST** store its own timestamp of reception as close to the actual reception as possible. The received timestamp information is then compared with local timestamp.

To account for packet delay variation (jitter), the measured difference is not used directly, but rather the moving average of last `TIME_SKEW_PKTS_AVG` packets time difference is calculated. This averaged value is referred to as the time skew. Note that the time skew algorithm allows cooperation between servers with completely desynchronized clocks as well as those whose desynchronization itself is not constant.

### 8.2. Lazy updates

Lazy update refers to the requirement placed on a server implementing a failover protocol to update its failover partner whenever the binding database changes. A failover protocol which didn't support lazy update would require the failover partner update to complete before a DHCPv6 server could respond to a DHCPv6 client request. Such approach is often referred to as 'lockstep' and is the opposite of lazy updates. The lazy update mechanism allows a server to allocate a new or extend an existing lease and then update its failover partner as time permits.

Although the lazy update mechanism does not introduce additional delays in server response times, it introduces other difficulties. The key problem with lazy update is that when a server fails after updating a client with a particular lease time and before updating its partner, the partner will believe that a lease has expired even though the client still retains a valid lease on that address or prefix. It is also possible that the partner will have no record at all of the lease of the resource to the client.

### 8.3. MCLT concept

In order to handle problem introduced by lazy updates (see Section 8.2), a period of time known as the "Maximum Client Lead Time" (MCLT) is defined and must be known to both the primary and secondary servers. Proper use of this time interval places an upper bound on the difference allowed between the lease time provided to a DHCPv6 client by a server and the lease time known by that server's failover partner.

The MCLT is typically much less than the lease time that a server has been configured to offer a client, and so some strategy must exist to allow a server to offer the configured lease time to a client. During a lazy update the updating server typically updates its partner with a potential expiration time which is longer than the lease time previously given to the client and which is longer than the lease time that the server has been configured to give a client. This allows that server to give a longer lease time to the client the next time the client renews its lease, since the time that it will give to the client will not exceed the MCLT beyond the potential expiration time acknowledged by its partner.

The fundamental relationship on which much of the correctness of this protocol depends is that the lease expiration time known to a DHCPv6 client MUST NOT be greater by more than the MCLT beyond the potential expiration time known to that server's failover partner.

The remainder of this section makes the above fundamental relationship more explicit.

This protocol requires a DHCPv6 server to deal with several different lease intervals and places specific restrictions on their relationships. The purpose of these restrictions is to allow the other server in the pair to be able to make certain assumptions in the absence of an ability to communicate between servers.

The different times are:

desired valid lifetime:

The desired valid lifetime is the lease interval that a DHCPv6 server would like to give to a DHCPv6 client in the absence of any restrictions imposed by the failover protocol. Its determination is outside of the scope of this protocol. Typically this is the result of external configuration of a DHCPv6 server.

actual valid lifetime:

The actual valid lifetime is the lease interval that a DHCPv6 server gives out to a DHCPv6 client. It may be shorter than the desired valid lifetime (as explained below).

potential valid lifetime:

The potential valid lifetime is the potential lease expiration interval the local server tells to its partner in a BNDUPD message.

acknowledged potential valid lifetime:

The acknowledged potential valid lifetime is the potential lease interval the partner server has most recently acknowledged in a BNDACK message.

#### 8.3.1. MCLT example

The following example demonstrates the MCLT concept in practice. The values used are arbitrarily chosen and not a recommendation for actual values. The MCLT in this case is 1 hour. The desired valid lifetime is 3 days, and its renewal time is half the valid lifetime.

When a server makes an offer for a new lease on an IP address to a DHCPv6 client, it determines the desired valid lifetime (in this case, 3 days). It then examines the acknowledged potential valid lifetime (which in this case is zero) and determines the remainder of the time left to run, which is also zero. It adds the MCLT to this value. Since the actual valid lifetime cannot be allowed to exceed the remainder of the current acknowledged potential valid lifetime plus the MCLT, the offer made to the client is for the remainder of the current acknowledged potential valid lifetime (i.e. zero) plus the MCLT. Thus, the actual valid lifetime is 1 hour (the MCLT).

Once the server has sent the REPLY to the DHCPv6 client, it will update its failover partner with the lease information. However, the desired potential valid lifetime will be composed of one half of the current actual valid lifetime added to the desired valid lifetime. Thus, the failover partner is updated with a BNDUPD with a potential valid lifetime of 1/2 hour + 3 days.

When the primary server receives a BNDACK to its update of the secondary server's (partner's) potential valid lifetime, it records

that as the acknowledged potential valid lifetime. A server MUST NOT send a BNDACK in response to a BNDUPD message until it is sure that the information in the BNDUPD message has been updated in its lease database. See Section 8.9. Thus, the primary server in this case can be sure that the secondary server has recorded the potential lease interval in its stable storage when the primary server receives a BNDACK message from the secondary server.

When the DHCPv6 client attempts to renew at T1 (approximately one half an hour from the start of the lease), the primary server again determines the desired valid lifetime, which is still 3 days. It then compares this with the original acknowledged potential valid lifetime (1/2 hour + 3 days) and adjusts for the time passed since the secondary was last updated (1/2 hour). Thus the time remaining of the acknowledged potential valid interval is 3 days. Adding the MCLT to this yields 3 days plus 1 hour, which is more than the desired valid lifetime of 3 days. So the client is renewed for the desired valid lifetime -- 3 days.

When the primary DHCPv6 server updates the secondary DHCPv6 server after the DHCPv6 client's renewal REPLY is complete, it will calculate the desired potential valid lifetime as the T1 fraction of the actual client valid lifetime (1/2 of 3 days this time = 1.5 days). To this it will add the desired client valid lifetime of 3 days, yielding a total desired potential valid lifetime of 4.5 days. In this way, the primary attempts to have the secondary always "lead" the client in its understanding of the client's valid lifetime so as to be able to always offer the client the desired client valid lifetime.

Once the initial actual client valid lifetime of the MCLT is past, the protocol operates effectively like the DHCPv6 protocol does today in its behavior concerning valid lifetimes. However, the guarantee that the actual client valid lifetime will never exceed the remaining acknowledged partner server potential valid lifetime by more than the MCLT allows full recovery from a variety of failures.

#### 8.4. Unreachability detection

Each partner MUST maintain a FO\_SEND timer for each failover connection. The FO\_SEND timer is reset every time any message is transmitted. If the timer reaches the FO\_SEND\_MAX value, a CONTACT message is transmitted and timer is reset. The CONTACT message may be transmitted at any time. An implementation MAY use additional mechanisms to detect partner unreachability.

Implementers are advised to keep in mind that the timer based CONTACT message mechanism is not perfect and may not detect some failures.



In particular, if the partner is using one interface to reach clients ("downlink") and another to reach its partner ("uplink"), it is possible that communication with the clients will break, yet the mechanism will still claim full reachability. For that reason it is beneficial to share the same interface for client traffic and communication with the failover partner. That approach may have drawbacks in some network topologies.

#### 8.5. Re-allocating Leases

When in PARTNER-DOWN state there is a waiting period after which a resource can be re-allocated to another client. For resources which are available when the server enters PARTNER-DOWN state, the period is the MCLT from the entry into PARTNER-DOWN state. For resources which are not available when the server enters PARTNER-DOWN state, the period is the MCLT after the later of the following times: the potential valid lifetime, the most recently transmitted potential valid lifetime, the most recently received acknowledged potential valid lifetime, and the most recently transmitted acknowledged potential valid lifetime. If this time would be earlier than the current time plus the MCLT, then the time the server entered PARTNER-DOWN state plus the maximum-client-lead-time is used.

In any other state, a server cannot reallocate a resource from one client to another without first notifying its partner (through a BNDUPD message) and receiving acknowledgement (through a BNDACK message) that its partner is aware that that first client is not using the resource.

This could be modeled in the following way. Though this specific implementation is in no way required, it may serve to better illustrate the concept.

An "available" resource on a server may be allocated to any client. A resource which was leased to a client and which expired or was released by that client would take on a new state, EXPIRED or RELEASED respectively. The partner server would then be notified that this resource was EXPIRED or RELEASED through a BNDUPD. When the sending server received the BNDACK for that resource showing it was FREE, it would move the resource from EXPIRED or RELEASED to FREE, and it would be available for allocation by the primary server to any clients.

A server MAY reallocate a resource in the EXPIRED or RELEASED state to the same client with no restrictions provided it has not sent a BNDUPD message to its partner. This situation would exist if the lease expired or was released after the transition into PARTNER-DOWN state, for instance.

## 8.6. Sending Binding Update

This and the following section is written as though every BNDUPD message contains only a single binding update transaction in order to reduce the complexity of the discussion. Servers MAY generate messages with multiple binding update transactions in them, and their partner servers MAY process these messages. Before multiple binding update transactions are to be sent and processed over a failover connection, their use MUST be negotiated during the CONNECT and CONNECTACK connection establishment processing.

Each server updates its failover partner about recent changes in lease states. Each update MUST include at least the following information:

1. resource type - non-temporary address or a prefix. Resource type can be indicated by the container that conveys the actual resource (e.g. an IA\_NA option indicates non-temporary IPv6 address);
2. resource information - the actual address or prefix. That is conveyed using the appropriate option, e.g. an IAADDR for an address or an IAPREFIX for a prefix;
3. valid life time sent to client\*;
4. IAID - Identity Association used by the client, while obtaining a given lease. (Note1: one client may use many IAIDs simultaneously. Note2: IAID for IA, TA and PD are orthogonal number spaces.)\*;
5. Next Expected Client Transmission (renewal time) - time interval since Client Last Transmission Time, when a response from a client is expected\*;
6. potential valid life time - a lifetime that the server is willing to set if there were no MCLT/failover restrictions imposed\*;
7. preferred life time sent to client - the actual value sent back to the client\*;
8. CLTT - Client Last Transaction Time, a timestamp of the last received transmission from a client\*;
9. Client DUID\*.
10. Resource state.

11. start time of state (especially for non-client updates).

Items marked with asterisk MUST appear only if the lease is/was associated with a client. Otherwise it MUST NOT appear.

The BNDUPD message MAY contain additional information related to the updated lease. The additional information MAY include, but is not limited to:

1. assigned FQDN name, defined in [RFC4704];
2. Options Requested by the client, i.e. content of the ORO;
3. Relay Data option from DHCPv6 Leasequery, see [RFC5007] Section 4.1.2.4
4. Any other options the updating partner deems useful.

The receiving partner MAY store any additional information received, but it MAY choose to ignore it as well. Some information may be useful, so it is a good idea to keep or update it. One reason is FQDN information. A server SHOULD be prepared to clean up DNS information once the lease expires or is released. See Section 11 for a detailed discussion about Dynamic DNS. Another reason the partner may be interested in keeping additional data is a better support for leasequery [RFC5007] or bulk leasequery [RFC5460], which features queries based on Relay-ID, by link address and by Remote-ID.

#### 8.7. Receiving Binding Update

When a server receives a BNDUPD message, it needs to decide how to process the binding update transaction it contains and whether that transaction represents a conflict of any sort. The conflict resolution process MUST be used on the receipt of every BNDUPD message, not just those that are received while in POTENTIAL-CONFLICT state, in order to increase the robustness of the protocol.

There are three sorts of conflicts:

1. Two clients, one resource - This is the duplicate resource allocation conflict. There two different clients each allocated the same resource. See Section 8.8.
2. Two resources, one client conflict - This conflict exists when a client on one server is associated with a one resource, and on the other server with a different resource in the same or related prefix. This does not refer to the case where a single client has resources in multiple different prefixes or administrative

domains (i.e. a mobile client that changed its location), but rather the case where on the same prefix the client has a lease on one IP address in one server and on a different IP address on the other server.

This conflict may or may not be a problem for a given DHCP server implementation and policy. If implementations and policies allow, both resources can be assigned to a given client. In the event that a DHCP server requires that a DHCP client have only one outstanding lease of a given type, the conflict **MUST** be resolved by accepting the lease which has the latest CLTT.

It should be further clarified that DHCPv6 protocol makes assignments based on a (client DUID, resource type, IAID) triplet. The possibility of using different IAIDs was omitted in this paragraph for clarity. If one client is assigned multiple resources of the same type, but with different IAIDs, there is no conflict. Also, IAID values for different resource types are orthogonal, i.e. an IA\_NA with IAID=1 is different than an IA\_PD with IAID=1 and there is no conflict.

3. binding-status conflict - This is normal conflict, where one server is updating the other with newer information. See Section 8.8 for details of how to resolve these conflicts.
4. configuration conflict -- This kind of conflict stems from a differing configuration on one server than on the other server. It may be transient (last until both servers can process a new configuration) or it may be chronic. It cannot be resolved by communications over the failover connection, but must be resolved (if it is not transient) by administrator action to resolve the conflicts.

#### 8.8. Conflict Resolution

The server receiving a lease update from its partner must evaluate the received lease information to see if it is consistent with already known state and decide which information - the previously known or that just received - is "better". The server should take into consideration the following aspects: if the lease is already assigned to a specific client, who had contact with client recently, start time of the lease, etc.

When analyzing a BNDUPD message from a partner server, if there is insufficient information in the BNDUPD to process it, then reject the BNDUPD with reject-reason "Missing binding information".

If the resource in the BNDUPD is not a resource associated with the failover endpoint which received the BNDUPD message, then reject it with reject-reason "Illegal IP address or prefix (not part of any address or prefix pool)".

Every BNDUPD message SHOULD contain a client-last-transaction-time option, which MUST, if it appears, be the time that the server last interacted with the DHCP client. It MUST NOT be, for instance, the time that the lease on an IP address expired. If there has been no interaction with the DHCP client in question (or there is no DHCP client presently associated with this resource), then there will be no client-last-transaction-time option in the BNDUPD message.

The list in Figure 3 presents the conflict resolution outcome. To "accept" a BNDUPD means to update the server's bindings database with the information contained in the BNDUPD and once the update is complete, send a BNDACK message corresponding to the BNDUPD message. To "reject" a BNDUPD means to leave the server's binding database unchanged and to respond to the BNDUPD with BNDACK with a reject-reason option included.

When interpreting the information in the following table (Figure 3), for those rules that are listed with "time" -- if a BNDUPD doesn't have a client-last-transaction-time value, then it MUST NOT be considered later than the client-last-transaction-time in the receiving server's binding. If the BNDUPD contains a client-last-transaction-time value and the receiving server's binding does not, then the client-last-transaction-time value in the BNDUPD MUST be considered later than the server's.

binding-status in received BNDUPD.					
binding-status in receiving server	ACTIVE	EXPIRED	RELEASED	FREE FREE_BACKUP	RESET ABANDONED
ACTIVE	accept(5)	time(2)	time(1)	time(2)	accept
EXPIRED	time(1)	accept	accept	accept	accept
RELEASED	time(1)	time(1)	accept	accept	accept
FREE/FREE_BACKUP	accept	accept	accept	accept	accept
RESET	time(3)	accept	accept	accept	accept
ABANDONED	reject(4)	reject(4)	reject(4)	reject(4)	accept

Figure 3: Conflict Resolution

time(1): If the client-last-transaction-time in the BNDUPD is later than the client-last-transaction-time in the receiving server's binding, accept it, else reject it.

time(2): If the current time is later than the receiving server's lease-expiration-time, accept it, else reject it.

time(3): If the client-last-transaction-time in the BNDUPD is later than the start-time-of-state in the receiving server's binding, accept it, else reject it.

(1,2,3): If rejecting, use reject reason "Outdated binding information".

(4): Use reject reason "Less critical binding information".

(5): If the clients in a BNDUPD message and in a receiving server's binding differ, then if the receiving server is a secondary accept it, else reject it with a reject reason of "Fatal conflict exists: address in use by other client".

The lease update may be accepted or rejected. Rejection SHOULD NOT change the flag in a lease that says that it should be transmitted to the failover partner. If this flag is set, then it should be transmitted, but if it is not already set, the rejection of a lease state update SHOULD NOT trigger an automatic update of the failover partner sending the rejected update. The potential for update storms is too great, and in the unusual case where the servers simply can't agree, that disagreement is better than an update storm.

## 8.9. Acknowledging Reception

Upon acceptance of a binding lease, the server MUST notify its partner that it updated its database. A server MUST NOT send the BNDACK before its database is updated. A BNDACK MUST contain at least the minimum set of information required to unambiguously identify the BNDUPD that triggered the BNDACK.

## 9. Endpoint States

### 9.1. State Machine Operation

Each server (or, more accurately, failover endpoint) can take on a variety of failover states. These states play a crucial role in determining the actions that a server will perform when processing a request from a DHCPv6 client as well as dealing with changing external conditions (e.g., loss of connection to a failover partner).

The failover state in which a server is running controls the following behaviors:

- o Responsiveness -- the server is either responsive to DHCPv6 client requests or it is not.
- o Allocation Pool -- which pool of addresses (or prefixes) can be used for advertisement on receipt of a SOLICIT or allocation on receipt of a REQUEST message.
- o MCLT -- ensure that valid lifetimes are not beyond what the partner has acked plus the MCLT (or not).

A server will transition from one failover state to another based on the specific values held by the following state variables:

- o Current failover state.
- o Communications status (OK or not OK).
- o Partner's failover state (if known).

Whenever any of the above state variables changes state, the state machine is invoked, which may then trigger a change in the current failover state. Thus, whenever the communications status changes, the state machine processing is invoked. This may or may not result in a change in the current failover state.

Whenever a server transitions to a new failover state, the new state MUST be communicated to its failover partner in a STATE message if the communications status is OK. In addition, whenever a server makes a transition into a new state, it MUST record the new state, its current understanding of its partner's state, and the time at which it entered the new state in stable storage.

The following state transition diagram gives a condensed view of the state machine. If there is a difference between the words describing a particular state and the diagram below, the words should be considered authoritative.

In the state transition diagram below, the "+" or "-" in the upper right corner of each state is a notation about whether communication is ongoing with the other server.

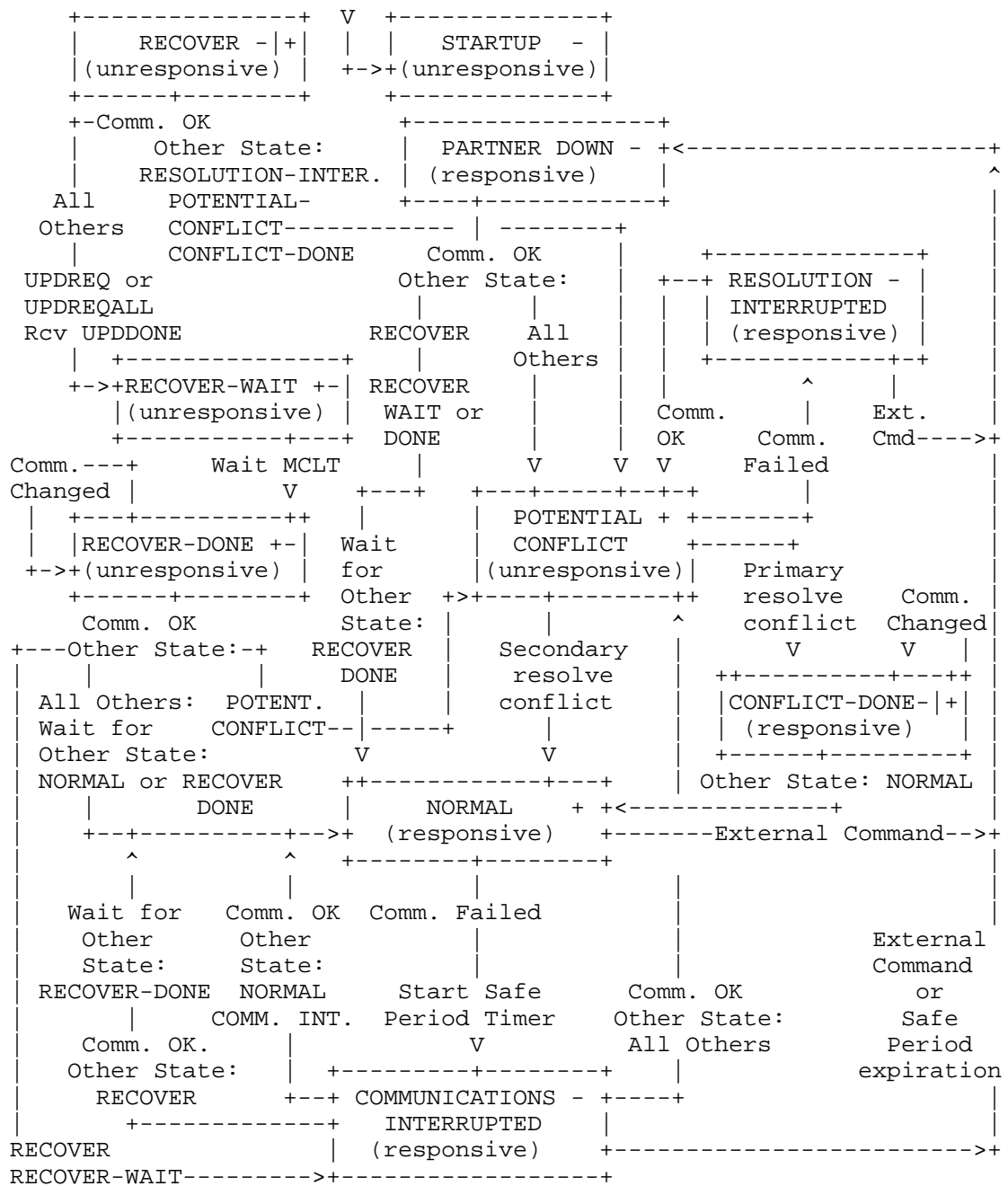


Figure 4: Failover Endpoint State Machine



## 9.2. State Machine Initialization

The state machine is characterized by storage (in stable storage) of at least the following information:

- o Current failover state.
- o Previous failover state.
- o Start time of current failover state.
- o Partner's failover state.
- o Start time of partner's failover state.
- o Time most recent packet received from partner.

The state machine is initialized by reading these data items from stable storage and restoring their values from the information saved. If there is no information in stable storage concerning these items, then they should be initialized as follows:

- o Current failover state: Primary: PARTNER-DOWN, Secondary: RECOVER
- o Previous failover state: None.
- o Start time of current failover state: Current time.
- o Partner's failover state: None until reception of STATE message.
- o Start time of partner's failover state: None until reception of STATE message.
- o Time most recent packet received from partner: None until packet received.

## 9.3. STARTUP State

The STARTUP state affords an opportunity for a server to probe its partner server, before starting to service DHCP clients. When in the STARTUP state, a server attempts to learn its partner's state and determine (using that information if it is available) what state it should enter.

The STARTUP state is not shown with any specific state transitions in the state machine diagram (Figure 4) because the processing during the STARTUP state can cause the server to transition to any of the other states, so that specific state transition arcs would only obscure other information.

#### 9.3.1. Operation in STARTUP State

The server **MUST NOT** be responsive to DHCPv6 clients in STARTUP state.

Whenever a STATE message is sent to the partner while in STARTUP state the STARTUP flag **MUST** be set in the message and the previously recorded failover state **MUST** be placed in the server-state option.

#### 9.3.2. Transition Out of STARTUP State

The following algorithm is followed every time the server initializes itself, and enters STARTUP state.

##### Step 1:

If there is any record in stable storage of a previous failover state for this server, set PREVIOUS-STATE to the last recorded value in stable storage, and go to Step 2.

If there is no record of any previous failover state in stable storage for this server, then set the PREVIOUS-STATE to RECOVER and set the TIME-OF-FAILURE to 0. This will allow two servers which already have lease information to synchronize themselves prior to operating.

In some cases, an existing server will be commissioned as a failover server and brought back into operation where its partner is not yet available. In this case, the newly commissioned failover server will not operate until its partner comes online -- but it has operational responsibilities as a DHCP server nonetheless. To properly handle this situation, a server **SHOULD** be configurable in such a way as to move directly into PARTNER-DOWN state after the startup period expires if it has been unable to contact its partner during the startup period.

