Internet Engineering Task Force   J. Bound
INTERNET DRAFT   Compaq
DHC Working Group   M. Carney
Obsoletes: draft-ietf-dhc-dhcpv6-18.txt   Sun Microsystems, Inc
                                                            C. Perkins
                                                 Nokia Research Center
                                                        R. Droms(ed.)
                                                        Cisco Systems
                                                         30 June 2001

Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
draft-ietf-dhc-dhcpv6-19.txt

Status of This Memo

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" [20], and can be used separately or concurrently with the latter to obtain configuration parameters.

Contents

1. Introduction

This document describes DHCP for IPv6 (DHCP), a UDP [18] client/server protocol designed to reduce the cost of management of IPv6 nodes in environments where network managers require more control over the allocation of IPv6 addresses and configuration of network stack parameters than that offered by "IPv6 Stateless Autoconfiguration" [20]. DHCP is a stateful counterpart to stateless autoconfiguration. Note that both stateful and stateless autoconfiguration can be used concurrently in the same environment, leveraging the strengths of both mechanisms in order to reduce the cost of ownership and management of network nodes.

DHCP reduces the cost of ownership by centralizing the management of network resources such as IP addresses, routing information, OS installation information, directory service information, and other such information on a few DHCP servers, rather than distributing such information in local configuration files among each network node. DHCP is designed to be easily extended to carry new configuration parameters through the addition of new DHCP "options" defined to carry this information.

Those readers familiar with DHCP for IPv4 [7] will find DHCP for IPv6 provides a superset of features, and benefits from the additional features of IPv6 and freedom from BOOTP [5]-backward compatibility constraints. For more information about the differences between DHCP for IPv6 and DHCP for IPv4, see Appendix A.

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 [3].

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

Related work in IPv6 that would best serve an implementor to study is the IPv6 Specification [6], the IPv6 Addressing Architecture [9], IPv6 Stateless Address Autoconfiguration [20], IPv6 Neighbor Discovery Processing [16], and Dynamic Updates to DNS [22]. These specifications enable DHCP to build upon the IPv6 work to provide both robust stateful autoconfiguration and autoregistration of DNS Host Names.

The IPv6 Specification provides the base architecture and design of IPv6. A key point for DHCP implementors to understand is that IPv6 requires that every link in the Internet have an MTU of 1280 octets or greater (in IPv4 the requirement is 68 octets). This means that a UDP packet of 536 octets will always pass through an internetwork (less 40 octets for the IPv6 header), as long as there are no IP options prior to the UDP header in the packet. But, IPv6 does not support fragmentation at routers, so that fragmentation takes place end-to-end between hosts. If a DHCP implementation needs to send a packet greater than 1500 octets it can either fragment the UDP packet into fragments of 1500 octets or less, or use Path MTU Discovery [11] to determine the size of the packet that will traverse a network path.

DHCP clients use Path MTU discovery when they have an address of sufficient scope to reach the DHCP server. If a DHCP client does not have such an address, that client MUST fragment its packets if the resultant message size is greater than the minimum 1280 octets.

Path MTU Discovery for IPv6 is supported for both UDP and TCP and can cause end-to-end fragmentation when the PMTU changes for a destination.

The IPv6 Addressing Architecture specification [9] 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. This means that a client can immediately use its link-local address and a well-known multicast address to begin communications to discover neighbors on the link. For instance, a client can send a Solicit message and locate a server or relay.

IPv6 Stateless Address Autoconfiguration [20] (Addrconf) specifies procedures by which a node may autoconfigure addresses based on router advertisements [16], 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 addrconf is a design requirement of DHCP (see Section 4).

IPv6 Neighbor Discovery [16] is the node discovery protocol in IPv6 which replaces and enhances functions of ARP [17]. To understand IPv6 and Addrconf it is strongly recommended that implementors understand IPv6 Neighbor Discovery.

Dynamic Updates to DNS [22] 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. Design Goals

- DHCP is a mechanism rather than a policy. Network administrators set their administrative policies through the configuration parameters they place upon the DHCP servers in the DHCP domain they're managing. DHCP is simply used to deliver parameters according to that policy to each of the DHCP clients within the domain.

- DHCP is compatible with IPv6 stateless autoconf [20].