##### Step 2:

Implementations will differ in the ways that they deal with the state machine for failover endpoint states. In many cases, state transitions will occur when communications goes from "OK" to failed, or from failed to "OK", and some implementations will implement a portion of their state machine processing based on these changes.

In these cases, during startup, if the previous state is one where communications was "OK", then set the previous state to the state that is the result of the communications failed state transition when in that state (if such transition exists -- some states don't have a communications failed state transition, since they allow both communications OK and failed).

Step 3:

Start the STARTUP state timer. The time that a server remains in the STARTUP state (absent any communications with its partner) is implementation dependent but SHOULD be short. It SHOULD be long enough for a TCP connection to be created to a heavily loaded partner across a slow network.

Step 4:

Attempt to create a TCP connection to the failover partner.

Step 5:

Wait for "communications OK".

When and if communications become "okay", clear the STARTUP flag, and set the current state to the PREVIOUS-STATE.

If the partner is in PARTNER-DOWN state, and if the time at which it entered PARTNER-DOWN state (as received in the start-time-of-state option in the STATE message) is later than the last recorded time of operation of this server, then set CURRENT-STATE to RECOVER. If the time at which it entered PARTNER-DOWN state is earlier than the last recorded time of operation of this server, then set CURRENT-STATE to POTENTIAL-CONFLICT.

Then, transition to the current state and take the "communications OK" state transition based on the current state of this server and the partner.

Step 6:

If the startup time expires the server SHOULD transition to the PREVIOUS-STATE.

#### 9.4. PARTNER-DOWN State

PARTNER-DOWN state is a state either server can enter. When in this state, the server assumes that it is the only server operating and serving the client base. If one server is in PARTNER-DOWN state, the other server MUST NOT be operating.

A server can enter PARTNER-DOWN state either as a result of operator intervention (when an operator determines that the server's partner is, indeed, down), or as a result of an optional auto-partner-down capability where PARTNER-DOWN state is entered automatically after a server has been in COMMUNICATIONS-INTERRUPTED state for a pre-determined period of time.

##### 9.4.1. Operation in PARTNER-DOWN State

The server MUST be responsive in PARTNER-DOWN state, regardless if it is primary or secondary.

It will allow renewal of all outstanding leases on resources. For those resources for which the server is using proportional allocation, it will allocate resources from its own pool, and after a fixed period of time (the MCLT interval) has elapsed from entry into PARTNER-DOWN state, it may allocate IP addresses from the set of all available pools. Server SHOULD fully deplete its own pool, before starting allocations from its downed partner's pool.

Any resource tagged as available for allocation by the other server (at entry to PARTNER-DOWN state) MUST NOT be allocated to a new client until the MCLT beyond the entry into PARTNER-DOWN state has elapsed.

A server in PARTNER-DOWN state MUST NOT allocate a resource to a DHCP client different from that to which it was allocated at the entrance to PARTNER-DOWN state until the MCLT beyond the maximum of the following times: client expiration time, most recently transmitted potential-expiration-time, most recently received ack of potential-expiration-time from the partner, and most recently acked potential-expiration-time to the partner. If this time would be earlier than the current time plus the maximum-client-lead-time, then the time the server entered PARTNER-DOWN state plus the maximum-client-lead-time is used.

The server is not restricted by the MCLT when offering lease times while in PARTNER-DOWN state.

In the unlikely case when there are two servers operating in a PARTNER-DOWN state, there is a chance of duplicate leases assigned.

This leads to a POTENTIAL-CONFLICT (unresponsive) state when they re-establish contact. The duplicate lease issue can be postponed to a large extent by the server granting new leases first from its own pool. Therefore the server operating in PARTNER-DOWN state MUST use its own pool first for new leases before assigning any leases from its downed partner pool.

#### 9.4.2. Transition Out of PARTNER-DOWN State

When a server in PARTNER-DOWN state succeeds in establishing a connection to its partner, its actions are conditional on the state and flags received in the STATE message from the other server as part of the process of establishing the connection.

If the STARTUP bit is set in the server-flags option of a received STATE message, a server in PARTNER-DOWN state MUST NOT take any state transitions based on reestablishing communications. Essentially, if a server is in PARTNER-DOWN state, it ignores all STATE messages from its partner that have the STARTUP bit set in the server-flags option of the STATE message.

If the STARTUP bit is not set in the server-flags option of a STATE message received from its partner, then a server in PARTNER-DOWN state takes the following actions based on the state of the partner as received in a STATE message (either immediately after establishing communications or at any time later when a new state is received)

- o If the partner is in: [ NORMAL, COMMUNICATIONS-INTERRUPTED, PARTNER-DOWN, POTENTIAL-CONFLICT, RESOLUTION-INTERRUPTED, or CONFLICT-DONE ] state, then transition to POTENTIAL-CONFLICT state
- o If the partner is in: [ RECOVER, RECOVER-WAIT ] state stay in PARTNER-DOWN state
- o If the partner is in: [ RECOVER-DONE ] state transition into NORMAL state

#### 9.5. RECOVER State

This state indicates that the server has no information in its stable storage or that it is re-integrating with a server in PARTNER-DOWN state after it has been down. A server in this state MUST attempt to refresh its stable storage from the other server.

##### 9.5.1. Operation in RECOVER State

The server MUST NOT be responsive in RECOVER state.

A server in RECOVER state will attempt to reestablish communications with the other server.

#### 9.5.2. Transition Out of RECOVER State

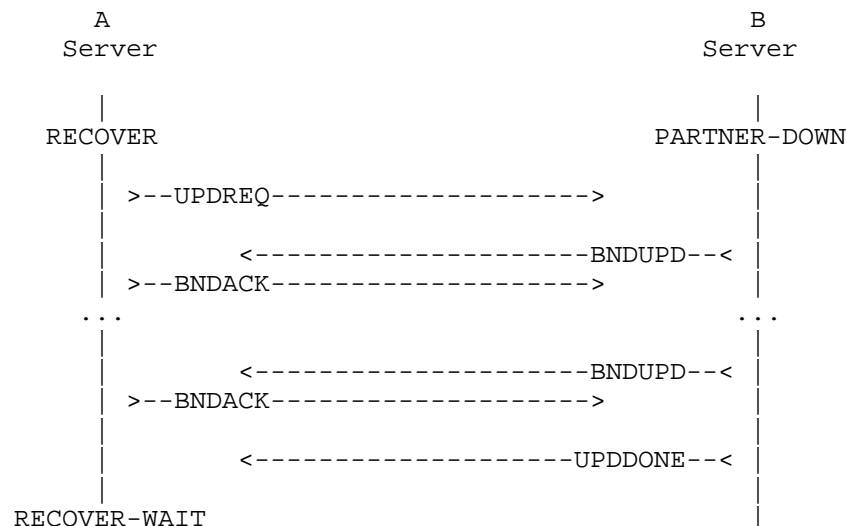
If the other server is in POTENTIAL-CONFLICT, RESOLUTION-INTERRUPTED, or CONFLICT-DONE state when communications are reestablished, then the server in RECOVER state will move to POTENTIAL-CONFLICT state itself.

If the other server is in any other state, then the server in RECOVER state will request an update of missing binding information by sending an UPDREQ message. If the server has determined that it has lost its stable storage because it has no record of ever having talked to its partner, while its partner does have a record of communicating with it, it MUST send an UPDREQALL message, otherwise it MUST send an UPDREQ message.

It will wait for an UPDDONE message, and upon receipt of that message it will transition to RECOVER-WAIT state.

If communications fails during the reception of the results of the UPDREQ or UPDREQALL message, the server will remain in RECOVER state, and will re-issue the UPDREQ or UPDREQALL when communications are re-established.

If an UPDDONE message isn't received within an implementation dependent amount of time, and no BNDUPD messages are being received, the connection SHOULD be dropped.



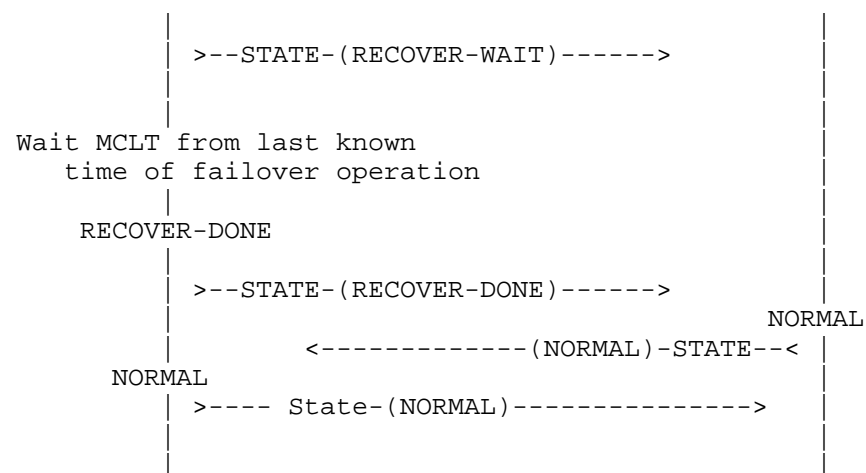


Figure 5: Transition out of RECOVER state

If at any time while a server is in RECOVER state communications fails, the server will stay in RECOVER state. When communications are restored, it will restart the process of transitioning out of RECOVER state.

#### 9.6. RECOVER-WAIT State

This state indicates that the server has sent an UPDREQ or UPDREQALL and has received the UPDDONE message indicating that it has received all outstanding binding update information. In the RECOVER-WAIT state the server will wait for the MCLT in order to ensure that any processing that this server might have done prior to losing its stable storage will not cause future difficulties.

##### 9.6.1. Operation in RECOVER-WAIT State

The server MUST NOT be responsive in RECOVER-WAIT state.

##### 9.6.2. Transition Out of RECOVER-WAIT State

Upon entry to RECOVER-WAIT state the server MUST start a timer whose expiration is set to a time equal to the time the server went down (if known) or the time the server started (if the down-time is unknown) plus the maximum-client-lead-time. When this timer expires, the server will transition into RECOVER-DONE state.

This is to allow any IP addresses that were allocated by this server prior to loss of its client binding information in stable storage to contact the other server or to time out.

If this is the first time this server has run failover -- as determined by the information received from the partner, not necessarily only as determined by this server's stable storage (as that may have been lost), then the waiting time discussed above may be skipped, and the server MAY transition immediately to RECOVER-DONE state.

If the server has never before run failover, then there is no need to wait in this state -- but, again, to determine if this server has run failover it is vital that the information provided by the partner be utilized, since the stable storage of this server may have been lost.

If communications fails while a server is in RECOVER-WAIT state, it has no effect on the operation of this state. The server SHOULD continue to operate its timer, and if the timer expires during the period where communications with the other server have failed, then the server SHOULD transition to RECOVER-DONE state. This is rare -- failover state transitions are not usually made while communications are interrupted, but in this case there is no reason to inhibit the timer.

#### 9.7. RECOVER-DONE State

This state exists to allow an interlocked transition for one server from RECOVER state and another server from PARTNER-DOWN or COMMUNICATIONS-INTERRUPTED state into NORMAL state.

##### 9.7.1. Operation in RECOVER-DONE State

A server in RECOVER-DONE state SHOULD be unresponsive, but MAY respond to RENEW requests but MUST only change the state of resources that appear in the RENEW request. It MUST NOT allocate any additional resources when in RECOVER-DONE state.

##### 9.7.2. Transition Out of RECOVER-DONE State

When a server in RECOVER-DONE state determines that its partner server has entered NORMAL or RECOVER-DONE state, then it will transition into NORMAL state.

If communication fails while in RECOVER-DONE state, a server will stay in RECOVER-DONE state.



## 9.8. NORMAL State

NORMAL state is the state used by a server when it is communicating with the other server, and any required resynchronization has been performed. While some bindings database synchronization is performed in NORMAL state, potential conflicts are resolved prior to entry into NORMAL state as is binding database data loss.

When entering NORMAL state, a server will send to the other server all currently unacknowledged binding updates as BNDUPD messages.

When the above process is complete, if the server entering NORMAL state is a secondary server, then it will request resources (addresses and/or prefixes) for allocation using the POOLREQ message.

### 9.8.1. Operation in NORMAL State

Primary server is responsive in NORMAL state. Secondary is unresponsive in NORMAL state.

When in NORMAL state a primary server will operate in the following manner:

#### Lease time calculations

As discussed in Section 8.3, the lease interval given to a DHCP client can never be more than the MCLT greater than the most recently received potential-expiration-time from the failover partner or the current time, whichever is later.

As long as a server adheres to this constraint, the specifics of the lease interval that it gives to a DHCP client or the value of the potential-expiration-time sent to its failover partner are implementation dependent.

#### Lazy update of partner server

After sending a REPLY that includes a lease update to a client, the server servicing a DHCP client request attempts to update its partner with the new binding information.

#### Reallocation of resources between clients

Whenever a client binding is released or expires, a BNDUPD message must be sent to the partner, setting the binding state to RELEASED or EXPIRED. However, until a BNDACK is received for this message, the resource cannot be allocated to another client. It cannot be allocated to the same client again if a BNDUPD was sent, otherwise it can. See Section 8.5 for details.

In NORMAL state, each server receives binding updates from its partner server in BNDUPD messages. It records these in its client binding database in stable storage and then sends a corresponding BNDACK message to its partner server.

#### 9.8.2. Transition Out of NORMAL State

If an external command is received by a server in NORMAL state informing it that its partner is down, then transition into PARTNER-DOWN state. Generally, this would be an unusual situation, where some external agency knew the partner server was down prior to the failover server discovering it on its own.

If a server in NORMAL state fails to receive acks to messages sent to its partner for an implementation dependent period of time, it MAY move into COMMUNICATIONS-INTERRUPTED state. This situation might occur if the partner server was capable of maintaining the TCP connection between the server and also capable of sending a CONTACT message periodically, but was (for some reason) incapable of processing BNDUPD messages.

If the communications is determined to not be "ok" (as defined in Section 8.4), then transition into COMMUNICATIONS-INTERRUPTED state.

If a server in NORMAL state receives any messages from its partner where the partner has changed state from that expected by the server in NORMAL state, then the server should transition into COMMUNICATIONS-INTERRUPTED state and take the appropriate state transition from there. For example, it would be expected for the partner to transition from POTENTIAL-CONFLICT into NORMAL state, but not for the partner to transition from NORMAL into POTENTIAL-CONFLICT state.

If a server in NORMAL state receives a DISCONNECT message from its partner, the server should transition into COMMUNICATIONS-INTERRUPTED state.

#### 9.9. COMMUNICATIONS-INTERRUPTED State

A server goes into COMMUNICATIONS-INTERRUPTED state whenever it is unable to communicate with its partner. Primary and secondary servers cycle automatically (without administrative intervention) between NORMAL and COMMUNICATIONS-INTERRUPTED state as the network connection between them fails and recovers, or as the partner server cycles between operational and non-operational. No duplicate resource allocation can occur while the servers cycle between these states.

When a server enters COMMUNICATIONS-INTERRUPTED state, if it has been configured to support an automatic transition out of COMMUNICATIONS-INTERRUPTED state and into PARTNER-DOWN state (i.e., a auto-partner-down has been configured), then a timer **MUST** be started for the length of the configured auto-partner-down period.

A server transitioning into the COMMUNICATIONS-INTERRUPTED state from the NORMAL state **SHOULD** raise some alarm condition to alert administrative staff to a potential problem in the DHCP subsystem.

#### 9.9.1. Operation in COMMUNICATIONS-INTERRUPTED State

In this state a server **MUST** respond to all DHCP client requests. When allocating new leases, each server allocates from its own pool, where the primary **MUST** allocate only FREE resources, and the secondary **MUST** allocate only FREE\_BACKUP resources. When responding to RENEW messages, each server will allow continued renewal of a DHCP client's current lease on a resource irrespective of whether that lease was given out by the receiving server or not, although the renewal period **MUST NOT** exceed the maximum client lead time (MCLT) beyond the latest of: 1) the potential valid lifetime already acknowledged by the other server, or 2) now, or 3) the potential valid lifetime received from the partner server.

However, since the server cannot communicate with its partner in this state, the acknowledged potential valid lifetime will not be updated in any new bindings. This is likely to eventually cause the actual valid lifetimes to converge to the MCLT (unless this is greater than the desired-client-lease-time).

The server should continue to try to establish a connection with its partner.

#### 9.9.2. Transition Out of COMMUNICATIONS-INTERRUPTED State

If the safe period timer expires while a server is in the COMMUNICATIONS-INTERRUPTED state, it will transition immediately into PARTNER-DOWN state.

If an external command is received by a server in COMMUNICATIONS-INTERRUPTED state informing it that its partner is down, it will transition immediately into PARTNER-DOWN state.

If communications is restored with the other server, then the server in COMMUNICATIONS-INTERRUPTED state will transition into another state based on the state of the partner:

- o NORMAL or COMMUNICATIONS-INTERRUPTED: Transition into the NORMAL state.
- o RECOVER: Stay in COMMUNICATIONS-INTERRUPTED state.
- o RECOVER-DONE: Transition into NORMAL state.
- o PARTNER-DOWN, POTENTIAL-CONFLICT, CONFLICT-DONE, or RESOLUTION-INTERRUPTED: Transition into POTENTIAL-CONFLICT state.

The following figure illustrates the transition from NORMAL to COMMUNICATIONS-INTERRUPTED state and then back to NORMAL state again.

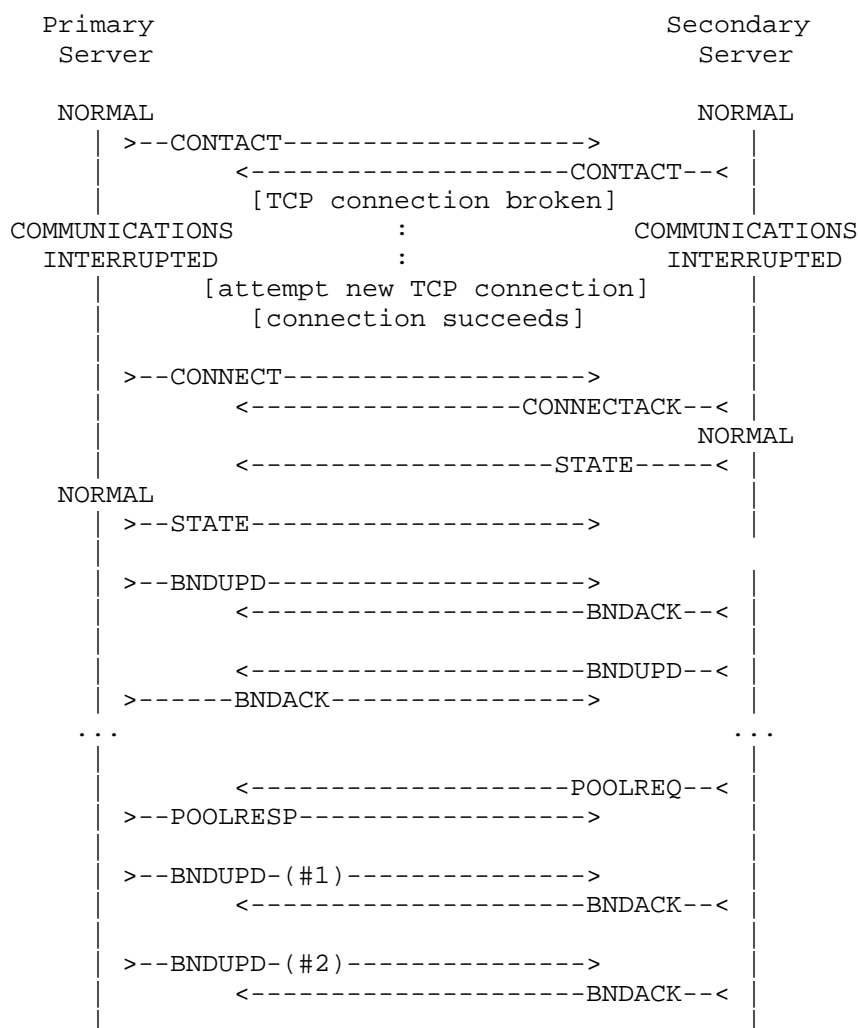


Figure 6: Transition from NORMAL to COMMUNICATIONS-INTERRUPTED and back (example with 2 addresses allocated to secondary)

#### 9.10. POTENTIAL-CONFLICT State

This state indicates that the two servers are attempting to reintegrate with each other, but at least one of them was running in a state that did not guarantee automatic reintegration would be possible. In POTENTIAL-CONFLICT state the servers may determine that the same resource has been offered and accepted by two different clients.

It is a goal of this protocol to minimize the possibility that POTENTIAL-CONFLICT state is ever entered.

When a primary server enters POTENTIAL-CONFLICT state it should request that the secondary send it all updates of which it is currently unaware by sending an UPDREQ message to the secondary server.

A secondary server entering POTENTIAL-CONFLICT state will wait for the primary to send it an UPDREQ message.

##### 9.10.1. Operation in POTENTIAL-CONFLICT State

Any server in POTENTIAL-CONFLICT state MUST NOT process any incoming DHCP requests.

##### 9.10.2. Transition Out of POTENTIAL-CONFLICT State

If communications fails with the partner while in POTENTIAL-CONFLICT state, then the server will transition to RESOLUTION-INTERRUPTED state.

Whenever either server receives an UPDDONE message from its partner while in POTENTIAL-CONFLICT state, it MUST transition to a new state. The primary MUST transition to CONFLICT-DONE state, and the secondary MUST transition to NORMAL state. This will cause the primary server to leave POTENTIAL-CONFLICT state prior to the secondary, since the primary sends an UPDREQ message and receives an UPDDONE before the secondary sends an UPDREQ message and receives its UPDDONE message.

When a secondary server receives an indication that the primary server has made a transition from POTENTIAL-CONFLICT to CONFLICT-DONE state, it SHOULD send an UPDREQ message to the primary server.

Primary  
Server

Secondary  
Server

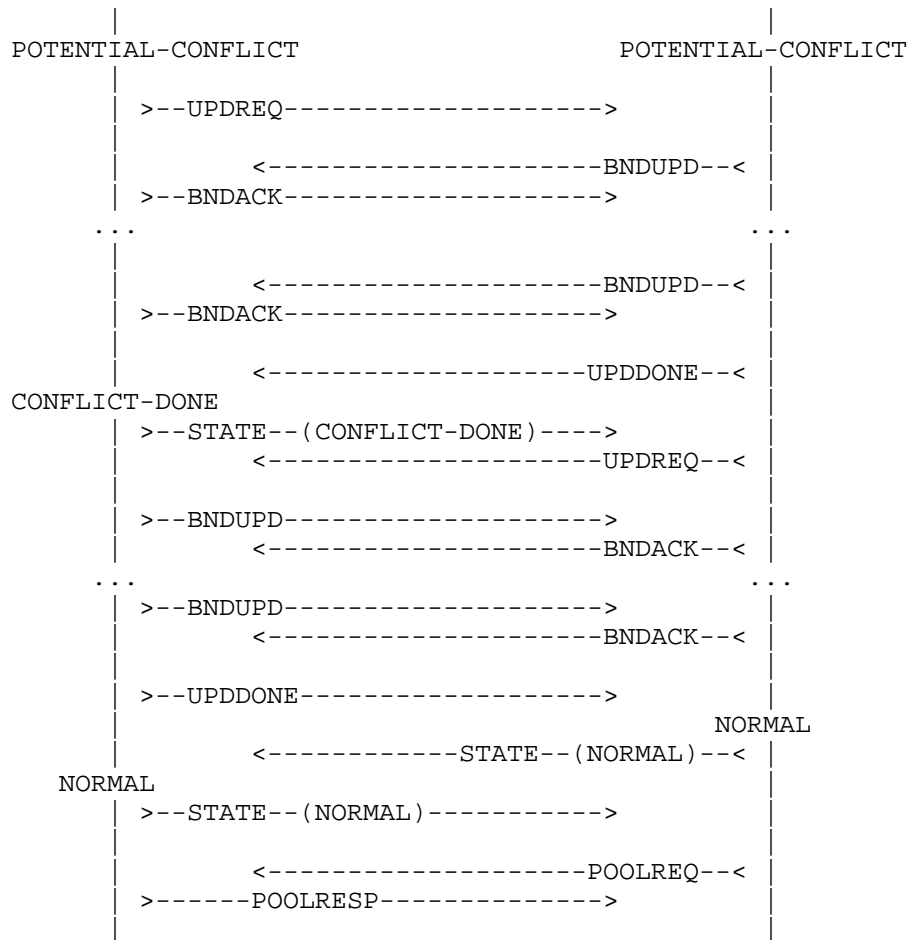


Figure 7: Transition out of POTENTIAL-CONFFLICT

#### 9.11. RESOLUTION-INTERRUPTED State

This state indicates that the two servers were attempting to reintegrate with each other in POTENTIAL-CONFFLICT state, but communications failed prior to completion of re-integration.

The RESOLUTION-INTERRUPTED state exists because servers are not responsive in POTENTIAL-CONFFLICT state, and if one server drops out of service while both servers are in POTENTIAL-CONFFLICT state, the server that remains in service will not be able to process DHCP client requests and there will be no DHCP service available. The RESOLUTION-INTERRUPTED state is the state that a server moves to if its partner disappears while it is in POTENTIAL-CONFFLICT state.

When a server enters RESOLUTION-INTERRUPTED state it SHOULD raise an alarm condition to alert administrative staff of a problem in the DHCP subsystem.

#### 9.11.1. Operation in RESOLUTION-INTERRUPTED State

In this state a server MUST respond to all DHCP client requests. When allocating new resources, each server SHOULD allocate from its own pool (if that can be determined), where the primary SHOULD allocate only FREE resources, and the secondary SHOULD allocate only FREE\_BACKUP resources. When responding to renewal requests, each server will allow continued renewal of a DHCP client's current lease independent of whether that lease was given out by the receiving server or not, although the renewal period MUST NOT exceed the maximum client lead time (MCLT) beyond the latest of: 1) the potential valid lifetime already acknowledged by the other server or 2) now or 3) potential valid lifetime received from the partner server.

However, since the server cannot communicate with its partner in this state, the acknowledged potential valid lifetime will not be updated in any new bindings.

#### 9.11.2. Transition Out of RESOLUTION-INTERRUPTED State

If an external command is received by a server in RESOLUTION-INTERRUPTED state informing it that its partner is down, it will transition immediately into PARTNER-DOWN state.

If communications is restored with the other server, then the server in RESOLUTION-INTERRUPTED state will transition into POTENTIAL-CONFLICT state.

#### 9.12. CONFLICT-DONE State

This state indicates that during the process where the two servers are attempting to re-integrate with each other, the primary server has received all of the updates from the secondary server. It makes a transition into CONFLICT-DONE state in order that it may be totally responsive to the client load. There is no operational difference between CONFLICT-DONE and NORMAL for primary as in both states it responds to all clients' requests. The distinction between CONFLICT-DONE and NORMAL states will be more apparent when load balancing extension will be defined.

##### 9.12.1. Operation in CONFLICT-DONE State

A primary server in CONFLICT-DONE state is fully responsive to all DHCP clients (similar to the situation in COMMUNICATIONS-INTERRUPTED state).

If communications fails, remain in CONFLICT-DONE state. If communications becomes OK, remain in CONFLICT-DONE state until the conditions for transition out become satisfied.

#### 9.12.2. Transition Out of CONFLICT-DONE State

If communications fails with the partner while in CONFLICT-DONE state, then the server will remain in CONFLICT-DONE state.

When a primary server determines that the secondary server has made a transition into NORMAL state, the primary server will also transition into NORMAL state.

### 10. Proposed extensions

The following section discusses possible extensions to the proposed failover mechanism. Listed extensions must be sufficiently simple to not further complicate failover protocol. Any proposals that are considered complex will be defined as stand-alone extensions in separate documents.

#### 10.1. Active-active mode

A very simple way to achieve active-active mode is to remove the restriction that secondary server MUST NOT respond to SOLICIT and REQUEST messages. Instead it could respond, but MUST have lower preference than primary server. Clients discovering available servers will receive ADVERTISE messages from both servers, but are expected to select the primary server as it has higher preference value configured. The following REQUEST message will be directed to primary server.

The benefit of this approach, compared to the "basic" active--passive solution is that there is no delay between primary failure and the moment when secondary starts serving requests.

### 11. Dynamic DNS Considerations

DHCP servers (and clients) can use DNS Dynamic Updates as described in RFC 2136 [RFC2136] to maintain DNS name-mappings as they maintain DHCP leases. Many different administrative models for DHCP-DNS integration are possible. Descriptions of several of these models, and guidelines that DHCP servers and clients should follow in carrying them out, are laid out in RFC 4704 [RFC4704].



The nature of the failover protocol introduces some issues concerning dynamic DNS (DDNS) updates that are not part of non-failover environments. This section describes these issues, and defines the information which failover partners should exchange in order to ensure consistent behavior. The presence of this section should not be interpreted as requiring an implementation of the DHCPv6 failover protocol to also support DDNS updates.

The purpose of this discussion is to clarify the areas where the failover and DHCP-DDNS protocols intersect for the benefit of implementations which support both protocols, not to introduce a new requirement into the DHCPv6 failover protocol. Thus, a DHCPv6 server which implements the failover protocol MAY also support dynamic DNS updates, but if it does support dynamic DNS updates it SHOULD utilize the techniques described here in order to correctly distribute them between the failover partners. See RFC 4704 [RFC4704] as well as RFC 4703 [RFC4703] for information on how DHCPv6 servers deal with potential conflicts when updating DNS even without failover.

From the standpoint of the failover protocol, there is no reason why a server which is utilizing the DDNS protocol to update a DNS server should not be a partner with a server which is not utilizing the DDNS protocol to update a DNS server. However, a server which is not able to support DDNS or is not configured to support DDNS SHOULD output a warning message when it receives BNDUPD messages which indicate that its failover partner is configured to support the DDNS protocol to update a DNS server. An implementation MAY consider this an error and refuse to operate, or it MAY choose to operate anyway, having warned the administrator of the problem in some way.

#### 11.1. Relationship between failover and dynamic DNS update

The failover protocol describes the conditions under which each failover server may renew a lease to its current DHCP client, and describes the conditions under which it may grant a lease to a new DHCP client. An analogous set of conditions determines when a failover server should initiate a DDNS update, and when it should attempt to remove records from the DNS. The failover protocol's conditions are based on the desired external behavior: avoiding duplicate address and prefix assignments; allowing clients to continue using leases which they obtained from one failover partner even if they can only communicate with the other partner; allowing the secondary DHCP server to grant new leases even if it is unable to communicate with the primary server. The desired external DDNS behavior for DHCP failover servers is similar to that described above for the failover protocol itself:

1. Allow timely DDNS updates from the server which grants a lease to a client. Recognize that there is often a DDNS update lifecycle which parallels the DHCP lease lifecycle. This is likely to include the addition of records when the lease is granted, and the removal of DNS records when the leased resource is subsequently made available for allocation to a different client.
2. Communicate enough information between the two failover servers to allow one to complete the DDNS update 'lifecycle' even if the other server originally granted the lease.
3. Avoid redundant or overlapping DDNS updates, where both failover servers are attempting to perform DDNS updates for the same lease-client binding.
4. Avoid situations where one partner is attempting to add RRs related to a lease binding while the other partner is attempting to remove RRs related to the same lease binding.

While DHCP servers configured for DDNS typically perform these operations on both the AAAA and the PTR resource records, this is not required. It is entirely possible that a DHCP server could be configured to only update the DNS with PTR records, and the DHCPv6 clients could be responsible for updating the DNS with their own AAAA records. In this case, the discussions here would apply only to the PTR records.

#### 11.2. Exchanging DDNS Information

In order for either server to be able to complete a DDNS update, or to remove DNS records which were added by its partner, both servers need to know the FQDN associated with the lease-client binding. In addition, to properly handle DDNS updates, additional information is required. All of the following information needs to be transmitted between the failover partners:

1. The FQDN that the client requested be associated with the resource. If the client doesn't request a particular FQDN and one is synthesized by the failover server or if the failover server is configured to replace a client requested FQDN with a different FQDN, then the server generated value would be used.
2. The FQDN that was actually placed in the DNS for this lease. It may differ from the client requested FQDN due to some form of disambiguation or other DHCP server configuration (as described above).
3. The status of and DDNS operations in progress or completed.

4. Information sufficient to allow the failover partner to remove the FQDN from the DNS should that become necessary.

These data items are the minimum necessary set to reliably allow two failover partners to successfully share the responsibility to keep the DNS up to date with the resources allocated to clients.

This information would typically be included in BNDUPD messages sent from one failover partner to the other. Failover servers MAY choose not to include this information in BNDUPD messages if there has been no change in the status of any DDNS update related to the lease.

The partner server receiving BNDUPD messages containing the DDNS information SHOULD compare the status information and the FQDN with the current DDNS information it has associated with the lease binding, and update its notion of the DDNS status accordingly.

Some implementations will instead choose to send a BNDUPD without waiting for the DDNS update to complete, and then will send a second BNDUPD once the DDNS update is complete. Other implementations will delay sending the partner a BNDUPD until the DDNS update has been acknowledged by the DNS server, or until some time-limit has elapsed, in order to avoid sending a second BNDUPD.

The FQDN option contains the FQDN that will be associated with the AAAA RR (if the server is performing an AAAA RR update for the client). The PTR RR can be generated automatically from the IP address or prefix value. The FQDN may be composed in any of several ways, depending on server configuration and the information provided by the client in its DHCP messages. The client may supply a hostname which it would like the server to use in forming the FQDN, or it may supply the entire FQDN. The server may be configured to attempt to use the information the client supplies, it may be configured with an FQDN to use for the client, or it may be configured to synthesize an FQDN.

Since the server interacting with the client may not have completed the DDNS update at the time it sends the first BNDUPD about the lease binding, there may be cases where the FQDN in later BNDUPD messages does not match the FQDN included in earlier messages. For example, the responsive server may be configured to handle situations where two or more DHCP client FQDNs are identical by modifying the most-specific label in the FQDNs of some of the clients in an attempt to generate unique FQDNs for them (a process sometimes called "disambiguation"). Alternatively, at sites which use some or all of the information which clients supply to form the FQDN, it's possible that a client's configuration may be changed so that it begins to supply new data. The server interacting with the client may react by

removing the DNS records which it originally added for the client, and replacing them with records that refer to the client's new FQDN. In such cases, the server SHOULD include the actual FQDN that was used in subsequent DDNS options in any BNDUPD messages exchanged between the failover partners. This server SHOULD include relevant information in its BNDUPD messages. This information may be necessary in order to allow the non-responsive partner to detect client configuration changes that change the hostname or FQDN data which the client includes in its DHCP requests.

#### 11.3. Adding RRs to the DNS

A failover server which is going to perform DDNS updates SHOULD initiate the DDNS update when it grants a new lease to a client. The server which did not grant the lease SHOULD NOT initiate a DDNS update when it receives the BNDUPD after the lease has been granted. The failover protocol ensures that only one of the partners will grant a lease to any individual client, so it follows that this requirement will prevent both partners from initiating updates simultaneously. The server initiating the update SHOULD follow the protocol in RFC 4704 [RFC4704]. The server may be configured to perform a AAAA RR update on behalf of its clients, or not. Ordinarily, a failover server will not initiate DDNS updates when it renews leases. In two cases, however, a failover server MAY initiate a DDNS update when it renews a lease to its existing client:

1. When the lease was granted before the server was configured to perform DDNS updates, the server MAY be configured to perform updates when it next renews existing leases. The server which granted the lease is the server which should initiate the DDNS update.
2. If a server is in PARTNER-DOWN state, it can conclude that its partner is no longer attempting to perform an update for the existing client. If the remaining server has not recorded that an update for the binding has been successfully completed, the server MAY initiate a DDNS update. It MAY initiate this update immediately upon entry to PARTNER-DOWN state, it may perform this in the background, or it MAY initiate this update upon next hearing from the DHCP client.

#### 11.4. Deleting RRs from the DNS

The failover server which makes a resource FREE\* SHOULD initiate any DDNS deletes, if it has recorded that DNS records were added on behalf of the client.

A server not in PARTNER-DOWN state "makes a resource FREE" when it initiates a BNDUPD with a binding-status of FREE, FREE\_BACKUP, EXPIRED, or RELEASED. Its partner confirms this status by acking that BNDUPD, and upon receipt of the BNDACK the server has "made the resource FREE". Conversely, a server in PARTNER-DOWN state "makes a resource FREE" when it sets the binding-status to FREE, since in PARTNER-DOWN state no communications is required with the partner.

It is at this point that it should initiate the DDNS operations to delete RRs from the DDNS. Its partner SHOULD NOT initiate DDNS deletes for DNS records related to the lease binding as part of sending the BNDACK message. The partner MAY have issued BNDUPD messages with a binding-status of FREE, EXPIRED, or RELEASED previously, but the other server will have rejected these BNDUPD messages.

The failover protocol ensures that only one of the two partner servers will be able to make a resource FREE\*. The server making the resource FREE may be doing so while it is in NORMAL communication with its partner, or it may be in PARTNER-DOWN state. If a server is in PARTNER-DOWN state, it may be performing DDNS deletes for RRs which its partner added originally. This allows a single remaining partner server to assume responsibility for all of the DDNS activity which the two servers were undertaking.

Another implication of this approach is that no DDNS RR deletes will be performed while either server is in COMMUNICATIONS-INTERRUPTED state, since no resource are moved into the FREE\* state during that period.

#### 11.5. Name Assignment with No Update of DNS

In some cases, a DHCP server is configured to return a name to the DHCPv6 client but not enter that name into the DNS. This is typically a name that it has discovered or generated from information it has received from the client. In this case this name information SHOULD be communicated to the failover partner, if only to ensure that they will return the same name in the event the partner becomes the server to which the DHCPv6 client begins to interact.

#### 12. Reservations and failover

Some DHCP servers support a capability to offer specific preconfigured resources to DHCP clients. These are real DHCP clients, they do the entire DHCP protocol, but these servers always offer the client a specific pre-configured resource, and they offer that resource to no other clients. Such a capability has several names, but it is sometimes called a "reservation", in that the resource is reserved for a particular DHCP client.

In a situation where there are two DHCP servers serving the same prefix without using failover, the two DHCP server's need to have disjoint resource pools, but identical reservations for the DHCP clients.

In a failover context, both servers need to be configured with the proper reservations in an identical manner, but if we stop there problems can occur around the edge conditions where reservations are made for resource that has already been leased to a different client. Different servers handle this conflict in different ways, but the goal of the failover protocol is to allow correct operation with any server's approach to the normal processing of the DHCP protocol.

The general solution with regards to reservations is as follows. Whenever a reserved resource becomes FREE (i.e., when first configured or whenever a client frees it or it expires or is reset), the primary server MUST show that resource as FREE (and thus available for its own allocation) and it MUST send it to the secondary server in a BNDUPD with a flag set showing that it is reserved and with a status of FREE\_BACKUP.

Note that this implies that a reserved resource goes through the normal state changes from FREE to ACTIVE (and possibly back to FREE). The failover protocol supports this approach to reservations, i.e., where the resource undergoes the normal state changes of any resource, but it can only be offered to the client for which it is reserved.

From the above, it follows that a reservation solely on the secondary will not necessarily allow the secondary to offer that address to client to whom it is reserved. The reservation must also appear on the primary as well for the secondary to be able to offer the resource to the client to which it is reserved.

When the reservation on a resource is cancelled, if the resource is currently FREE and the server is the primary, or FREE\_BACKUP and the server is the secondary, the server MUST send a BNDUPD to the other server with the binding-status FREE and an indication that the resource is no longer reserved.

### 13. Security Considerations

DHCPv6 failover is an extension of a standard DHCPv6 protocol, so all security considerations from [RFC3315], Section 23 and [RFC3633], Section 15 related to the server apply.

As traffic exchange between clients and server is not encrypted, an attacker that penetrated the network and is able to intercept traffic, will not gain any additional information by also sniffing communication between partners.

An attacker that is able to impersonate one partner can efficiently perform a denial of service attack on the remaining uncompromised server. Several techniques may be used: pretending that conflict resolution is required, requesting rebalance, claiming that a valid lease was released or declined etc. For that reason the communication between servers SHOULD support failover connections over TLS, as explained in Section 5.1. Such secure connections SHOULD be optional and configurable by the administrator.

A server MUST NOT operate in PARTNER-DOWN if its partner is up. Network administrators are expected to switch the remaining active server to PARTNER-DOWN state only if they are sure that its partner server is indeed down. Failing to obey this requirement will result in both servers likely assigning duplicate leases to different clients. Implementers should take that into consideration if they decide to implement the auto-partner-down timer-based transition to PARTNER-DOWN state.

Running a network protected by DHCPv6 failover requires more resources than running without it. In particular some of the resources are allocated to the secondary server and they are not usable in a normal (i.e. non failures) operation immediately, though over time they will be rebalanced and end up on the server that needs them. While limiting this pool may be preferable from resource utilization perspective, it must be a reasonably large pool, so the secondary may take over once the primary becomes unavailable.

### 14. IANA Considerations

IANA is not requested to perform any actions at this time.

### 15. Acknowledgements

This document extensively uses concepts, definitions and other parts of [dhcpv4-failover] document. Authors would like to thank Shawn Rother, Greg Rabil, Bernie Volz and Marcin Siodelski for their significant involvement and contributions. Authors would like to

thank VithalPrasad Gaitonde, Krzysztof Gierlowski, Krzysztof Nowicki and Michal Hoeft for their insightful comments.

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

## 16. References

### 16.1. 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.
- [RFC4703] Stapp, M. and B. Volz, "Resolution of Fully Qualified Domain Name (FQDN) Conflicts among Dynamic Host Configuration Protocol (DHCP) Clients", RFC 4703, October 2006.
- [RFC4704] Volz, B., "The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option", RFC 4704, October 2006.
- [RFC5007] Brzozowski, J., Kinnear, K., Volz, B., and S. Zeng, "DHCPv6 Leasequery", RFC 5007, September 2007.

### 16.2. Informative References

- [I-D.ietf-dhc-dhcpv6-failover-requirements]  
Mrugalski, T. and K. Kinnear, "DHCPv6 Failover Requirements", draft-ietf-dhc-dhcpv6-failover-requirements-07 (work in progress), July 2013.
- [I-D.ietf-dhc-dhcpv6-load-balancing]  
Kostur, A., "DHC Load Balancing Algorithm for DHCPv6", draft-ietf-dhc-dhcpv6-load-balancing-00 (work in progress), December 2012.



[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound,  
"Dynamic Updates in the Domain Name System (DNS UPDATE)",  
RFC 2136, April 1997.

[RFC5460] Stapp, M., "DHCPv6 Bulk Leasequery", RFC 5460, February  
2009.

[dhcpv4-failover]  
Droms, R., Kinnear, K., Stapp, M., Volz, B., Gonczi, S.,  
Rabil, G., Dooley, M., and A. Kapur, "DHCP Failover  
Protocol", draft-ietf-dhc-failover-12 (work in progress),  
March 2003.

#### Authors' Addresses

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

Phone: +1 650 423 1345  
Email: tomasz.mrugalski@gmail.com

Kim Kinnear  
Cisco Systems, Inc.  
1414 Massachusetts Ave.  
Boxborough, Massachusetts 01719  
USA

Phone: +1 (978) 936-0000  
Email: kkinnear@cisco.com

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 WG  
Internet-Draft  
Intended status: Standards Track  
Expires: November 29, 2015

Y. Cui  
Q. Sun  
Tsinghua University  
I. Farrer  
Deutsche Telekom AG  
Y. Lee  
Comcast  
Q. Sun  
China Telecom  
M. Boucadair  
France Telecom  
May 28, 2015

Dynamic Allocation of Shared IPv4 Addresses  
draft-ietf-dhc-dynamic-shared-v4allocation-09

Abstract

This memo describes the dynamic allocation of shared IPv4 addresses to clients using DHCPv4. Address sharing allows a single IPv4 address to be allocated to multiple active clients simultaneously, each client being differentiated by a unique set of transport layer source port numbers. The necessary changes to existing DHCPv4 client and server behavior are described and a new DHCPv4 option for provisioning clients with shared IPv4 addresses is included.

Due to the nature of IP address sharing, some limitations to its applicability are necessary. This memo describes these limitations and recommends suitable architectures and technologies where address sharing may be utilized.

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 November 29, 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.

## Table of Contents

1. Introduction . . . . .	2
2. Applicability Statement . . . . .	3
3. Requirements Language . . . . .	3
4. Terminology . . . . .	3
5. Functional Overview . . . . .	4
6. Client-Server Interaction . . . . .	4
7. Client Behavior . . . . .	5
7.1. Restrictions to Client Usage of a Shared IPv4 Address . .	6
8. Server Behavior . . . . .	7
8.1. Leasing Shared and Non-Shared IPv4 Addresses from a Single DHCP 4o6 Server . . . . .	8
9. DHCPv4 Port Parameters Option . . . . .	8
10. Security Considerations . . . . .	9
10.1. Port Randomization . . . . .	10
11. IANA Considerations . . . . .	10
12. Acknowledgements . . . . .	11
13. References . . . . .	11
13.1. Normative References . . . . .	11
13.2. Informative References . . . . .	12
Authors' Addresses . . . . .	13

## 1. Introduction

The shortage of available public IPv4 addresses means that it is not always possible for operators to allocate a full IPv4 address to every connected device. This problem is particularly acute whilst an operator is migrating from their existing, native IPv4 network to a native IPv6 network with IPv4 provided as an overlay service. During this phase, public IPv4 addresses are needed to provide for both existing and transition networks.

Two main types of solutions have emerged to address the problem (see Appendix A of [RFC6269]):

1. Deploying Carrier Grade Network Address Translation devices (CGNAT, [RFC6888]).
2. Distributing the same public IPv4 address to multiple clients differentiated by non-overlapping layer 4 port sets.

This memo focuses on the second category of solutions.

[RFC7341] introduces a "DHCP 4o6 Server", which offers dynamic leasing for IPv4 addresses to clients as in DHCPv4 [RFC2131] but transported within a DHCPv6 message flow. This memo specifies a new DHCPv4 option: `OPTION_V4_PORTPARAMS`, and describes how it can be used for the dynamic leasing of shared IPv4 addresses.

Although DHCPv4 over DHCPv6 is used as the underlying DHCPv4 transport mechanism throughout this document, `OPTION_V4_PORTPARAMS` as a DHCPv4 option may also be used in other solutions, if required.

## 2. Applicability Statement

The solution allows multiple hosts to be simultaneously allocated the same IP address. As the IP address is no longer a unique identifier for a host, this extension is only suitable for specific architectures based on the Address plus Port model (A+P) [RFC6346]. Specifically, this document presents a solution that applies to [I-D.ietf-software-lw4over6] and certain configurations of [I-D.ietf-software-map] (e.g., EA-bit length set to 0).

The solution should only be used on point-to-point links, tunnels, and/or in environments where authentication at the link layer is performed before IP address assignment. It is not suitable for network access over shared media, including Ethernet, WLAN, cable, etc..

## 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 [RFC2119].

## 4. Terminology

This document makes use of the following terms:

Shared IPv4 address: An IPv4 address with a restricted layer 4 port set.

Port Set ID (PSID): Identifier for a range of ports assigned to a DHCP client.

## 5. Functional Overview

Functionally, the dynamic allocation of shared IPv4 addresses by the DHCP 4o6 Server is similar to the dynamic allocation process for 'full' IPv4 addresses described in [RFC2131]. The essential difference is that the DHCP 4o6 Server can allocate the same IPv4 address to more than one DHCP 4o6 client simultaneously, providing that each shared address allocation also includes a range of layer 4 source ports unique to that address (i.e., the combined tuple of IPv4 address and Port Set ID is to be unique for each active lease).

The DHCP 4o6 client implements OPTION\_V4\_PORTPARAMS (described below), which is a DHCPv4 option containing PSID (Port Set ID) information. The client includes this option within the Parameter Request List option [RFC2132] in its DHCPv4 DHCPDISCOVER and DHCPREQUEST messages, indicating its support for shared, dynamic address leasing to the DHCP 4o6 server.

OPTION\_V4\_PORTPARAMS is also implemented by the server to identify clients that support shared, dynamic address leasing. With this option, the server can dynamically allocate PSIDs to clients and maintain shared IPv4 address leases. The server then manages unique client leases based the IPv4 address and PSID tuple, instead of using only the IPv4 address.

In the event that a dynamic, shared addressing capable client receives more than one DHCP 4o6 offer, where a received offer does not contain OPTION\_V4\_PORTPARAMS (i.e., is an offer for a full IPv4 address), then the client SHOULD prefer the full IPv4 offer over the shared IPv4 address offer(s), unless specifically configured otherwise.

## 6. Client-Server Interaction

The following DHCPv4 message flow is transported within the DHCPv4-query and DHCPv4-response messages as in DHCPv4 over DHCPv6 [RFC7341].

1. When the client constructs the DHCPv4 DHCPDISCOVER message to be transported within the DHCPv4-query message, the DHCPDISCOVER message MUST include the client identifier option (constructed as per [RFC4361]) and the Parameter Request List (PRL) option with the code of OPTION\_V4\_PORTPARAMS. The client MAY insert an OPTION\_V4\_PORTPARAMS with preferred values in related fields as a suggestion to the DHCP 4o6 Server.

2. DHCP 4o6 Servers that receive the DHCPDISCOVER message and support shared IPv4 addresses respond with a DHCPOFFER message with the shared IPv4 address in the 'yiaddr' field and MUST add an OPTION\_V4\_PORTPARAMS option containing an available restricted port set. If the DHCPDISCOVER included an OPTION\_V4\_PORTPARAMS option containing a non-zero PSID-Len field, the DHCP 4o6 Server MAY allocate a port set of the requested size to the client (depending on policy). The DHCPOFFER message is then encapsulated in the DHCPv4-response message and sent to the client.
3. The client evaluates all received DHCPOFFER messages and selects one (e.g., based on the configuration parameters received, such as the size of the offered port set). The client then sends a DHCPREQUEST encapsulated in the DHCPv4-query message containing the corresponding OPTION\_V4\_PORTPARAMS received in the DHCPOFFER message.
4. The server identified in the DHCPREQUEST message 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 a lease in any DHCP message. The server MUST respond with a DHCPACK message containing OPTION\_V4\_PORTPARAMS for the requesting client.
5. On receipt of the DHCPACK message with the configuration parameters, the client MUST NOT perform an in-use probe 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 leased network address and OPTION\_V4\_PORTPARAMS (with the allocated PSID) to identify the lease to be released.

In the case that the client has stored the previously allocated address and restricted port set, the logic described in Section 3.2 of [RFC2131] MUST be followed on the condition that the client's source IPv6 address for DHCP 4o6 does not change. Note, this corresponds to the INIT-REBOOT state defined in [RFC2131]. The client MUST include the OPTION\_V4\_PORTPARAMS with the requested port set information in the message flow, which starts with a DHCPREQUEST message. If the client's DHCP 4o6 IPv6 source address is changed for any reason, the client MUST re-initiate the DHCP 4o6 shared-address provisioning process by sending a DHCPDISCOVER message.

#### 7. Client Behavior

A DHCP 4o6 client sending a DHCPDISCOVER message for a shared IPv4 address MUST include the OPTION\_V4\_PORTPARAMS option code in the Parameter Request List option. If a client has been successfully allocated an IPv4 address and PSID previously, the client's DHCPDISCOVER message MAY include the 'requested IP address' option

along with an `OPTION_V4_PORTPARAMS` to request that a specific IPv4 address and PSID be re-assigned. Alternatively, the client MAY omit the 'requested IP address' option, but include an `OPTION_V4_PORTPARAMS` with a non-zero value in only the PSID-Len field, as a hint to the server for the preferred size of the port set.

A client that requests `OPTION_V4_PORTPARAMS`, but receives `DHCP OFFER` and `DHCP ACK` messages without `OPTION_V4_PORTPARAMS` SHOULD proceed as defined in [RFC7341] and configure a full IPv4 address with no address sharing (see Section 8.1 for the server's behavior).

When receiving a `DHCP ACK` message containing `OPTION_V4_PORTPARAMS`, the client MUST use the received explicit PSID for configuring the interface for which the DHCP 4o6 request was made.

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

When the client renews or releases its DHCP lease, it MUST put the values of offset, PSID length and PSID into `OPTION_V4_PORTPARAMS`, and send it to the server within corresponding DHCPv4 messages that are conveyed through DHCPv4-query message.

In the event that the client's DHCP 4o6 IPv6 source address is changed for any reason, the client MUST re-initiate the DHCP 4o6 shared-address provisioning process by sending a `DHCP DISCOVER` message.

#### 7.1. Restrictions to Client Usage of a Shared IPv4 Address

As a single IPv4 address is being shared between a number of different clients, the allocated shared address is only suitable for certain uses. 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, or accepting inbound connections.

The client MUST apply the following rules for all traffic destined to or originating from the shared IPv4 address:

- o The client MUST use only port-aware protocols (e.g., TCP, UDP, DCCP etc.) or ICMP implementing [RFC5508].
- o All connections originating from the shared IPv4 address MUST use a source port taken from the allocated restricted port set.
- o The client MUST NOT accept inbound connections on ports outside of the allocated restricted port set.

In order to prevent addressing conflicts which could arise from the allocation of the same IPv4 address, the client MUST NOT use the received restricted IPv4 address to perform ARP operations.

The mechanism by which a client implements the above rules is out of the scope of this document.

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).

## 8. Server Behavior

The DHCP 4o6 Server MUST NOT reply with `OPTION_V4_PORTPARAMS` unless the client has explicitly listed the option code in the Parameter Request List (Option 55) [RFC2132].

The DHCP 4o6 Server SHOULD reply with `OPTION_V4_PORTPARAMS` if the client includes `OPTION_V4_PORTPARAMS` in its Parameter Request List. In order to achieve the dynamic management of shared IPv4 addresses, the server is required to implement an address and port-set pool that provides the same function as the address pool in a regular DHCP server. Also, the server uses the combination of address and PSID as the key for maintaining the state of a lease, and for searching for an available lease for assignment. The leasing database is required to include the IPv4 address, PSID and client identifier of the requesting client.

When a server receives a DHCPDISCOVER message with `OPTION_V4_PORTPARAMS` in the Parameter Request List option, the server determines an IPv4 address with a PSID for the requesting client. If an IPv4 address with a PSID is available, the server SHOULD follow the logic below to select which specific address and PSID to provision to the client. The logic is similar to that in Section 4.3.1 of [RFC2131].

- o The client's current address with the PSID as recorded in the client's current lease binding, ELSE
- o The client's previous address with PSID as recorded in the client's (expired or released) binding, if that address with PSID is in the server's pool of available addresses and PSIDs, and not already allocated, ELSE
- o The address requested in the 'Requested IP Address' option along with the PSID parameters requested in the `OPTION_V4_PORTPARAMS`, if that pair of address and PSID is valid and not already allocated, ELSE
- o A new address with a PSID allocated from the server's pool of available addresses and PSIDs.

Upon receipt of a DHCPRELEASE message with OPTION\_V4\_PORTPARAMS, the server searches for the lease using the address in the 'ciaddr' field and the PSID information in the OPTION\_V4\_PORTPARAMS, and marks the lease as unallocated if a record (matching that PSID) is maintained by the server for that client.

The port-set assignment MUST be coupled with the address assignment process. Therefore the server MUST assign the address and port set in the same DHCP message.

When defining the pools of IPv4 addresses and PSIDs which are available to lease to clients, the server MUST implement a mechanism to reserve some port ranges (e.g., 0-1023) from allocation to clients. The reservation policy SHOULD be configurable.

#### 8.1. Leasing Shared and Non-Shared IPv4 Addresses from a Single DHCP 4o6 Server

A single DHCP 4o6 server may serve clients that do not support OPTION\_V4\_PORTPARAMS as well as those that do. As the rules for the allocation of shared addresses differ from the rules for full IPv4 address assignment, the DHCP 4o6 server MUST implement a mechanism to ensure that clients not supporting OPTION\_V4\_PORTPARAMS do not receive shared addresses. For example, two separate IPv4 addressing pools could be used, one of which allocates IPv4 addresses and PSIDs only to clients that have requested them.

If the server is only configured with address pools for shared address allocation, it MUST discard requests that do not contain OPTION\_V4\_PORTPARAMS in the Parameter Request List option.

A server configured with non-shared address pools can be instructed to honor received requests that contain OPTION\_V4\_PORTPARAMS in the Parameter Request List option (that is ignore OPTION\_V4\_PORTPARAMS and serve the requesting clients with non-shared IPv4 addresses).

#### 9. DHCPv4 Port Parameters Option

The meaning of 'offset', 'PSID-len', and 'PSID' fields of the DHCPv4 Port Parameters Option is identical to that of 'offset', 'PSID-len', and 'PSID' fields of the S46 Port Parameters Option (Section 4.5 of [I-D.ietf-software-map-dhcp]). The use of the same encoding in both options is meant to ensure compatibility with existing port set implementations.

The format of OPTION\_V4\_PORTPARAMS is shown in Figure 1.



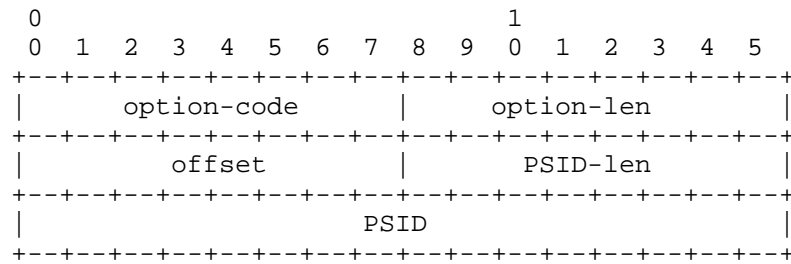


Figure 1: DHCPv4 Port Parameters Option

- o option-code: OPTION\_V4\_PORTPARAMS (TBA)
- o option-len: 4
- o offset: (PSID offset) 8 bits long field that specifies the numeric value for the excluded port range/offset bits (A-bits), as per section 5.1 of [I-D.ietf-softwire-map]. Allowed values are between 0 and 15, with the default value being 6 for MAP based implementations. 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 the value 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 client. 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.

In order to exclude the system ports ([RFC6335]) or ports reserved by ISPs, the former port-sets that contain well-known ports MUST NOT be assigned unless the operator has explicitly configured otherwise (e.g., by allocating a full IPv4 address).

## 10. Security Considerations

The security considerations described in [RFC2131] and [RFC7341] are also potentially applicable to this solution. Unauthorised DHCP 4o6 servers in the network could be used to stage an amplification attack or to supply invalid configuration leading to service disruption. The risks of these types of attacks can be reduced through the use of unicast DHCP 4o6 message flows (enabled by supplying DHCP 4o6 server unicast addresses within the OPTION\_DHCP4\_O\_DHCP6\_SERVER option).

A malicious user could attempt a DoS attack by requesting a large number of IPv4 address (or fractional address) and port sets allocations, exhausting the available addresses and port sets for other clients. This can be mitigated through DHCPv4 address allocation policy, limiting the number of simultaneously active IPv4 leases for clients whose request originate from each customer site.

The purpose of the client identifier option is to ensure that the same client retains the same parameters over time. This interferes with the client's privacy, as it allows the server to track the client. Clients can manage their privacy exposure by controlling the value of the client identifier, trading off stability of parameter allocation for privacy. We expect that guidance on this trade-off will be discussed in a future version of [I-D.ietf-dhc-anonymity-profile].

Additional security considerations are discussed in Section 11 of [I-D.ietf-softwire-map] and Section 9 of [I-D.ietf-softwire-lw4over6].

#### 10.1. Port Randomization

Preserving port randomization [RFC6056] may be more difficult because the host can only randomize the ports inside a fixed port range (see Section 13.4 of [RFC6269]).

More discussion to improve the robustness of TCP against Blind In-Window Attacks can be found at [RFC5961]. Other means than the (IPv4) source port randomization to provide protection against attacks should be used (e.g., use [RFC5961] to improve the robustness of TCP against Blind In-Window Attacks, use IPv6).

#### 11. IANA Considerations

IANA is requested to assign the following new DHCPv4 Option Code in the registry maintained in: <http://www.iana.org/assignments/bootp-dhcp-parameters/>:

Option Name	Value	Data length	Meaning
OPTION_V4_PORTPARAMS	TBA	4	This option is used to configure a set of ports bound to a shared IPv4 address.

## 12. Acknowledgements

This document is merged from [I-D.sun-dhc-port-set-option] and [I-D.farrer-dhc-shared-address-lease].

The authors would like to thank Peng Wu, Gabor Bajko, Teemu Savolainen, Ted Lemon, Tina Tsou, Pierre Levis, Cong Liu, Marcin Siodelski, and Christian Huitema for their contributions.

Many thanks to Brian Haberman for the review.

## 13. References

### 13.1. Normative 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-lw4over6-13 (work in progress), November 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-13 (work in progress), March 2015.
- [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.
- [RFC4361] Lemon, T. and B. Sommerfeld, "Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4)", RFC 4361, February 2006.
- [RFC5961] Ramaiah, A., Stewart, R., and M. Dalal, "Improving TCP's Robustness to Blind In-Window Attacks", RFC 5961, August 2010.
- [RFC6056] Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, January 2011.

- [RFC7341] Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4-over-DHCPv6 (DHCP 4o6) Transport", RFC 7341, August 2014.

### 13.2. Informative References

- [I-D.farrer-dhc-shared-address-lease]  
Farrer, I., "Dynamic Allocation of Shared IPv4 Addresses using DHCPv4 over DHCPv6", draft-farrer-dhc-shared-address-lease-00 (work in progress), June 2013.
- [I-D.ietf-dhc-anonymity-profile]  
Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity profile for DHCP clients", draft-ietf-dhc-anonymity-profile-00 (work in progress), May 2015.
- [I-D.ietf-softwire-map-dhcp]  
Mrugalski, T., Troan, O., Farrer, I., Perreault, S., Dec, W., Bao, C., Yeh, L., and X. Deng, "DHCPv6 Options for configuration of Softwire Address and Port Mapped Clients", draft-ietf-softwire-map-dhcp-12 (work in progress), March 2015.
- [I-D.sun-dhc-port-set-option]  
Qiong, Q., Lee, Y., Sun, Q., Bajko, G., and M. Boucadair, "Dynamic Host Configuration Protocol (DHCP) Option for Port Set Assignment", draft-sun-dhc-port-set-option-02 (work in progress), October 2013.
- [RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, May 2005.
- [RFC5508] Srisuresh, P., Ford, B., Sivakumar, S., and S. Guha, "NAT Behavioral Requirements for ICMP", BCP 148, RFC 5508, April 2009.
- [RFC6269] Ford, M., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, June 2011.
- [RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, August 2011.

- [RFC6346] Bush, R., "The Address plus Port (A+P) Approach to the IPv4 Address Shortage", RFC 6346, August 2011.
- [RFC6888] Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, April 2013.

## 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

Ian Farrer  
Deutsche Telekom AG  
CTO-ATI, Landgrabenweg 151  
Bonn, NRW 53227  
Germany

Email: ian.farrer@telekom.de

Yiu L. Lee  
Comcast  
One Comcast Center  
Philadelphia PA 19103  
USA

Email: yiu\_lee@cable.comcast.com

Qiong Sun  
China Telecom  
Room 708, No.118, Xizhimennei Street  
Beijing 100035  
P.R. China

Phone: +86-10-58552936  
Email: [sunqiong@ctbri.com.cn](mailto:sunqiong@ctbri.com.cn)

Mohamed Boucadair  
France Telecom  
Rennes 35000  
France

Email: [mohamed.boucadair@orange.com](mailto:mohamed.boucadair@orange.com)

DHC Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: August 25, 2017

L. Li  
Tsinghua University  
S. Jiang  
Huawei Technologies Co., Ltd  
Y. Cui  
Tsinghua University  
T. Jinmei  
Infoblox Inc.  
T. Lemon  
Nominum, Inc.  
D. Zhang  
February 21, 2017

Secure DHCPv6  
draft-ietf-dhc-sedhcpv6-21

Abstract

DHCPv6 includes no deployable security mechanism that can protect end-to-end communication between DHCP clients and servers. This document describes a mechanism for using public key cryptography to provide such security. The mechanism provides encryption in all cases, and can be used for authentication based on pre-sharing of authorized certificates.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 25, 2017.

Copyright Notice

Copyright (c) 2017 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. Requirements Language . . . . .	3
3. Terminology . . . . .	3
4. Security Issues of DHCPv6 . . . . .	4
5. Secure DHCPv6 Overview . . . . .	5
5.1. Solution Overview . . . . .	5
5.2. New Components . . . . .	6
5.3. Support for Algorithm Agility . . . . .	7
5.4. Impact on RFC3315 . . . . .	7
5.5. Applicability . . . . .	8
6. DHCPv6 Client Behavior . . . . .	8
7. DHCPv6 Server Behavior . . . . .	11
8. Relay Agent Behavior . . . . .	13
9. Processing Rules . . . . .	14
9.1. Increasing Number Check . . . . .	14
9.2. Encryption Key Tag Calculation . . . . .	14
10. Extensions for Secure DHCPv6 . . . . .	15
10.1. New DHCPv6 Options . . . . .	15
10.1.1. Algorithm Option . . . . .	15
10.1.2. Certificate Option . . . . .	17
10.1.3. Signature option . . . . .	18
10.1.4. Increasing-number Option . . . . .	20
10.1.5. Encryption-Key-Tag Option . . . . .	20
10.1.6. Encrypted-message Option . . . . .	21
10.2. New DHCPv6 Messages . . . . .	21
10.3. Status Codes . . . . .	22
11. Security Considerations . . . . .	22
12. IANA Considerations . . . . .	23
13. Acknowledgements . . . . .	25
14. Change log [RFC Editor: Please remove] . . . . .	25
15. References . . . . .	28
15.1. Normative References . . . . .	28
15.2. Informative References . . . . .	29
Authors' Addresses . . . . .	30



## 1. Introduction

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6, [RFC3315]) allows DHCPv6 servers to flexibly provide addressing and other configuration information relating to local network infrastructure to DHCP clients. The protocol provides no deployable security mechanism, and consequently is vulnerable to various attacks.

This document provides a brief summary of the security vulnerabilities of the DHCPv6 protocol and then describes a new extension to the protocol that provides two additional types of security:

- o authentication of the DHCPv6 client and the DHCPv6 server to defend against active attacks, such as spoofing.
- o encryption between the DHCPv6 client and the DHCPv6 server in order to protect the DHCPv6 communication from pervasive monitoring.

The extension specified in this document applies only to end-to-end communication between DHCP servers and clients. Options added by relay agents in Relay-Forward messages, and options other than the client message in Relay-Reply messages sent by DHCP servers, are not protected. Such communications are already protected using the mechanism described in [I-D.ietf-dhc-relay-server-security].

This extension introduces two new DHCPv6 messages: the Encrypted-Query and the Encrypted-Response messages. It defines six new DHCPv6 options: the Algorithm, Certificate, Signature, Increasing-number, Encryption-Key-Tag option and Encrypted-message options.

## 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] 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. Terminology

This section defines terminology specific to secure DHCPv6 used in this document.

secure DHCPv6 client: A node that initiates a DHCPv6 request on a link to obtain DHCPv6 configuration parameters from

one or more DHCPv6 servers using the encryption and optional authentication mechanisms defined in this document.

secure DHCPv6 server: A DHCPv6 server that implements the authentication and encryption mechanisms defined in this document, and is configured to use them.

#### 4. Security Issues of DHCPv6

[RFC3315] defines an authentication mechanism with integrity protection. This mechanism uses a symmetric key that is shared by the client and server for authentication. It does not provide any key distribution mechanism.

For this approach, operators can set up a key database for both servers and clients from which the client obtains a key before running DHCPv6. However, manual key distribution runs counter to the goal of minimizing the configuration data needed at each host. Consequently, there are no known deployments of this security mechanism.

[RFC3315] provides an additional mechanism for preventing off-network timing attacks using the Reconfigure message: the Reconfigure Key authentication method. However, this method protects only the Reconfigure message. The key is transmitted in plaintext to the client in earlier exchanges and so this method is vulnerable to on-path active attacks.

Anonymity Profile for DHCP Clients [RFC7844] explains how to generate DHCPv4 or DHCPv6 requests that minimize the disclosure of identifying information. However, the anonymity profile limits the use of the certain options. It also cannot anticipate new options that may contain private information. In addition, the anonymity profile does not work in cases where the client wants to maintain anonymity from eavesdroppers but must identify itself to the DHCP server with which it intends to communicate.

Privacy consideration for DHCPv6 [RFC7824] presents an analysis of the privacy issues associated with the use of DHCPv6 by Internet users. No solutions are presented.

Current DHCPv6 messages are still transmitted in cleartext and the privacy information within the DHCPv6 message is not protected from passive attack, such as pervasive monitoring [RFC7258]. The privacy information of the IPv6 host, such as DUID, may be gleaned to find location information, previous visited networks and so on. [RFC7258]

claims that pervasive monitoring should be mitigated in the design of IETF protocol, where possible.

To better address the problem of passive monitoring and to achieve authentication without requiring a symmetric key distribution solution for DHCP, this document defines an asymmetric key authentication and encryption mechanism. This protects against both active attacks, such as spoofing, and passive attacks, such as pervasive monitoring.

## 5. Secure DHCPv6 Overview

### 5.1. Solution Overview

The following figure illustrates the secure DHCPv6 procedure. Briefly, this extension establishes the server's identity with an anonymous Information-Request exchange. Once the server's identity has been established, the client may either choose to communicate with the server or not. Not communicating with an unknown server avoids revealing private information, but if there is no known server on a particular link, the client will be unable to communicate with a DHCP server.

If the client chooses to communicate with the selected server(s), it uses the Encrypted-Query message to encapsulate its communications to the DHCP server. The server responds with Encrypted-Response messages. Normal DHCP messages are encapsulated in these two new messages using the new defined Encrypted-message option. Besides the Encrypted-message option, the Signature option is defined to verify the integrity of the DHCPv6 messages and then authentication of the client and the server. The Increasing number option is defined to detect a replay attack.

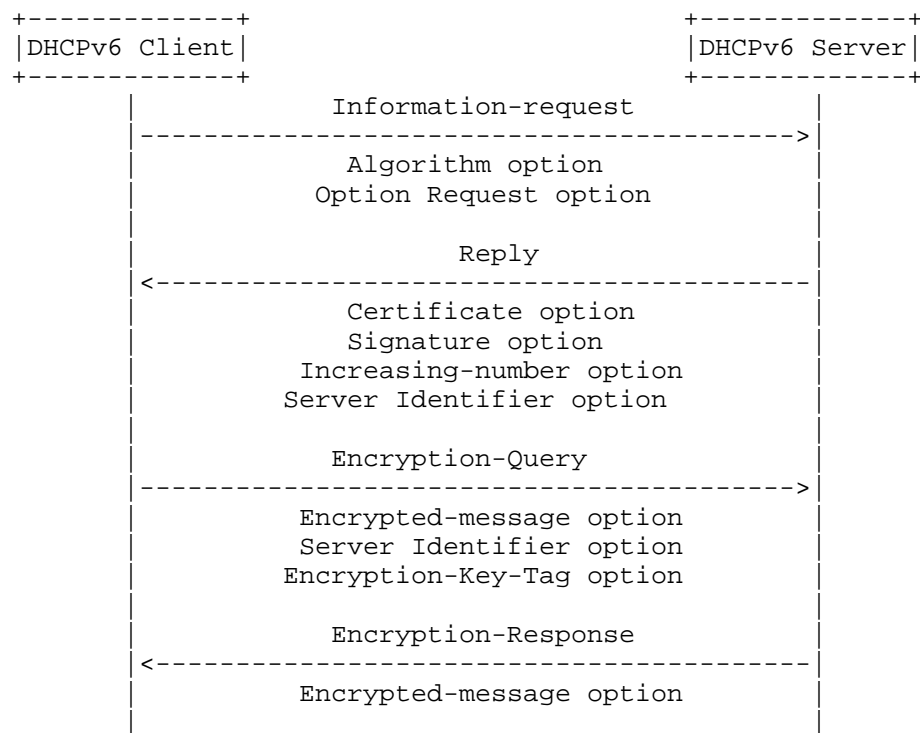


Figure 1: Secure DHCPv6 Procedure

## 5.2. New Components

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

- o Servers and clients that use certificates first generate a public/private key pair and then obtain a certificate that signs the public key. The Certificate option is defined to carry the certificate of the sender.
- o The algorithm option is defined to carry the algorithms lists for algorithm agility.
- o The signature is generated using the private key to verify the integrity of the DHCPv6 messages. The Signature option is defined to carry the signature.
- o The increasing number is used to detect replayed packet. The Increasing-number option is defined to carry a strictly-increasing serial number.

- o The encryption key Tag is calculated from the public key data. The Encryption-Key-Tag option is defined to identify the used public/private key pair.
- o The Encrypted-message option is defined to contain the encrypted DHCPv6 message.
- o The Encrypted-Query message is sent from the secure DHCPv6 client to the secure DHCPv6 server. The Encrypted-Query message MUST contain the Encrypted-message option and Encryption-Key-Tag option. In addition, the Server Identifier option MUST be included if it is contained in the original DHCPv6 message. The Encrypted-Query message MUST NOT contain any other options.
- o The Encrypted-Response message is sent from the secure DHCPv6 server to the secure DHCPv6 client. The Encrypted-Response message MUST contain the Encrypted-message option. The Encrypted-Response message MUST NOT contain any other options.

### 5.3. Support for Algorithm Agility

In order to provide a means of addressing problems that may emerge with existing hash algorithms, signature algorithm and encryption algorithms in the future, this document provides a mechanism to support algorithm agility. The support for algorithm agility in this document is mainly a algorithm notification mechanism between the client and the server. The same client and server MUST use the same algorithm in a single communication session. The client can offer a set of algorithms, and then the server selects one algorithm for the future communication.

### 5.4. Impact on RFC3315

For secure DHCPv6, the Solicit and Rebind messages can be sent only to the selected server(s) which share one common certificate. If the client doesn't like the received Advertise(s) it could restart the whole process and selects another certificate, but it will be more expensive, and there's no guarantee that other servers can provide better Advertise(s).

[RFC3315] provides an additional mechanism for preventing off-network timing attacks using the Reconfigure message: the Reconfigure Key authentication method. Secure DHCPv6 can protect the Reconfigure message using the encryption method. So the Reconfigure Key authentication method SHOULD NOT be used if Secure DHCPv6 is applied.

### 5.5. Applicability

In principle, secure DHCPv6 is applicable in any environment where physical security on the link is not assured and attacks on DHCPv6 are a concern. In practice, however, authenticated and encrypted DHCPv6 configuration will rely on some operational assumptions mainly regarding public key distribution and management. In order to achieve the wider use of secure DHCPv6, opportunistic security [RFC7435] can be applied to secure DHCPv6 deployment, which allows DHCPv6 encryption in environments where support for authentication or a key distribution mechanism is not available.

Secure DHCPv6 can achieve authentication and encryption based on pre-sharing of authorized certificates. One feasible environment in an early deployment stage would be enterprise networks. In enterprise networks, the client is manually pre-configured with the trusted servers' public key and the server can also be manually pre-configured with the trusted clients' public keys. In some scenario, such as coffee shop where the certificate cannot be validated and one wants access to the Internet, then the DHCPv6 configuration process can be encrypted without authentication.

Note that this deployment scenario based on manual operation is not much different from the existing, shared-secret based authentication mechanisms defined in [RFC3315] in terms of operational costs. However, Secure DHCPv6 is still securer than the shared-secret mechanism in that even if clients' keys stored for the server are stolen that does not mean an immediate threat as these are public keys. In addition, if some kind of Public Key Infrastructure (PKI) is used with Secure DHCPv6, even if the initial installation of the certificates is done manually, it will help reduce operational costs of revocation in case a private key (especially that of the server) is compromised.

### 6. DHCPv6 Client Behavior

The secure DHCPv6 client is pre-configured with a certificate and its corresponding private key for client authentication. If the client does not obtain a certificate from Certificate Authority (CA), it can generate the self-signed certificate.

The secure DHCPv6 client sends an Information-request message as per [RFC3315]. The Information-request message is used by the DHCPv6 client to request the server's certificate information without having addresses, prefixes or any non-security options assigned to it. The contained Option Request option MUST carry the option code of the Certificate option. In addition, the contained Algorithm option MUST be constructed as explained in Section 10.1.1. The Information-

request message MUST NOT include any other DHCPv6 options except the above options to minimize the client's privacy information leakage.

When receiving the Reply messages from the DHCPv6 servers, a secure DHCPv6 client discards any DHCPv6 message that meets any of the following conditions:

- o the Signature option is missing,
- o multiple Signature options are present,
- o the Certificate option is missing.

And then the client first checks acknowledged hash, signature and encryption algorithms that the server supports. The client checks the signature/encryption algorithms through the certificate option and checks the signature/hash algorithms through the signature option. The SA-id in the certificate option must be equal to the SA-id in the signature option. If they are different, then the client drops the Reply message. The client uses the acknowledged algorithms in the subsequent messages.

Then the client checks the authority of the server. In some scenario where non-authenticated encryption can be accepted, such as coffee shop, then authentication is optional and can be skipped. For the certificate check method, the client validates the certificates through the pre-configured local trusted certificates list or other methods. A certificate that finds a match in the local trust certificates list is treated as verified. If the certificate check fails, the Reply message is dropped.

The client MUST now authenticate the server by verifying the signature and checking increasing number, if there is a Increasing-number option. The order of two procedures is left as an implementation decision. It is RECOMMENDED to check increasing number first, because signature verification is much more computationally expensive. The client checks the Increasing-number option according to the rule defined in Section 9.1. For the message without an Increasing-number option, according to the client's local policy, it MAY be acceptable or rejected. The Signature field verification MUST show that the signature has been calculated as specified in Section 10.1.3. Only the messages that get through both the signature verification and increasing number check (if there is a Increasing-number option) are accepted. Reply message that does not pass the above tests MUST be discarded.

If there are multiple authenticated DHCPv6 certs, the client selects one DHCPv6 cert for the following communication. The selected

certificate may correspond to multiple DHCPv6 servers. If there are no authenticated DHCPv6 certs or existing servers fail authentication, the client should retry a number of times. The client conducts the server discovery process as per section 18.1.5 of [RFC3315] to avoid a packet storm. In this way, it is difficult for a rogue server to beat out a busy "real" server. And then the client takes some alternative action depending on its local policy, such as attempting to use an unsecured DHCPv6 server.

Once the server has been authenticated, the DHCPv6 client sends the Encrypted-Query message to the DHCPv6 server. The Encrypted-Query message contains the Encrypted-message option, which MUST be constructed as explained in Section 10.1.6. The Encrypted-message option contains the encrypted DHCPv6 message using the public key contained in the selected cert. In addition, the Server Identifier option MUST be included if it is in the original message (i.e. Request, Renew, Decline, Release) to avoid the need for other servers receiving the message to attempt to decrypt it. The Encrypted-Query message MUST include the Encryption-Key-Tag option to identify the used public/private key pair, which is constructed as explained in Section 10.1.5. The Encrypted-Query message MUST NOT contain any other DHCPv6 option except the Server Identifier option, Encryption-Key-Tag option, Encrypted-Message option.

The first DHCPv6 message sent from the client to the server, such as Solicit message, MUST contain the related information for client authentication. The encryption text SHOULD be formatted as explain in [RFC5652]. The Certificate option MUST be constructed as explained in Section 10.1.2. In addition, one and only one Signature option MUST be contained, which MUST be constructed as explained in Section 10.1.3. One and only one Increasing-number option SHOULD be contained, which MUST be constructed as explained in Section 10.1.4. In addition, the subsequent encrypted DHCPv6 message sent from the client can also contain the Increasing-number option to defend against replay attack.

For the received Encrypted-Response message, the client MUST drop the Encrypted-Response message if other DHCPv6 option except Encrypted-message option is contained. If the transaction-id is 0, the client also try to decrypt it. Then, the client extracts the Encrypted-message option and decrypts it using its private key to obtain the original DHCPv6 message. In this document, it is assumed that the client will not have multiple DHCPv6 sessions with different DHCPv6 servers using different key pairs and only one key pair is used for the encrypted DHCPv6 configuration process. After the decryption, it handles the message as per [RFC3315]. If the decrypted DHCPv6 message contains the Increasing-number option, the DHCPv6 client checks it according to the rule defined in Section 9.1.



If the client fails to get the proper parameters from the chosen server(s), it can select another authenticated certificate and send the Encrypted-Query message to another authenticated server(s) for parameters configuration until the client obtains the proper parameters.

When the decrypted message is Reply message with an error status code, the error status code indicates the failure reason on the server side. According to the received status code, the client MAY take follow-up action:

- o Upon receiving an AuthenticationFail error status code, the client is not able to build up the secure communication with the server. However, there may be other DHCPv6 servers available that successfully complete authentication. The client MAY use the AuthenticationFail as a hint and switch to other DHCPv6 server if it has another one. The client SHOULD retry with another authenticated certificate. However, if the client decides to retransmit using the same certificate after receiving AuthenticationFail, it MUST NOT retransmit immediately and MUST follow normal retransmission routines defined in [RFC3315].
- o Upon receiving a ReplayDetected error status code, the client MAY resend the message with an adjusted Increasing-number option according to the returned number from the DHCPv6 server.
- o Upon receiving a SignatureFail error status code, the client MAY resend the message following normal retransmission routines defined in [RFC3315].

## 7. DHCPv6 Server Behavior

The secure DHCPv6 server is pre-configured with a certificate and its corresponding private key for server authentication. If the server does not obtain the certificate from Certificate Authority (CA), it can generate the self-signed certificate.

When the DHCPv6 server receives the Information-request message and the contained Option Request option identifies the request is for the server's certificate information, it SHOULD first check the hash, signature, encryption algorithms sets that the client supports. The server selects one hash, signature, encryption algorithm from the acknowledged algorithms sets for the future communication. And then, the server replies with a Reply message to the client. The Reply message MUST contain the requested Certificate option, which MUST be constructed as explained in Section 10.1.2, and Server Identifier option. In addition, the Reply message MUST contain one and only one Signature option, which MUST be constructed as explained in

Section 10.1.3. Besides, the Reply message SHOULD contain one and only one Increasing-number option, which MUST be constructed as explained in Section 10.1.4.

Upon the receipt of Encrypted-Query message, the server MUST drop the message if the other DHCPv6 option is contained except Server Identifier option, Encryption-Key-Tag option, Encrypted-message option. Then, the server checks the Server Identifier option. The DHCPv6 server drops the message that is not for it, thus not paying cost to decrypt messages. If it is the target server, according to the Encryption-Key-Tag option, the server identifies the used public/private key pair and decrypts the Encrypted-message option using the corresponding private key. It is essential to note that the encryption key tag is not a unique identifier. It is theoretically possible for two different public keys to share one common encryption key tag. The encryption key tag is used to limit the possible candidate keys, but it does not uniquely identify a public/private key pair. The server MUST try all corresponding key pairs. If the server cannot find the corresponding private key of the key tag or the corresponding private key of the key tag is invalid for decryption, then the server drops the received message.

If secure DHCPv6 server needs client authentication and decrypted message is a Solicit/Information-request message which contains the information for client authentication, the secure DHCPv6 server discards the received message that meets any of the following conditions:

- o the Signature option is missing,
- o multiple Signature options are present,
- o the Certificate option is missing.

For the signature failure, the server SHOULD send an encrypted Reply message with an UnspecFail (value 1, [RFC3315]) error status code to the client.

The server validates the client's certificate through the local pre-configured trusted certificates list. A certificate that finds a match in the local trust certificates list is treated as verified. If the server does not know the certificate and can accept the non-authenticated encryption, then the server skips the authentication process and uses it for encryption only. The message that fails authentication validation MUST be dropped. In such failure, the DHCPv6 server replies with an encrypted Reply message with an AuthenticationFail error status code, defined in Section 10.3, back

to the client. At this point, the server has either recognized the authentication of the client, or decided to drop the message.

If the decrypted message contains the Increasing-number option, the server checks it according to the rule defined in Section 9.1. If the check fails, an encrypted Reply message with a ReplayDetected error status code, defined in Section 10.3, should be sent back to the client. In the Reply message, a Increasing-number option is carried to indicate the server's stored number for the client to use. According to the server's local policy, the message without an Increasing-number option MAY be acceptable or rejected.

The Signature field verification MUST show that the signature has been calculated as specified in Section 10.1.3. If the signature check fails, the DHCPv6 server SHOULD send an encrypted Reply message with a SignatureFail error status code. Only the clients that get through both the signature verification and increasing number check (if there is a Increasing-number option) are accepted as authenticated clients and continue to be handled their message as defined in [RFC3315].

Once the client has been authenticated, the DHCPv6 server sends the Encrypted-response message to the DHCPv6 client. If the DHCPv6 message is Reconfigure message, then the server set the transaction-id of the Encrypted-Response message to 0. The Encrypted-response message MUST only contain the Encrypted-message option, which MUST be constructed as explained in Section 10.1.6. The encryption text SHOULD be formatted as explain in [RFC5652]. The Encrypted-message option contains the encrypted DHCPv6 message that is encrypted using the authenticated client's public key. To provide the replay protection, the Increasing-number option SHOULD be contained in the encrypted DHCPv6 message.

## 8. Relay Agent Behavior

When a DHCPv6 relay agent receives an Encrypted-query or Encrypted-response message, it may not recognize this message. The unknown messages MUST be forwarded as described in [RFC7283].

When a DHCPv6 relay agent recognizes the Encrypted-query and Encrypted-response messages, it forwards the message according to section 20 of [RFC3315]. There is nothing more the relay agents have to do, it neither needs to verify the messages from client or server, nor add any secure DHCPv6 options. Actually, by definition in this document, relay agents MUST NOT add any secure DHCPv6 options.

Relay-forward and Relay-reply messages MUST NOT contain any additional Certificate option or Increasing-number option, aside from

those present in the innermost encapsulated messages from the client or server.

## 9. Processing Rules

### 9.1. Increasing Number Check

In order to check the Increasing-number option, defined in Section 10.1.4, the client/server has one stable stored number for replay attack detection. The server should keep a record of the increasing number forever. And the client keeps a record of the increasing number during the DHCPv6 configuration process with the DHCPv6 server. And the client can forget the increasing number information after the transaction is finished. The client's initial locally stored increasing number is set to zero.

It is essential to remember that the increasing number is finite. All arithmetic dealing with sequence numbers must be performed modulo  $2^{64}$ . This unsigned arithmetic preserves the relationship of sequence numbers as they cycle from  $2^{64} - 1$  to 0 again.

In order to check the Increasing-number option, the following comparison is needed.

NUM.STO = the stored number in the client/server

NUM.REC = the acknowledged number from the received message

The Increasing-number option in the received message passes the increasing number check if NUM.REC is more than NUM.STO. And then, the value of NUM.STO is changed into the value of NUM.REC.

The increasing number check fails if NUM.REC is equal with or less than NUM.STO.

### 9.2. Encryption Key Tag Calculation

The generation method of the encryption key tag adopts the method define in Appendix B in [RFC4034].

The following reference implementation calculates the value of the encryption key tag. The input is the data of the public key. The code is written for clarity not efficiency.

```

/*
 * First octet of the key tag is the most significant 8 bits of the
 * return value;
 * Second octet of the key tag is the least significant 8 bits of the
 * return value.
 */

unsigned int
keytag (
    unsigned char key[], /* the RDATA part of the DNSKEY RR */
    unsigned int keysize /* the RDLENGTH */
)
{
    unsigned long ac;      /* assumed to be 32 bits or larger */
    int i;                 /* loop index */

    for ( ac = 0, i = 0; i < keysize; ++i )
        ac += (i & 1) ? key[i] : key[i] << 8;
    ac += (ac >> 16) & 0xFFFF;
    return ac & 0xFFFF;
}

```

## 10. Extensions for Secure DHCPv6

This section describes the extensions to DHCPv6. Six new DHCPv6 options, two new DHCPv6 messages and six new status codes are defined.

### 10.1. New DHCPv6 Options

#### 10.1.1. Algorithm Option

The Algorithm option carries the algorithms sets for algorithm agility, which is contained in the Information-request message.

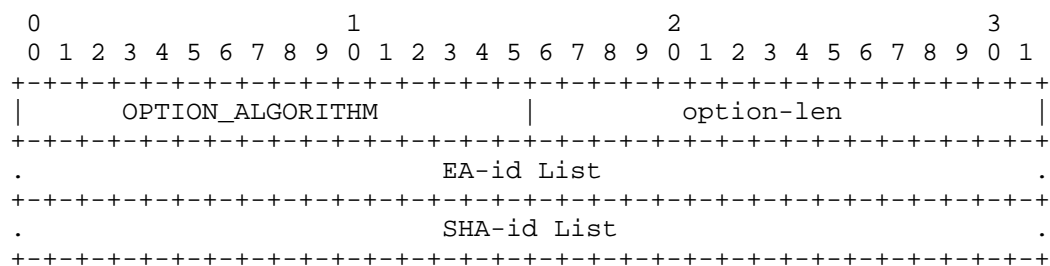
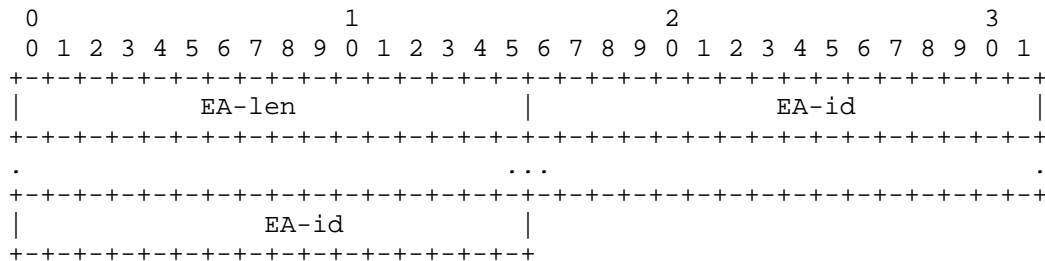


Figure 2: Algorithm Option

- o option-code: OPTION\_ALGORITHM (TBA1).
- o option-len: length of EA-id List + length of SHA-id List in octets.
- o EA-id: The format of the EA-id List field is shown in Figure 3.

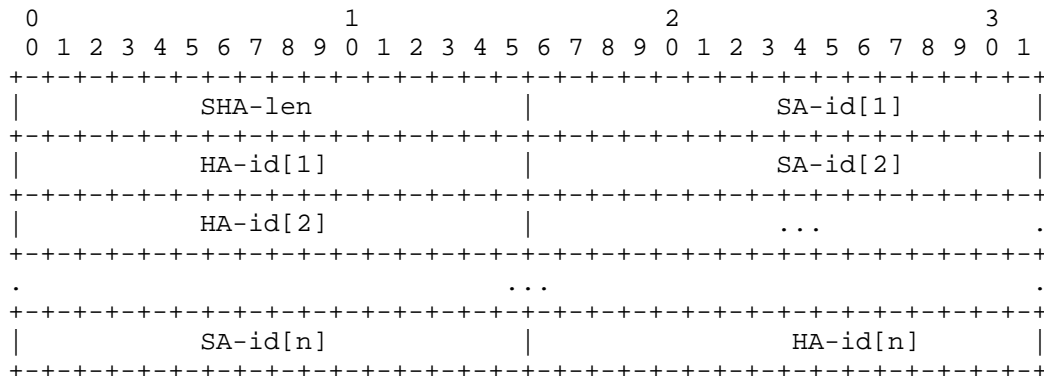


EA-len            The length of the following EA-ids.

EA-id            2-octets value to indicate the Encryption Algorithm id. The client enumerates the list of encryption algorithms it supports to the server. The encryption algorithm is used for the encrypted DHCPv6 configuration process. This design is adopted in order to provide encryption algorithm agility. The value is from the Encryption Algorithm for Secure DHCPv6 registry in IANA. A registry of the initial assigned values is defined in Section 12. The RSA algorithm, as the mandatory encryption algorithm, MUST be included.

Figure 3: EA-id List Field

- o SHA-id List: The format of the SHA-id List field is shown in Figure 4. The SHA-id List contains multiple pair of (SA-id, HA-id). Each pair of (SA-id[i], HA-id[i]) is considered to specify a specific signature method.



SHA-len                    The length of the following SA-id and HA-id pairs.

SA-id                    2-octets value to indicate the Signature Algorithm id. The client enumerates the list of signature algorithms it supports to the server. 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 support of RSASSA-PKCS1-v1\_5 is mandatory. A registry of the initial assigned values is defined in Section 12. The mandatory signature algorithms MUST be included.

HA-id                    2-octets value to indicate the Hash Algorithm id. The client enumerates the list of hash algorithms it supports to the server. 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 support of SHA-256 is mandatory. A registry of the initial assigned values is defined in Section 12. The mandatory hash algorithms MUST be included.

Figure 4: SHA-id List Field

#### 10.1.2. Certificate Option

The Certificate option carries the certificate of the client/server, which is contained in the Reply message. The format of the Certificate option is described as follows:

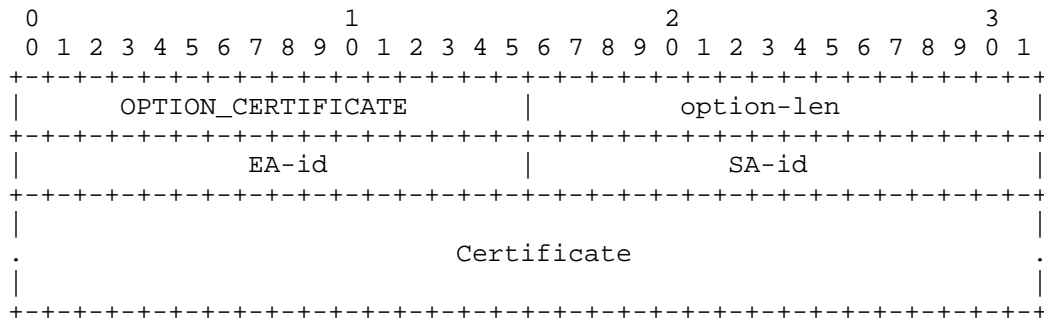


Figure 5: Certificate Option

- o option-code: OPTION\_CERTIFICATE (TBA2).
- o option-len: 4 + length of Certificate in octets.
- o EA-id: Encryption Algorithm id which is used for the certificate. If the value of the EA-id is 0, then the public key in the certificate is not used for encryption calculation.
- o SA-id: Signature Algorithm id which is used for the certificate. If the value of the EA-id is 0, then the public key in the certificate is not used for signature calculation.
- o Certificate: A variable-length field containing certificates. The encoding of certificate and certificate data MUST be in format as defined in Section 3.6, [RFC7296]. The support of X.509 certificate is mandatory.

It should be noticed that the scenario where the values of EA-id and SA-id are both 0 makes no sense and the client MUST discard a message with such values.

#### 10.1.3. Signature option

The Signature option contains a signature that is signed by the private key to be attached to the Reply message. The Signature option could be in any place within the DHCPv6 message while it is logically created after the entire DHCPv6 header and options. It protects the entire DHCPv6 header and options, including itself. The format of the Signature option is described as follows:



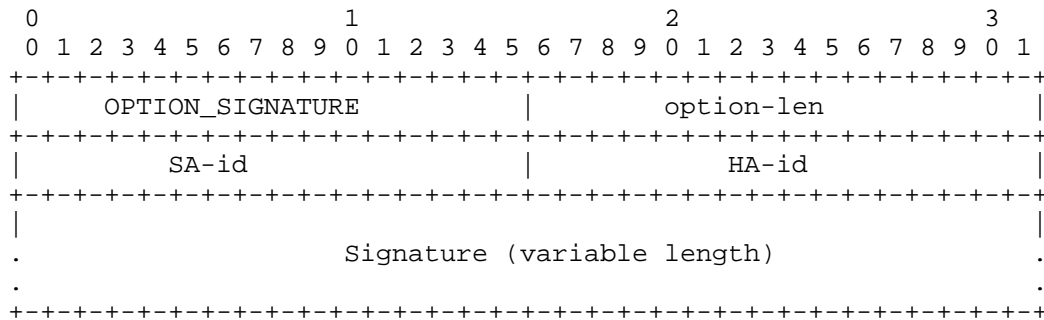


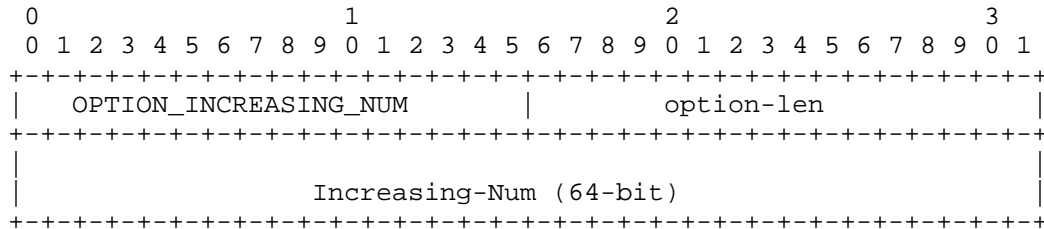
Figure 6: Signature Option

- o option-code: OPTION\_SIGNATURE (TBA3).
- o option-len: 4 + length of Signature field in octets.
- o 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 support of RSASSA-PKCS1-v1\_5 is mandatory. A registry of the initial assigned values is defined in Section 12.
- o 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 support of SHA-256 is mandatory. A registry of the initial assigned values is defined in Section 12.
- o 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 field MUST be padded, with all 0, to the next octet boundary if its size is not a multiple of 8 bits. The padding length depends on the signature algorithm, which is indicated in the SA-id field.

Note: If Secure DHCPv6 is used, the DHCPv6 message is encrypted in a way that the authentication mechanism defined in RFC3315 does not understand. So the Authentication option SHOULD NOT be used if Secure DHCPv6 is applied.

## 10.1.4. Increasing-number Option

The Increasing-number option carries the strictly increasing number for anti-replay protection, which is contained in the Reply message and the encrypted DHCPv6 message. It is optional.



option-code      OPTION\_INCREASING\_NUM (TBA4).

option-len        8, in octets.

Increasing-Num    A strictly increasing number for the replay attack detection which is more than the local stored number.

Figure 7: Increasing-number Option

## 10.1.5. Encryption-Key-Tag Option

The Encryption-Key-Tag option carries the key identifier which is calculated from the public key data. The Encrypted-Query message MUST contain the Encryption-Key-Tag option to identify the used public/private key pair.

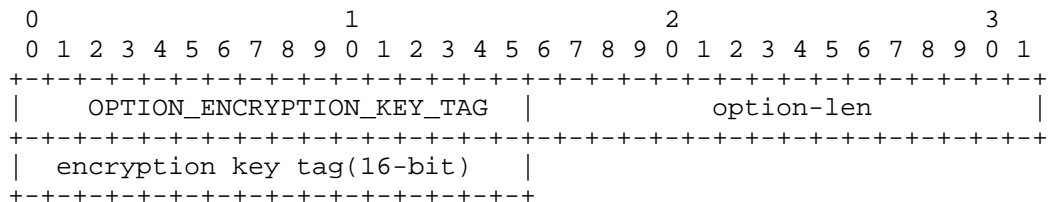


Figure 8: Encryption-Key-Tag Option

option-code      OPTION\_ENCRYPTION\_KEY\_TAG (TBA5).

option-len        2, in octets.

encryption key tag    A 16 bits field containing the encryption key tag sent from the client to server to identify the used public/private key pair. The encryption key tag is calculated from the public

key data, like fingerprint of a specific public key. The specific calculation method of the encryption key tag is illustrated in Section 9.2.

#### 10.1.6. Encrypted-message Option

The Encrypted-message option carries the encrypted DHCPv6 message, which is calculated with the recipient's public key. The Encrypted-message option is contained in the Encrypted-Query message or the Encrypted-Response message.

The format of the Encrypted-message option is:

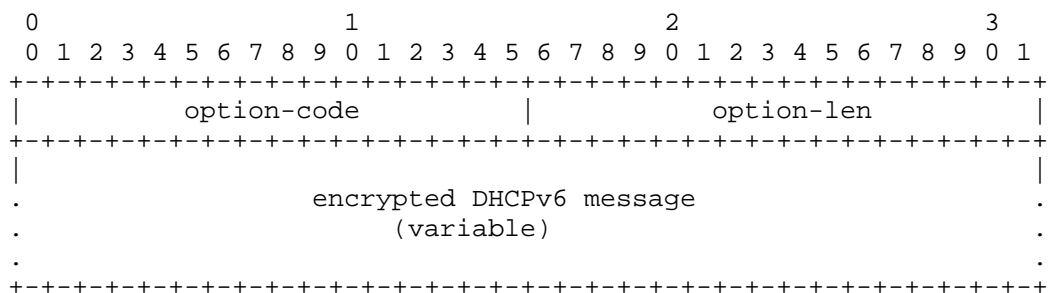


Figure 9: Encrypted-message Option

option-code    OPTION\_ENCRYPTED\_MSG (TBA6).

option-len    Length of the encrypted DHCPv6 message in octets.

encrypted DHCPv6 message    A variable length field containing the encrypted DHCPv6 message. In Encrypted-Query message, it contains encrypted DHCPv6 message sent from a client to server. In Encrypted-response message, it contains encrypted DHCPv6 message sent from a server to client.

#### 10.2. New DHCPv6 Messages

Two new DHCPv6 messages are defined to achieve the DHCPv6 encryption: Encrypted-Query and Encrypted-Response. Both the DHCPv6 messages defined in this document share the following format:

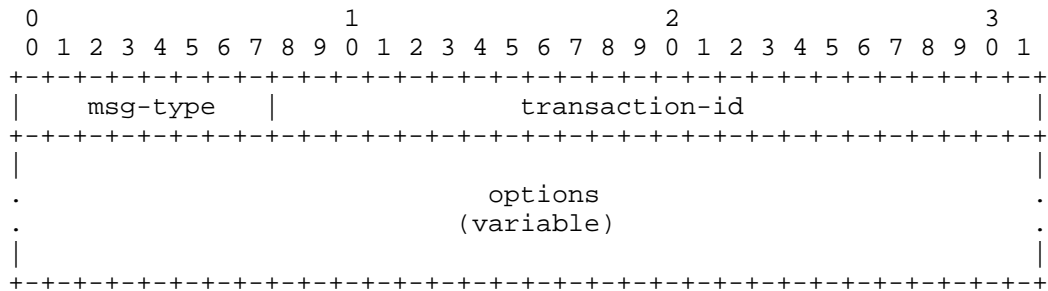


Figure 10: The format of Encrypted-Query and Encrypted-Response Messages

msg-type	Identifier of the message type. It can be either Encrypted-Query (TBA7) or DHCPv6-Response (TBA8).
transaction-id	The transaction ID for this message exchange.
options	The Encrypted-Query message MUST contain the Encrypted-message option, Encryption-Key-Tag option and Server Identifier option if the message in the Encrypted-message option has a Server Identifier option. The Encrypted-Response message MUST only contain the Encrypted-message option.

### 10.3. Status Codes

The following new status codes, see Section 5.4 of [RFC3315] are defined.

- o AuthenticationFail (TBD9): indicates that the message from the DHCPv6 client fails authentication check.
- o ReplayDetected (TBD10): indicates the message from DHCPv6 client fails the increasing number check.
- o SignatureFail (TBD11): indicates the message from DHCPv6 client fails the signature check.

## 11. Security Considerations

This document provides the authentication and encryption mechanisms for DHCPv6.

There are some mandatory algorithm for encryption algorithm in this document. It may be at some point that the mandatory algorithm is no longer safe to use.

A server or a client, whose local policy accepts messages without a Increasing-number option, may have to face the risk of replay attacks.

Since the algorithm option isn't protected by a signature, the list can be forged without detection, which can lead to a downgrade attack.

Likewise, since the Encryption-Key-Tag Option isn't protected, an attacker that can intercept the message can forge the value without detection.

If the client tries more than one cert for client authentication, the server can easily get a client that implements this to enumerate its entire cert list and probably learn a lot about a client that way. For this security item, It is RECOMMENDED that client certificates could be tied to specific server certificates by configuration.

## 12. IANA Considerations

This document defines six new DHCPv6 [RFC3315] options. The IANA is requested to assign values for these six options from the DHCPv6 Option Codes table of the DHCPv6 Parameters registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>. The six options are:

The Algorithm Option (TBA1), described in Section 10.1.2.

The Certificate Option (TBA2), described in Section 10.1.2.

The Signature Option (TBA3), described in Section 10.1.3.

The Increasing-number Option (TBA4), described in Section 10.1.4.

The Encryption-Key-Tag Option (TBA5), described in Section 10.1.5.

The Encrypted-message Option (TBA6), described in Section 10.1.6.

The IANA is also requested to assign value for these two messages from the DHCPv6 Message Types table of the DHCPv6 Parameters registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>. The two messages are:

The Encrypted-Query Message (TBA7), described in Section 10.2.

The Encrypted-Response Message (TBA8), described in Section 10.2.

The IANA is also requested to add three new registry tables to the DHCPv6 Parameters registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>. The three tables are the Hash Algorithm for Secure DHCPv6 table, the Signature Algorithm for Secure DHCPv6 table and the Encryption 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
SHA-256	0x01	this document
SHA-512	0x02	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
Non-SigAlg	0x00	this document
RSASSA-PKCS1-v1_5	0x01	this document

Encryption algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for encryption algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
Non-EncryAlg	0x00	this document
RSA	0x01	this document

IANA is requested to assign the following new DHCPv6 Status Codes, defined in Section 10.3, in the DHCPv6 Parameters registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>:

Code	Name	Reference
TBD9	AuthenticationFail	this document
TBD10	ReplayDetected	this document
TBD11	SignatureFail	this document

### 13. Acknowledgements

The authors would like to thank Tomek Mrugalski, Bernie Volz, Jianping Wu, Randy Bush, Yiu Lee, Sean Shen, Ralph Droms, Jari Arkko, Sean Turner, Stephen Farrell, Christian Huitema, Stephen Kent, Thomas Huth, David Schumacher, Francis Dupont, Gang Chen, Suresh Krishnan, Fred Templin, Robert Elz, Nico Williams, Erik Kline, Alan DeKok, Bernard Aboba, Sam Hartman, Zilong Liu and other members of the IETF DHC working group for their valuable comments.

This document was produced using the xml2rfc tool [RFC2629].

### 14. Change log [RFC Editor: Please remove]

draft-ietf-dhc-sedhcpv6-21: Add the reference of draft-ietf-dhc-relay-server-security. Change the SA-ID List as SHA-ID List and delete the HA-id List. The SHA-id List contains the SA-id and HA-id pairs. Add some statements about the Reconfigure message process. Add some specific text on the encryption key tag calculation method; Add more text on security consideration; Changes some mistakes and grammar mistakes

draft-ietf-dhc-sedhcpv6-20: Correct a few grammar mistakes.

draft-ietf-dhc-sedhcpv6-19: In client behavior part, we adds some description about opportunistic security. In this way, in some scenario, authentication is optional. Add the reference of RFC 4034 for the encryption key tag calculation. Delete the part that the relay agent cache server announcements part. Add the assumption that the client's initial stored increasing number is set to zero. In this way, for the first time increasing number check in the Reply message, the check will always succeed, and then the locally stored number is changed into the contained number in the Reply message. Correct many grammar mistakes.

draft-ietf-dhc-sedhcpv6-18: Add the Algorithm option. The algorithm option contains the EA-id List, SA-id List, HA-id List, and then the certificate and signature options do not contain the algorithm list; Add the Encryption Key Tag option to identify the used public/private key pair; Delete the AlgorithmNotSupported error status code; Delete some description on that secure DHCPv6 exchanges the server selection method; Delete the DecryptionFail error status code; For the case where the client's certificate is missed, then the server discards the received message. Add the assumption that: For DHCPv6 client, just one certificate is used for the DHCPv6 configuration. Add the statement that: For the first Encrypted-Query message, the server needs to try all the possible private keys and then records the relationship between the public key and the encryption key tag.

draft-ietf-dhc-sedhcpv6-17: Change the format of the certificate option according to the comments from Bernie.

draft-ietf-dhc-sedhcpv6-16: For the algorithm agility part, the provider can offer multiple EA-id, SA-id, HA-id and then receiver choose one from the algorithm set.

draft-ietf-dhc-sedhcpv6-15: Increasing number option only contains the strictly increasing number; Add some description about why encryption is needed in Security Issues of DHCPv6 part;

draft-ietf-dhc-sedhcpv6-14: For the deployment part, Tofu is out of scope and take Opportunistic security into consideration; Increasing number option is changed into 64 bits; Increasing number check is a separate section; IncreasingnumFail error status code is changed into ReplayDetected error status code; Add the section of "caused change to RFC3315";

draft-ietf-dhc-sedhcpv6-13: Change the Timestamp option into Increasing-number option and the corresponding check method; Delete the OSCP stamping part for the certificate check; Add the scenario where the hash and signature algorithms cannot be separated; Add the comparison with RFC7824 and RFC7844; Add the encryption text format and reference of RFC5652. Add the consideration of scenario where multiple DHCPv6 servers share one common DHCPv6 server. Add the statement that Encrypted-Query and Encrypted-Response messages can only contain certain options: Server Identifier option and Encrypted-message option. Add opportunistic security for deployment consideration. Besides authentication+encryption mode, encryption-only mode is added.

draft-ietf-dhc-sedhcpv6-12: Add the Signature option and timestamp option during server/client authentication process. Add the hash function and signature algorithm. Add the requirement: The Information-request message cannot contain any other options except ORO option. Modify the use of "SHOULD"; Delete the reference of RFC5280 and modify the method of client/server cert verification; Add the relay agent cache function for the quick response when there is no authenticated server. 2016-4-24.

draft-ietf-dhc-sedhcpv6-11: Delete the Signature option, because the encrypted DHCPv6 message and the Information-request message (only contain the Certificate option) don't need the Signature option for message integrity check; Rewrite the "Applicability" section; Add the encryption algorithm negotiation process; To support the encryption algorithm negotiation, the Certificate option contains the EA-id(encryption algorithm identifier) field; Reserve the Timestamp option to defend against the replay attacks for encrypted DHCPv6



configuration process; Modify the client behavior when there is no authenticated DHCPv6 server; Add the DecryptionFail error code. 2016-3-9.

draft-ietf-dhc-sedhcpv6-10: merge DHCPv6 authentication and DHCPv6 encryption. The public key option is removed, because the device can generate the self-signed certificate if it is pre-configured the public key not the certificate. 2015-12-10.

draft-ietf-dhc-sedhcpv6-09: change some texts about the deployment part. 2015-12-10.

draft-ietf-dhc-sedhcpv6-08: clarified what the client and the server should do if it receives a message using unsupported algorithm; refined the error code treatment regarding to AuthenticationFail and TimestampFail; added consideration on how to reduce the DoS attack when using TOFU; other general editorial cleanups. 2015-06-10.

draft-ietf-dhc-sedhcpv6-07: removed the deployment consideration section; instead, described more straightforward use cases with TOFU in the overview section, and clarified how the public keys would be stored at the recipient when TOFU is used. The overview section also clarified the integration of PKI or other similar infrastructure is an open issue. 2015-03-23.

draft-ietf-dhc-sedhcpv6-06: remove the limitation that only clients use PKI- certificates and only servers use public keys. The new text would allow clients use public keys and servers use PKI-certificates. 2015-02-18.

draft-ietf-dhc-sedhcpv6-05: addressed comments from mail list that responded to the second WGLC. 2014-12-08.

draft-ietf-dhc-sedhcpv6-04: addressed comments from mail list. Making timestamp an independent and optional option. Reduce the serverside authentication to base on only client's certificate. Reduce the clientside authentication to only Leaf of Faith base on server's public key. 2014-09-26.

draft-ietf-dhc-sedhcpv6-03: addressed comments from WGLC. Added a new section "Deployment Consideration". Corrected the Public Key Field in the Public Key Option. Added consideration for large DHCPv6 message transmission. Added TimestampFail error code. Refined the retransmission rules on clients. 2014-06-18.

draft-ietf-dhc-sedhcpv6-02: addressed comments (applicability statement, redesign the error codes and their logic) from IETF89 DHC WG meeting and volunteer reviewers. 2014-04-14.

draft-ietf-dhc-sedhcpv6-01: addressed comments from IETF88 DHC WG meeting. Moved Dacheng Zhang from acknowledgement to be co-author. 2014-02-14.

draft-ietf-dhc-sedhcpv6-00: adopted by DHC WG. 2013-11-19.

draft-jiang-dhc-sedhcpv6-02: 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.

## 15. References

### 15.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <<http://www.rfc-editor.org/info/rfc2460>>.
- [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <<http://www.rfc-editor.org/info/rfc3315>>.
- [RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <<http://www.rfc-editor.org/info/rfc3971>>.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<http://www.rfc-editor.org/info/rfc4034>>.

- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, DOI 10.17487/RFC4443, March 2006, <<http://www.rfc-editor.org/info/rfc4443>>.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <<http://www.rfc-editor.org/info/rfc5652>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<http://www.rfc-editor.org/info/rfc5905>>.
- [RFC6840] Weiler, S., Ed. and D. Blacka, Ed., "Clarifications and Implementation Notes for DNS Security (DNSSEC)", RFC 6840, DOI 10.17487/RFC6840, February 2013, <<http://www.rfc-editor.org/info/rfc6840>>.
- [RFC7283] Cui, Y., Sun, Q., and T. Lemon, "Handling Unknown DHCPv6 Messages", RFC 7283, DOI 10.17487/RFC7283, July 2014, <<http://www.rfc-editor.org/info/rfc7283>>.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<http://www.rfc-editor.org/info/rfc7296>>.
- [RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <<http://www.rfc-editor.org/info/rfc7435>>.
- [RFC7824] Krishnan, S., Mrugalski, T., and S. Jiang, "Privacy Considerations for DHCPv6", RFC 7824, DOI 10.17487/RFC7824, May 2016, <<http://www.rfc-editor.org/info/rfc7824>>.
- [RFC7844] Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity Profiles for DHCP Clients", RFC 7844, DOI 10.17487/RFC7844, May 2016, <<http://www.rfc-editor.org/info/rfc7844>>.

## 15.2. Informative References

- [I-D.ietf-dhc-relay-server-security]  
Volz, B. and Y. Pal, "Security of Messages Exchanged  
Between Servers and Relay Agents", draft-ietf-dhc-relay-  
server-security-03 (work in progress), February 2017.
- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629,  
DOI 10.17487/RFC2629, June 1999,  
<<http://www.rfc-editor.org/info/rfc2629>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an  
IANA Considerations Section in RFCs", BCP 26, RFC 5226,  
DOI 10.17487/RFC5226, May 2008,  
<<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC6273] Kukec, A., Krishnan, S., and S. Jiang, "The Secure  
Neighbor Discovery (SEND) Hash Threat Analysis", RFC 6273,  
DOI 10.17487/RFC6273, June 2011,  
<<http://www.rfc-editor.org/info/rfc6273>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an  
Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May  
2014, <<http://www.rfc-editor.org/info/rfc7258>>.
- [RSA] RSA Laboratories, "RSA Encryption Standard, Version 2.1,  
PKCS 1", November 2002.

## Authors' Addresses

Lishan Li  
Tsinghua University  
Beijing 100084  
P.R.China

Phone: +86-15201441862  
Email: [lilishan48@gmail.com](mailto:lilishan48@gmail.com)

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

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

Yong Cui  
Tsinghua University  
Beijing 100084  
P.R.China

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

Tatuya Jinmei  
Infoblox Inc.  
3111 Coronado Drive  
Santa Clara, CA  
US

Email: jinmei@wide.ad.jp

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

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

Dacheng Zhang  
Beijing  
CN

Email: dacheng.zhang@gmail.com

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: January 5, 2015

T. Lemon  
Nominum, Inc.  
T. Mrugalski  
Internet Systems Consortium, Inc.  
July 4, 2014

Customizing DHCP Configuration on the Basis of Network Topology  
draft-ietf-dhc-topo-conf-02

Abstract

DHCP servers have evolved over the years to provide significant functionality beyond that which is described in the DHCP base specifications. One aspect of this functionality is support for context-specific configuration information. This memo describes some such features and makes recommendations as to how they can be used.

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 5, 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 . . . . .	3
3. Locality . . . . .	3
4. Simple Subnetted Network . . . . .	8
5. Relay agent running on a host . . . . .	10
6. Cascade relays . . . . .	10
7. Regional Configuration Example . . . . .	11
8. Dynamic Lookup . . . . .	13
9. Multiple subnets on the same link . . . . .	14
10. Acknowledgments . . . . .	14
11. Security Considerations . . . . .	14
12. IANA Considerations . . . . .	15
13. Informative References . . . . .	15
Authors' Addresses . . . . .	15

## 1. Introduction

The DHCPv4 [RFC2131] and DHCPv6 [RFC3315] protocol specifications describe how addresses can be allocated to clients based on network topology information provided by the DHCP relay infrastructure. Address allocation decisions are integral to the allocation of addresses and prefixes in DHCP.

The DHCP protocol also describes mechanisms for provisioning devices with additional configuration information; for example, DNS [RFC1034] server addresses, default DNS search domains, and similar information.

Although it was the intent of the authors of these specifications that DHCP servers would provision devices with configuration information appropriate to each device's location on the network, this practice was never documented, much less described in detail.

Existing DHCP server implementations do in fact provide such capabilities; the goal of this document is to describe those capabilities for the benefit both of operators and of protocol designers who may wish to use DHCP as a means for configuring their own services, but may not be aware of the capabilities provided by most modern DHCP servers.

## 2. Terminology

- o Routable IP address: an IP address with a scope of use wider than the local link.
- o PE router: Provider Edge Router. The provider router closest to the customer.
- o CPE device: customer premise equipment device. Typically a router belonging to the customer that connects directly to the provider link.
- o Shared subnet: a case where two or more subnets of the same protocol family are available on the same link. 'Share subnet' terminology is typically used in Unix environments. It is typically called 'multinet' in Windows environment. The administrative configuration inside a Microsoft DHCP server is called 'DHCP Superscope'.

## 3. Locality

Figure 1 illustrates a simple hierarchy of network links with Link D serving as a backbone to which the DHCP server is attached.

Figure 2 illustrates a more complex case. Although some of its aspects are unlikely to be seen in an actual production networks, they are beneficial for explaining finer aspects of the DHCP protocols. Note that some nodes act as routers (which forward all IPv6 traffic) and some are relay agents (i.e. run DHCPv6 specific software that forwards only DHCPv6 traffic).



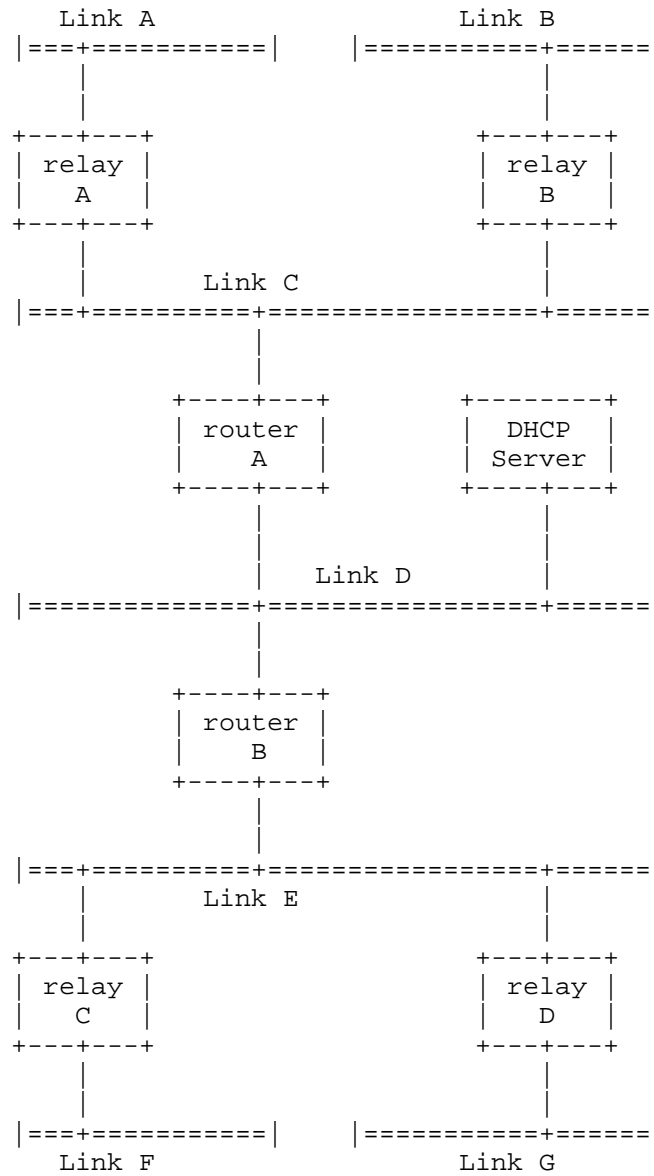


Figure 1: A simple network

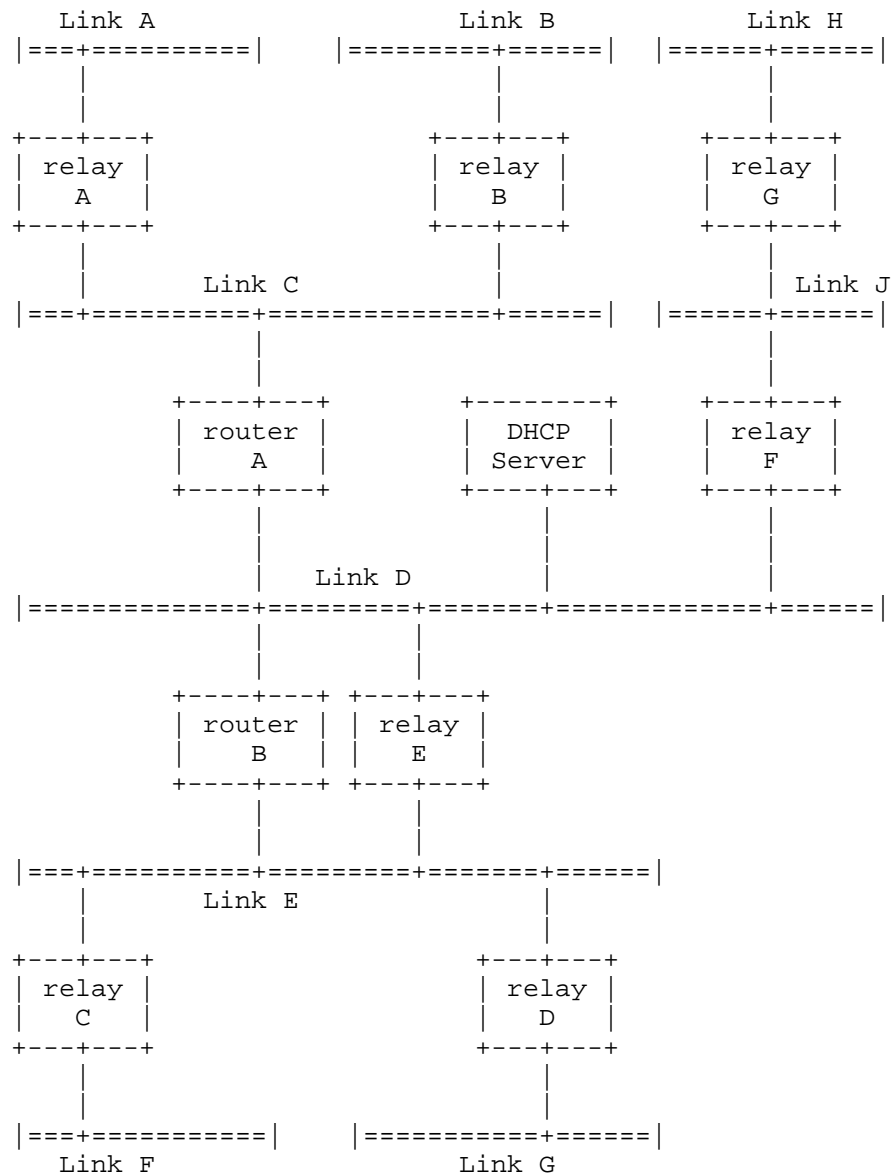


Figure 2: Complex network

This diagram allows us to represent a variety of different network configurations and illustrate how existing DHCP servers can provide configuration information customized to the particular location from which a client is making its request.

It's important to understand the background of how DHCP works when considering this diagram. DHCP clients are assumed not to have routable IP addresses when they are attempting to obtain configuration information.

The reason for making this assumption is that one of the functions of DHCP is to bootstrap the DHCP client's IP address configuration; if the client does not yet have an IP address configured, it cannot route packets to an off-link DHCP server, therefore some kind of relay mechanism is required.

The details of how packet delivery between clients and servers works are different between DHCPv4 and DHCPv6, but the essence is the same: whether or not the client actually has an IP configuration, it generally communicates with the DHCP server by sending its requests to a DHCP relay agent on the local link; this relay agent, which has a routable IP address, then forwards the DHCP requests to the DHCP server. In some cases in DHCPv4, when a DHCP client has a routable IPv4 address, the message is unicast to the DHCP server rather than going through a relay agent. In DHCPv6 that is also possible in case where the server is configured with a Server Unicast option (see Section 22.12 in [RFC3315]) and clients are able to take advantage of it. In such case once the clients get their (presumably global) addresses, they are able to contact server directly, bypassing relays. It should be noted that such a mode is completely controllable by administrators in DHCPv6. (They may simply choose to not configure server unicast option, thus forcing clients to send their messages always via relay agents).

In all cases, the DHCP server is able to obtain an IP address that it knows is on-link for the link to which the DHCP client is connected: either the DHCPv4 client's routable IPv4 address, or the relay agent's IPv4 address on the link to which the client is connected. So in every case the server is able to determine the client's point of attachment and select appropriate subnet- or link-specific configuration.

In the DHCPv6 protocol, there are two mechanisms defined in [RFC3315] that allow server to distinguish which link the relay agent is connected to. The first mechanism is a link-address field in the RELAY-FORW and RELAY-REPL messages. Somewhat contrary to its name, relay agents insert in the link-address field an address that is typically global and can be used to uniquely identify the link on which the client is located. In normal circumstances this is the solution that is easiest to maintain. It requires, however, for the relay agent to have an address with a scope larger than link-local configured on its client-facing interface. If for whatever reason that is not feasible (e.g. because the relay agent does not have a

global address configured on the client-facing interface), the relay agent includes an Interface-Id option (see Section 22.18 of [RFC3315]) that identifies the link clients are connected to. It is up to administrator to make sure that the interface-id is unique within his administrative domain. It should be noted that RELAY-FORW and RELAY-REPL messages are exchanged between relays and servers only. Clients are never exposed to those messages. Also, servers never receive RELAY-REPL messages. Relay agents must be able to process both RELAY-FORW (sending already relayed message further towards the server, when there is more than one relay agent in a chain) and RELAY-REPL (when sending back the response towards the client, when there is more than one relay agent in a chain).

DHCPv6 also has support for more finely grained link identification, using Lightweight DHCPv6 Relay Agents [RFC6221] (LDRA). In this case, in addition to receiving an IPv6 address that is on-link for the link to which the client is connected, the DHCPv6 server also receives an Interface-Id option from the relay agent that can be used to more precisely identify the client's location on the network.

What this means in practice is that the DHCP server in all cases has sufficient information to pinpoint, at the very least, the layer 3 link to which the client is connected, and in some cases which layer 2 link the client is connected to, when the layer 3 link is aggregated out of multiple layer 2 links.

In all cases, then, the DHCP server will have a link-identifying IP address, and in some cases it may also have a link-specific identifier (e.g. Interface-Id Option or Link Address Option defined in Section 5 of [RFC6977]). It should be noted that there is no guarantee that the link-specific identifier will be unique outside the scope of the link-identifying IP address.

It is also possible for link-specific identifiers to be nested, so that the actual identifier that identifies the link is an aggregate of two or more link-specific identifiers sent by a set of LDRAs in a chain; in general this functions exactly as if a single identifier were received from a single LDRA, so we do not treat it specially in the discussion below, but sites that use chained LDRA configurations will need to be aware of this when configuring their DHCP servers.

So let's examine the implications of this in terms of how a DHCP server can deliver targeted supplemental configuration information to DHCP clients.

#### 4. Simple Subnetted Network

Consider Figure 1 in the context of a simple subnetted network. In this network, there are four leaf subnets: links A, B, F and G, on which DHCP clients will be configured. Relays A, B, C and D in this example are represented in the diagram as IP routers with an embedded relay function, because this is a very typical configuration, but the relay function can also be provided in a separate node on each link.

In a simple network like this, there may be no need for link-specific configuration in DHCPv6, since local routing information is delivered through router advertisements. However, in IPv4, it is very typical to configure the default route using DHCP; in this case, the default route will be different on each link. In order to accomplish this, the DHCP server will need link-specific configuration for the default route.

To illustrate, we will use an example from a hypothetical DHCP server that uses a simple JSON notation for configuration. Although we know of no DHCP server that uses this specific syntax, most modern DHCP server provides similar functionality.

```
{
  "prefixes": {
    "192.0.2.0/26": {
      "options": {
        "routers": ["192.0.2.1"]
      },
      "on-link": ["a"]
    },
    "192.0.2.64/26": {
      "options": {
        "routers": ["192.0.2.65"]
      },
      "on-link": ["b"]
    },
    "192.0.2.128/26": {
      "options": {
        "routers": ["192.0.2.129"]
      },
      "on-link": ["f"]
    },
    "192.0.2.192/26": {
      "options": {
        "routers": ["192.0.2.193"]
      },
      "on-link": ["g"]
    }
  }
}
```

Figure 3: Configuration example

In Figure 3, we see a configuration example for this scenario: a set of prefixes, each of which has a set of options and a list of links for which it is on-link. We have defined one option for each prefix: a routers option. This option contains a list of values; each list only has one value, and that value is the IP address of the router specific to the prefix.

When the DHCP server receives a request, it searches the list of prefixes for one that encloses the link-identifying IP address provided by the client or relay agent. The DHCP server then examines the options list associated with that prefix and returns those options to the client.

So for example a client connected to link A in the example would have a link-identifying IP address within the 192.0.2.0/26 prefix, so the DHCP server would match it to that prefix. Based on the configuration, the DHCP server would then return a routers option

containing a single IP address: 192.0.2.1. A client on link F would have a link-identifying address in the 192.0.2.128/26 prefix, and would receive a routers option containing the IP address 192.0.2.129.

#### 5. Relay agent running on a host

Relay agent is a DHCP software that may be run on any IP node. Although it is typically run on a router, this is by no means required by the DHCP protocol. The relay agent is simply a service that operates on a link, receiving link-local multicasts or broadcasts and relaying them, using IP routing, to a DHCP server. As long as the relay has an IP address on the link, and a default route or more specific route through which it can reach a DHCP server, it need not be a router, or even have multiple interfaces.

Relay agent can be run on a host connected to two links. That case is presented in Figure 2. There is router B that is connected to links D and E. At the same time there is also a host that is connected to the same links. The relay agent software is running on that host. That is uncommon, but legal configuration.

#### 6. Cascade relays

Let's observe another case shown in Figure 2. Note that in typical configuration, the clients connected to link G will send their requests to relay D which will forward its packets directly to the DHCP server. That is typical, but not the only possible configuration. It is possible to configure relay agent D to forward client messages to relay E which in turn will send it to the DHCP server. This configuration is sometimes referred to as cascade relay agents.

Note that the relaying mechanism works differently in DHCPv4 and in DHCPv6. In DHCPv4 only the first relay is able to set the GIADDR field in the DHCPv4 packet. Any following relays that receive that packet will not change it as the server needs GIADDR information from the first relay (i.e. the closest to the client). Server will send the response back to the GIADDR address, which is the address of the first relay agent that saw the client's message. That means that the client messages travel on a different path than the server's responses. A message from client connected to link G will travel via relay D, relay E and to the server. A response message will be sent from the server to relay D via router B, and relay D will send it to the client on link G.

Relaying in DHCPv6 is more structured. Each relay agent encapsulates a packet that is destined to the server and sends it towards the server. Depending on the configuration that can be server's unicast

address, a multicast address or next relay agent address. The next relay repeats the encapsulation process. Although the resulting packet is more complex (may have up to 32 levels of encapsulation if traveled through 32 relays), every relay may insert its own options and it is clear which relay agent inserted which option.

## 7. Regional Configuration Example

In this example, link C is a regional backbone for an ISP. Link E is also a regional backbone for that ISP. Relays A, B, C and D are PE routers, and Links A, B, F and G are actually link aggregators with individual layer 2 circuits to each customer--for example, the relays might be DSLAMs or cable head-end systems. At each customer site we assume there is a single CPE device attached to the link.

We further assume that links A, B, F and G are each addressed by a single prefix, although it would be equally valid for each CPE device to be numbered on a separate prefix.

In a real-world deployment, there would likely be many more than two PE routers connected to each regional backbone; we have kept the number small for simplicity.

In the example presented in Figure 4, the goal is to configure all the devices within a region with server addresses local to that region, so that service traffic does not have to be routed between regions unnecessarily.



```

{
  "prefixes": {
    "2001:db8:0:0::/40": {
      "on-link": ["A"]
    },
    "2001:db8:100:0::/40": {
      "on-link": ["B"]
    },
    "2001:db8:200:0::/40": {
      "on-link": ["F"]
    },
    "2001:db8:300:0::/40": {
      "on-link": ["G"]
    }
  },
  "links": {
    "A": {"region": "omashu"},
    "B": {"region": "omashu"},
    "F": {"region": "gaoling"},
    "G": {"region": "gaoling"}
  },
  "regions": {
    "omashu": {
      "options": {
        "sip-servers": ["sip.omashu.example.org"],
        "dns-servers": ["dns1.omashu.example.org",
                        "dns2.omashu.example.org"]
      }
    },
    "gaoling": {
      "options": {
        "sip-servers": ["sip.gaoling.example.org"],
        "dns-servers": ["dns1.gaoling.example.org",
                        "dns2.gaoling.example.org"]
      }
    }
  }
}

```

Figure 4: An example regions configuration

In this example, when a request comes in to the DHCP server with a link-identifying IP address in the 2001:DB8:0:0::/40 prefix, it is identified as being on link A. The DHCP server then looks on the list of links to see what region the client is in. Link A is identified as being in omashu. The DHCP server then looks up omashu in the set of regions, and discovers a list of region-specific options.

The DHCP server then resolves the domain names listed in the options and sends a sip-server option containing the IP addresses that the resolver returned for sip.omashu.example.org, and a dns-server option containing the IP addresses returned by the resolver for dns1.omashu.example.org and dns2.omashu.example.org. Depending on the server capability and configuration, it may cache resolved responses for specific period of time, repeat queries every time or even keep the response until reconfiguration or shutdown.

Similarly, if the DHCP server receives a request from a DHCP client where the link-identifying IP address is contained by the prefix 2001:DB8:300:0::/40, then the DHCP server identifies the client as being connected to link G. The DHCP server then identifies link G as being in the gaoling region, and returns the sip-servers and dns-servers options specific to that region.

As with the previous example, the exact configuration syntax and structure shown above does not precisely match what existing DHCP servers do, but the behavior illustrated in this example can be accomplished with most existing modern DHCP servers.

## 8. Dynamic Lookup

In the Regional example, the configuration listed several domain names as values for the sip-servers and dns-servers options. The wire format of both of these options contains one or more IPv6 addresses--there is no way to return a domain name to the client.

This was understood to be an issue when the original DHCP protocol was defined, and historical implementations even from the very early days would accept domain names and resolve them. Some early DHCP implementations, particularly those based on earlier BOOTP implementations, had very limited capacity for reconfiguration.

However, most modern DHCP servers handle name resolution by querying the resolver each time a DHCP packet comes in. This means that if DHCP servers and DNS servers are managed by different administrative entities, there is no need for the administrators of the DHCP servers and DNS servers to communicate when changes are made. When changes are made to the DNS server, these changes are promptly and automatically adopted by the DHCP server. Similarly, when DHCP server configurations change, DNS server administrators need not be aware of this.

However, it should be noted that even though the DHCP server may be configured to query the DNS server every time it uses configured names, the changes made in the DNS zone may not be visible to the server until the DNS cache expires. If this is not desired, the DHCP

server can be configured to query the authoritative DNS server directly, bypassing any caching DNS servers.

It's worth noting that DNS is not the only way to resolve names, and not all DHCP servers support other techniques (e.g., NIS+ or WINS). However, since these protocols have all but vanished from common use, this won't be an issue in new deployments.

## 9. Multiple subnets on the same link

There are scenarios where there is more than one subnet from the same protocol family (i.e. two or more IPv4 subnets or two or more IPv6 subnets) configured on the same layer 3 link. One example is a slow network renumbering where some services are migrated to the new addressing scheme, but some aren't yet. Second example is a cable network, where cable modems and the devices connected behind them are connected to the same layer 2 link. However, operators want the cable modems and user devices to get addresses from distinct address spaces, so users couldn't easily access their modems management interfaces. Such a configuration is often referred to as 'shared subnets' in Unix environments or 'multinet' in Microsoft terminology.

To support such an configuration, additional differentiating information is required. Many DHCP server implementations offer a feature that is typically called client classification. The server segregates incoming packets into one or more classes based on certain packet characteristics, e.g. presence or value of certain options or even a match between existing options. Servers require additional information to handle such configuration, as it can't use the topographical property of the relay addresses alone to properly choose a subnet. Such information is always implementation specific.

## 10. Acknowledgments

Thanks to Dave Thaler for suggesting that even though "everybody knows" how DHCP servers are deployed in the real world, it might be worthwhile to have an IETF document that explains what everybody knows, because in reality not everybody is an expert in how DHCP servers are administered. Thanks to Andre Kostur, Carsten Strotmann, Simon Perreault, Jinmei Tatuya and Suresh Krishnan for their reviews, comments and feedback.

## 11. Security Considerations

This document explains existing practice with respect to the use of Dynamic Host Configuration Protocol [RFC2131] and Dynamic Host Configuration Protocol Version 6 [RFC3315]. The security considerations for these protocols are described in their

specifications and in related documents that extend these protocols. This document introduces no new functionality, and hence no new security considerations.

## 12. IANA Considerations

The IANA is hereby absolved of any requirement to take any action in relation to this document.

## 13. Informative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.
- [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.
- [RFC6221] Miles, D., Ooghe, S., Dec, W., Krishnan, S., and A. Kavanagh, "Lightweight DHCPv6 Relay Agent", RFC 6221, May 2011.
- [RFC6977] Boucadair, M. and X. Pournard, "Triggering DHCPv6 Reconfiguration from Relay Agents", RFC 6977, July 2013.

## Authors' Addresses

Ted Lemon  
Nominum, Inc.  
2000 Seaport Blvd  
Redwood City, CA 94063  
USA  
  
Phone: +1-650-381-6000  
Email: Ted.Lemon@nominum.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

DHC Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: August 17, 2014

S. Krishnan  
Ericsson  
J. Korhonen  
Broadcom  
S. Bhandari  
Cisco Systems  
February 13, 2014

Support for multiple provisioning domains in DHCPv6  
draft-kkb-mpvd-dhcp-support-01

Abstract

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks. One part of the solution requires associating configuration information with provisioning domains. This document details how configuration information provided through DHCPv6 can be associated with provisioning domains.

Status of this Memo

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

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

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

This Internet-Draft will expire on August 17, 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 . . . . .	3
2. Terminology . . . . .	3
3. PVD Container option . . . . .	3
4. PVD Identity option . . . . .	4
5. PVD Authentication and Authorization option . . . . .	4
6. Set of allowable options . . . . .	6
7. Behaviour of DHCPv6 entities . . . . .	6
7.1. Client and Requesting Router Behavior . . . . .	6
7.2. Server and Delegating Router Behavior . . . . .	6
8. Security Considerations . . . . .	7
9. IANA Considerations . . . . .	8
10. Acknowledgements . . . . .	8
11. Normative References . . . . .	8
Authors' Addresses . . . . .	9

## 1. Introduction

The MIF working group is producing a solution to solve the issues that are associated with nodes that can be attached to multiple networks based on the Multiple Provisioning Domains (MPVD) architecture work [I-D.anipko-mif-mpvd-arch]. One part of the solution requires associating configuration information with provisioning domains. This document describes a DHCPv6 mechanism for explicitly indicating provisioning domain information along with any configuration that will be provided. The proposed mechanism uses a DHCPv6 option that indicates the identity of the provisioning domain and encapsulates the options that contain the configuration information as well as any accompanying authentication/authorization information.

## 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].

## 3. PVD Container option

The PVD container option is used to encapsulate and group together all the configuration options that belong to the explicitly identified provisioning domain. The PVD container option MUST encapsulate exactly one OPTION\_PVD\_ID. The PVD container option MAY occur multiple times in the same message, but each of these PVD container options MUST have a different PVD identity specified under its PVD identity option. The PVD container option SHOULD contain exactly one OPTION\_PVD\_AUTH.

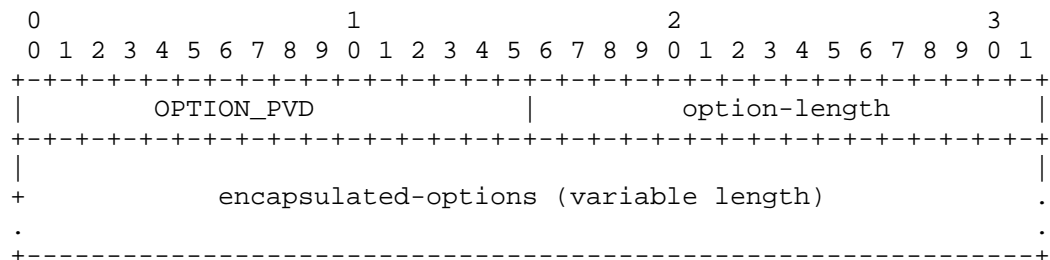


Figure 1: PVD Container Option

- o option-code: OPTION\_PVD (TBA1)
- o option-length: Length of encapsulated options
- o encapsulated-options: options associated with this provisioning domain.

#### 4. PVD Identity option

The PVD identity option is used to explicitly indicate the identity of the provisioning domain that is associated with the configuration information encapsulated by the PVD container option.

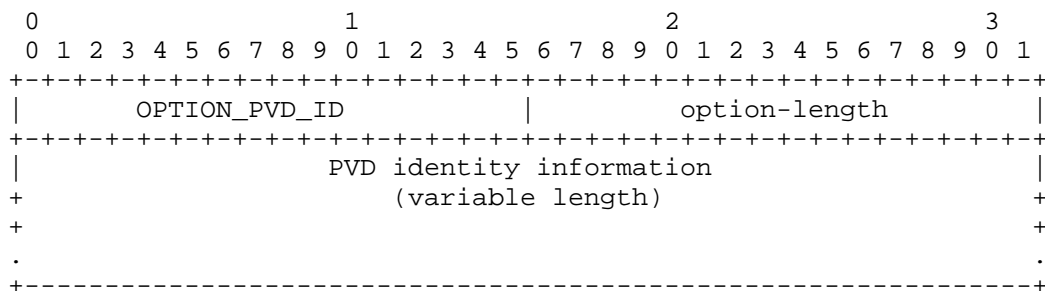


Figure 2: PVD ID Option

- o option-code: OPTION\_PVD\_ID (TBA2)
- o option-length: Length of PVD identity information
- o PVD identity information: The provisioning domain identity. The contents of this field is defined in a separate document [PVDIDS].

#### 5. PVD Authentication and Authorization option

The PVD authentication and authorization option contains information that could be used by the DHCPv6 client to verify whether the configuration information provided was not tampered with by the DHCPv6 server as well as establishing that the DHCPv6 server was authorized to advertise the information on behalf of the PVD per OPTION\_PVD basis. The contents of the authentication/authorization information is provided by the owner of the provisioning domain and is completely opaque to the DHCPv6 server that passes along the information unmodified. Every OPTION\_PVD option SHOULD contain at



most one OPTION\_PVD\_AUTH option. The OPTION\_PVD\_AUTH option MUST be the last option inside the OPTION\_PVD option.

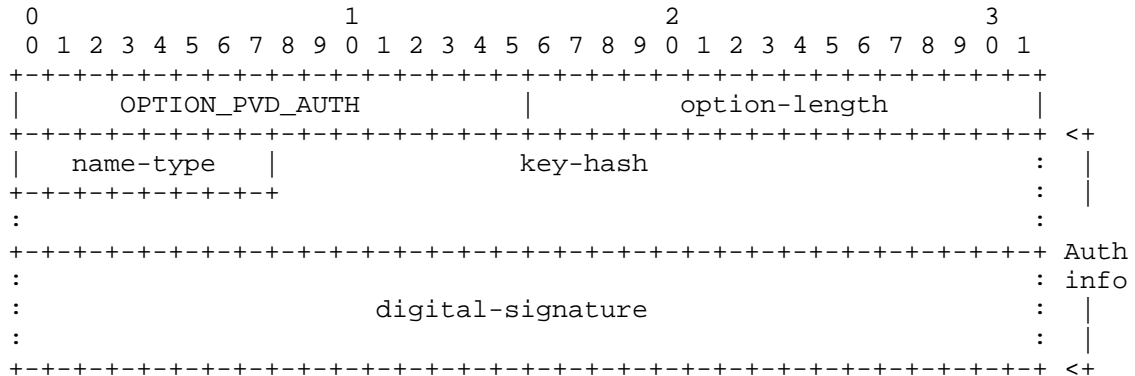


Figure 3: PVD Auth Option

- o option-code: OPTION\_PVD\_AUTH (TBA3)
- o option-length: Length of the Auth info
- o name-type: Names the algorithm used to identify a specific X.509 certificate using the method defined for the Subject Key Identifier (SKI) extension for the X.509 certificates. The usage and the Name Type registry aligns with the mechanism defined for SeND [RFC6494][RFC6495]. Name Type values starting from 3 are supported and an implementation MUST at least support SHA-1 (value 3).
- o key-hash: A hash of the public key using the algorithm identified by the Name Type. The procedure how the Key Hash is calculated is defined in [RFC3971] and [RFC6495]
- o digital-signature: A signature calculated over the encapsulating OPTION\_PVD including all option data from the beginning of the option while setting the digital-signature field to zero. The procedure of calculating the signature is identical to the one defined for SeND [RFC3971].

[TODO: There may be some alignment considerations here for some implementations as DHCPv6 options are not aligned.]

## 6. Set of allowable options

The PVD container option MAY be used to encapsulate any allocated DHCPv6 options but MUST NOT be used to encapsulate another OPTION\_PVD option. [TODO: Should we add any other exclusions?]

## 7. Behaviour of DHCPv6 entities

This section describes role of DHCPv6 entities involved in requesting and receiving DHCPv6 configuration or prefix and address allocation.

### 7.1. Client and Requesting Router Behavior

DHCPv6 client or requesting router can request for configuration from provisioning domain in the following ways:

- o In the SOLICIT message it MAY include OPTION\_PVD\_ID requesting configuration for the specific PVD ID indicated in the OPTION\_PVD\_ID option. It can include multiple OPTION\_PVD\_ID options to indicate its preference for more than one provisioning domain. The PVD ID 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\_PVD option code to request configuration from all the PVDs that the DHCPv6 server can provide.

The client or requesting router parses OPTION\_PVD options in the response message. The Client or Requesting router MUST then include all or subset of the received OPTION\_PVD options in the REQUEST message so that it will be responsible for the configuration information selected.

If DHCPv6 client or requesting router receives OPTION\_PVD options but does not support PVD, it SHOULD ignore the received option(s).

### 7.2. Server and Delegating Router Behavior

If the Server or Delegating router supports PVD and it is configured to provide configuration data in one or more provisioning domains, it selects configuration for the PVD based allocation in the following way:

- o If OPTION\_PVD option code within OPTION\_ORO is not present in the request, it MUST NOT include provisioning domain based configuration. It MAY select configuration and prefix allocation from a default PVD defined.

- o If OPTION\_PVD\_ID is included, it selects information to be offered from that specific PVD if available.
- o If OPTION\_PVD option code within OPTION\_ORO is included, then based on its configuration and policy it MAY offer configuration from the available PVD(s).

When PVD information and configuration are selected for address and prefix allocation the server or delegating router responds with an ADVERTISE message after populating OPTION\_PVD.

If OPTION\_PVD is not included, then the server or delegating router MAY allocate the prefix and provide configuration as specified in [RFC3315] and [RFC3633] and MUST NOT include OPTION\_PVD option in the response.

If OPTION\_ORO option includes the OPTION\_PVD option code but the server or delegating router does not support PVD, then it SHOULD ignore the OPTION\_PVD and OPTION\_PVD\_ID options received.

If both client/requesting router and server/delegating router support PVD but cannot offer configuration with PVD for any other reason, it MUST respond to client/requesting router with appropriate status code as specified in [RFC3315] and [RFC3633].

## 8. Security Considerations

An attacker may attempt to modify the information provided inside the PVD container option. These attacks can easily be prevented by using the DHCPv6 AUTH option [RFC3315] that would detect any form of tampering with the DHCPv6 message contents.

A compromised DHCPv6 server or relay agent may insert configuration information related to PvDs it is not authorized to advertise. e.g. A coffee shop DHCPv6 server may provide configuration information purporting to be from an enterprise and may try to attract enterprise related traffic. The only real way to avoid this is that the PVD container contains embedded authentication and authorization information from the owner of the PVD. Then, this attack can be detected by the client by verifying the authentication and authorization information provided inside the PVD container option after verifying its trust towards the PVD owner (e.g. a certificate with a well-known/common trust anchor).

A compromised configuration source or an on-link attacker may try to capture advertised configuration information and replay it on a different link or at a future point in time. This can be avoided by including some replay protection mechanism such as a timestamp or a nonce inside the PVD container to ensure freshness of the provided

information.

## 9. IANA Considerations

This document defines three new DHCPv6 options to be allocated out of the registry at <http://www.iana.org/assignments/dhcpv6-parameters/>

OPTION\_PVD (TBA1)  
OPTION\_PVD\_ID (TBA2)  
OPTION\_PVD\_AUTH (TBA3)

## 10. Acknowledgements

The authors would like to thank the members of the MIF architecture design team for their comments that led to the creation of this draft.

## 11. Normative References

- [I-D.anipko-mif-mpvd-arch] Anipko, D., "Multiple Provisioning Domain Architecture", draft-anipko-mif-mpvd-arch-05 (work in progress), November 2013.
- [PVDIDS] Krishnan, S., Korhonen, J., Bhandari, S., and S. Gundavelli, "Identification of provisioning domains", draft-kkbg-mpvd-id-00 (work in progress), February 2014.
- [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.
- [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005.
- [RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, December 2005.

- [RFC6494] Gagliano, R., Krishnan, S., and A. Kukec, "Certificate Profile and Certificate Management for SEcure Neighbor Discovery (SEND)", RFC 6494, February 2012.
- [RFC6495] Gagliano, R., Krishnan, S., and A. Kukec, "Subject Key Identifier (SKI) SEcure Neighbor Discovery (SEND) Name Type Fields", RFC 6495, February 2012.

#### Authors' Addresses

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

Phone: +1 514 345 7900 x42871  
Email: suresh.krishnan@ericsson.com

Jouni Korhonen  
Broadcom Communications  
Porkkalankatu 24  
FIN-00180 Helsinki  
Finland

Email: jouni.nospam@gmail.com

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

Phone: +91 80 4426 0474  
Email: shwethab@cisco.com



Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 5, 2015

L. Xue  
D. Guo  
Huawei  
July 4, 2014

Dynamic Stateless GRE Tunnel  
draft-xue-dhc-dynamic-gre-02

Abstract

Generic Routing Encapsulation (GRE) is regarded as a popular encapsulation tunnel technology because of simpleness and easy implementation. When a node tries to encapsulate the user traffic in a GRE tunnel, it needs to first obtain the IP address of the destination node which need to decapsulate the GRE packets. In practice, the GRE tunnel destination IP address may be manually configured. This configuration introduces efficiency issues for operators, especially, in the scenarios where there are a large number entities need to deploy GRE tunnels. This work proposes an approach to configure the GRE information dynamiclly.

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 January 5, 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 . . . . .	3
3. Use Case . . . . .	3
4. Dynamic GRE Tunnel . . . . .	4
5. DHCP Option Definition . . . . .	6
6. IANA Considerations . . . . .	9
7. References . . . . .	9
7.1. Normative References . . . . .	9
7.2. Informative References . . . . .	9
Authors' Addresses . . . . .	9

## 1. Introduction

Generic Routing Encapsulation (GRE) [RFC1701][RFC2784] is widely deployed in the operators' networks. When a node tries to encapsulate the user traffic in a GRE tunnel, it needs to first obtain the IP address of the destination node which can decapsulate the GRE packets.

In practice, the decapsulation node IP address of GRE may be manually configured. This configuration introduces efficiency issues for operators, especially, in the scenarios where there are large amount of GRE tunnels needed. This work describes a case about large amount of GRE tunnels deployment required and proposes a solution which extends Dynamic Host Configuration Protocol (DHCP) so as to enable to configure the GRE destination node IP address dynamically.



## 2. Terminology

The following terminologies 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 CPE equipment is the box that a provider may distribute to the customers, which could be Home Gateway (HG), Cable Modem (CM), etc. When CPE is using DHCP to obtain network address, CPE is acting as "DHCP Client"

### Network Facing Equipment (NPE)

The NPE is the device to be deployed with the signalling and control functions. It is kind of Service Gateway.

### User Equipment (UE)

The UE is the a device of the customers, which could be PC or Mobile Phone.

### User Facing Equipment (UPE)

The UPE is the device to make forwarding decisions at the ingress of the provider network, which could be Cable Modem Termination Systems (CMTS). UPE is the "DHCP Server" or "DHCP relay agent" in DHCP framework.

### 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. Use Case

Wireless Local Area Network (WLAN) has emerged as an important access technology for service operators. Some operators deploy a large number of WTPs in the specific environments with the dense crowd. In this scenario, WTPs are preferred to be managed and controlled in a centralized location by AC. The traffic of WTPs are generally handled on the access router of the network, which is a different

node from AC. This architecture can avoid the overload for traffic management on the AC. This motivates the need for the WTP to support some tunnel encapsulation technologies to the Access Router. GRE is one of the preferred tunnel solution, because of simple and easy deployment reasons.

Currently, several tunnel mechanisms have been standardized, for example Layer Two Tunneling Protocol version 3 (L2TPv3) [RFC3931]. L2TPv3 supports IP/UDP encapsulation and fulfills the tunnel requirements. However, as a multi-layers encapsulation protocol, L2TPv3 has to carry multiple protocol headers per data packet. It is complicated and costly, mostly used for Virtual Private Network (VPN). Most CPE devices are too simple to be a L2TPv3 initiator.

An illustration of WLAN network is shown in figure 1. When WTP tries to encapsulate the user traffic in a GRE tunnel, it needs to first obtain the Access Router (AR) IP address. In practice, this IP address is usually deployed on WTP manually, which introduces efficiency issues for operators. Especially, a large number of WTPs are deployed with the dense crowd.

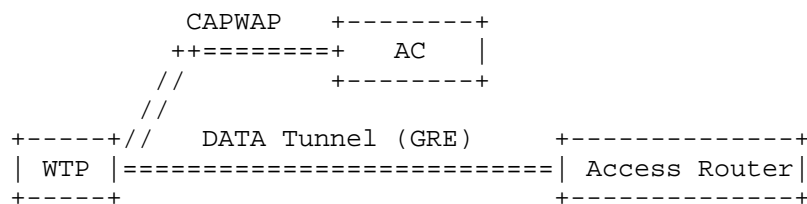


Figure 1: WLAN Illustration

#### 4. Dynamic GRE Tunnel

This work proposes an automatic solution which extends Dynamic Host Configuration Protocol (DHCP) so as to configure the GRE destination IP address. Due to successful IP address configuration, GRE tunnel can be setup dynamically.

Figure 2 illustrates the procedure for dynamic GRE tunnel in WLAN network. The WTP, AR in the picture are respectively the CPE and NPE.

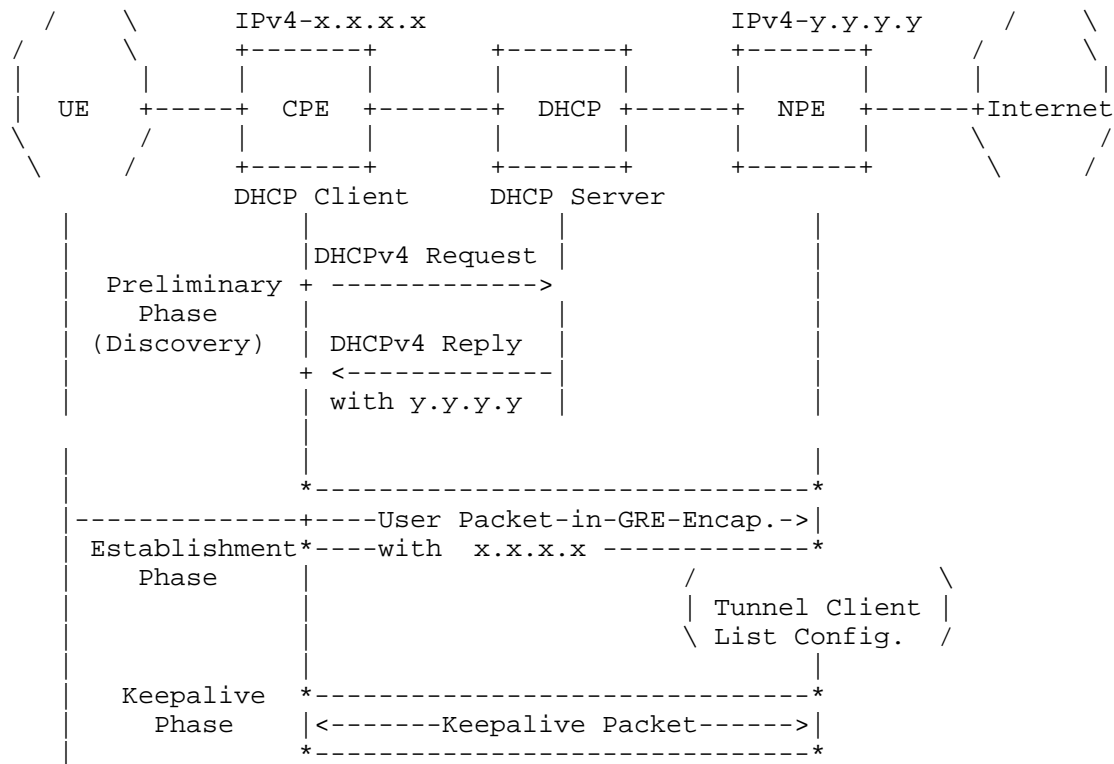


Figure 2: Dynamic GRE Tunnel in WLAN Network

At Preliminary Phase, the CPE as one endpoint of GRE tunnel, may get information of NPE by the DHCP approach. CPE sends the DHCP request to initiate a IPv4 address request. When a DHCP server replies the CPE request message, the NPE information can be carried in a DHCP reply message via DHCP options. Thus CPE configures the NPE address y.y.y.y and tunnel parameter of GRE tunnel, such as GRE key etc if they are carried in the DHCP message. For load sharing or single-point failure recovery purposes, a DHCP reply message may carry information of more than one NPEs.

Consequently, NPE can discover CPE via the received GRE encapsulated packet from CPE. Then the GRE tunnel information, such as IP address of CPE as source address is checked and restored as destination of GRE tunnel on NPE side. Generally, CPE can encapsulate UE's first packet with GRE, no matter data packet or control packet. For example, during a User Equipment (UE) subscriber attached initiates the DHCP procedure for an inner address, CPE should encapsulated this DHCP message via GRE.

When NPE receives the packet with GRE encapsulation, it should look up the outer source IP of the packet in its tunnel client list. If it is a new client, the NPE adds source IP into the tunnel client list, decapsulates GRE header and deals with the packet encapsulated by GRE.

There could be a keepalive mechanism for GRE tunnel between CPE and NPE. If there is neither keepalive packet nor data packet when the deployed timer expires, the NPE will tear down the tunnel and releases resource

## 5. DHCP Option Definition

As introduced above, The DHCPv4 GRE Discovery (GD) Option is defined, when CPE wants to obtain an NPE address in IPv4 network. This Option is carried in DHCPv4.

The DHCPv4 GD Option is structured as follows:

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								
+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-								
Code										Len										Ver		Reserved				Protocol Type													
+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-								
cont.										NPE address																													
+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-								
cont.																																							
-----+																																							

### DHCPv4 GRE Discovery Option

Code: TBD1

Len: The length of value field. If there are several instance for multiple NPE address considering redundancy, the length should be Len1 + Len2 + ... + Len n + Len of sub option in octets.

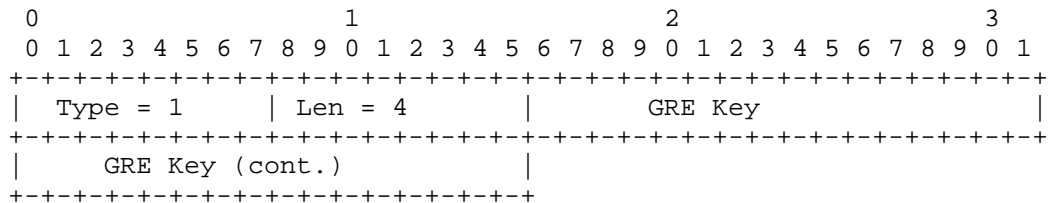
Ver: The Version Number field which is contained in GRE header, defined in [RFC2784].

Reserved: This field is reserved for future use. A receiver MUST discard a packet where this field is non-zero. These bits MUST be sent as zero and MUST be ignored on receipt.

Protocol Type: The Protocol Type field contains the protocol type of the payload packet. This field is defined in [RFC2784].

NPE Address: IPv4 address of NPE, the endpoint of GRE tunnel.

Sub-Option (Optional): DHCPv4 GRE Key Suboption is structured in TLV style shown as follows.



#### DHCPv4 GRE Key Suboption

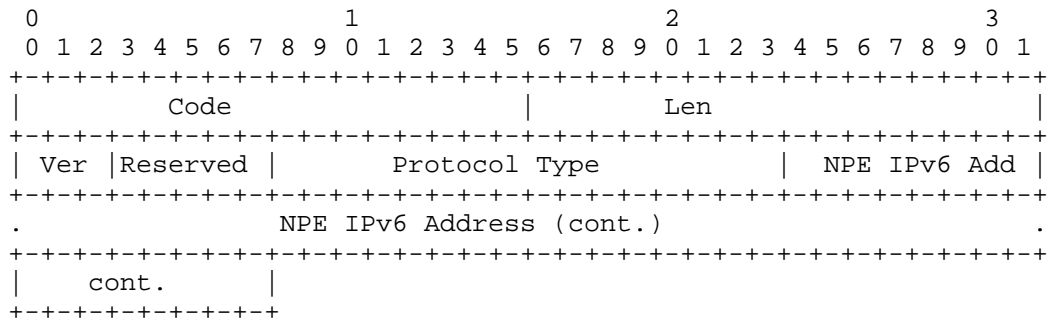
Based on requirement defined in [RFC2784] [RFC2890], GRE Key Suboption is used in this document to configure the complementary tunnel information. GRE Key is generated from [RFC2890]. If the client receives the GRE Key suboption, the key MUST be inserted into the GRE encapsulation header. It is used for identifying extra context information about the received payload. The payload packets without the correspondent GRE Key or with an unmatched GRE Key will be silently dropped.

Code 1 for GRE Key Suboption.

Len (1 octet): The total octets of the suboption value field.

Suboption Value : GRE Key according definition in [RFC2890].

The DHCPv6 GRE Discovery (GD) Option is mainly used when CPE wants to obtain an NPE address in IPv6 network. This option is carried in DHCPv6. According to [I-D.ietf-dhc-option-guidelines]The DHCPv6 GD Option is structured as follows.



## DHCPv6 GRE Discovery Option

Code: TBD2

Len: The length of the option value.

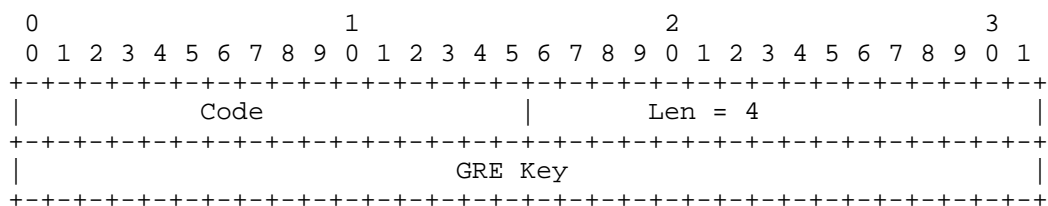
Ver: The Version Number field which is contained in GRE header, defined in [RFC2784].

Reserved: This field is reserved for future use. A receiver MUST discard a packet where this field is non-zero. These bits MUST be sent as zero and MUST be ignored on receipt.

Protocol Type: The Protocol Type field contains the protocol type of the payload packet. This field is defined in [RFC2784].

NPE Address: IPv6 address of NPE, the endpoint of GRE tunnel.

Based on requirement defined in [RFC2784] [RFC2890], DHCPv6 GRE Key Option is used in this document to configure the complementary tunnel information. Optionally, the DHCPv6 GRE Key Option is encapsulated in DHCPv6 GRE Discovery Option. It is structured in TLV style shown as follows.



## DHCPv6 GRE Key Option

Code 1 for DHCPv6 GRE Key Option: TBD3

Len (1 octet): The total octets of the option value field.

Option Value : GRE Key according definition in [RFC2890].

## 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.
- [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.
- [RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

### 7.2. Informative References

- [I-D.ietf-dhc-option-guidelines]  
Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", draft-ietf-dhc-option-guidelines-17 (work in progress), January 2014.

Authors' Addresses

Li Xue  
Huawei  
No.156 Beiqing Rd. Z-park, Shi-Chuang-Ke-Ji-Shi-Fan-Yuan, HaiDian District  
Beijing 100095  
China

Email: xueli@huawei.com

Dayong Guo  
Huawei  
No.156 Beiqing Rd. Z-park, Shi-Chuang-Ke-Ji-Shi-Fan-Yuan, HaiDian District  
Beijing 100095  
China

Email: guoseu@huawei.com