- DHCP does not require manual configuration of network parameters on DHCP clients, except in cases where such configuration is needed for security reasons. A node configuring itself using DHCP should require no user intervention.

- DHCP does not require a server on each link. To allow for scale and economy, DHCP must work across DHCP relays.

- DHCP coexists with statically configured, non-participating nodes and with existing network protocol implementations.

- DHCP clients can operate on a link without IPv6 routers present.

- DHCP will provide the ability to renumber network(s) when required by network administrators [4].

- A DHCP client can make multiple, different requests for configuration parameters when necessary from one or more DHCP servers at any time.

- DHCP will contain the appropriate time out and retransmission mechanisms to efficiently operate in environments with high latency and low bandwidth characteristics.

5. Non-Goals

This specification explicitly does not cover the following:

- Specification of a DHCP server to server protocol.

- How a DHCP server stores its DHCP data.

- How to manage a DHCP domain or DHCP server.

- How a DHCP relay is configured or what sort of information it may log.

6. Terminology

6.1. IPv6 Terminology

IPv6 terminology relevant to this specification from the IPv6 Protocol [6], IPv6 Addressing Architecture [9], and IPv6 Stateless Address Autoconfiguration [20] is included below.

address — An IP layer identifier for an interface or a set of interfaces.

unicast address — An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that 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.

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 IP address having link-only scope, indicated by having the prefix (FE80::0000/64), that can be used to reach neighboring nodes attached to the same link. Every interface has a link-local address.

message	A unit of data carried in a packet, exchanged between DHCP agents and clients.
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 address 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.

6.2. DHCP Terminology

Terminology specific to DHCP can be found below.

abort status	A status value returned to the application that has invoked a DHCP client operation, indicating anything other than success.
agent address	The address of a neighboring DHCP Agent on the same link as the DHCP client.
binding	A binding (or, client binding) is a group of server data records containing the server's information about the addresses in an IA and any other configuration information assigned to the client. A binding is indexed by the tuple <DUID, IAID>.
DHCP	Dynamic Host Configuration Protocol for IPv6. The terms DHCPv4 and DHCPv6 are used only in contexts where it is necessary to avoid ambiguity.
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.

DHCP client (or client) A node that initiates requests on a link to obtain configuration parameters from one or more DHCP servers.

DHCP domain A set of links managed by DHCP and operated by a single administrative entity.

DHCP server (or server) A server is a node that responds to requests from clients, and may or may not be on the same link as the client(s).

DHCP relay (or relay) A node that acts as an intermediary to deliver DHCP messages between clients and servers, and is on the same link as a client.

DHCP agent (or agent) Either a DHCP server on the same link as a client, or a DHCP relay.

DUID A DHCP unique identifier for a client.

Identity association (IA) A collection of addresses assigned to a client. Each IA has an associated IAID. An IA may have 0 or more addresses associated with it.

Identity association identifier (IAID) An identifier for an IA, chosen by the client. Each IA has an IAID, which is chosen to be unique among all IAIDs for IAs belonging to that client.

transaction-ID An unsigned integer to match responses with replies initiated either by a client or server.

7. DHCP Constants

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

7.1. Multicast Addresses

DHCP makes use of the following multicast addresses:

All DHCP Agents address: FF02::1:2 This link-scoped multicast address is used by clients to communicate with the on-link agent(s) when they do not know those agents' link-local address(es). All agents (servers and relays) are members of this multicast group.

All DHCP Servers address: FF05::1:3 This site-scoped multicast address is used by clients or relays to communicate with server(s), either because they want to send messages to all servers or because they do not know the server(s) unicast address(es). Note that in order for a client to use this address, it must have an address of sufficient scope to be reachable by the server(s). All servers within the site are members of this multicast group.

7.2. UDP ports

DHCP uses the following destination UDP [18] port numbers. While source ports MAY be arbitrary, client implementations SHOULD permit their specification through a local configuration parameter to facilitate the use of DHCP through firewalls.

546 Client port. Used by servers as the destination port for messages sent to clients and relays. Used by relay agents as the destination port for messages sent to clients.

547 Agent port. Used as the destination port by clients for messages sent to agents. Used as the destination port by relays for messages sent to servers.

7.3. DHCP message types

DHCP defines the following message types. More detail on these message types can be found in Section 9. Message types 0 and TBD--255 are reserved and MUST be silently ignored. The message code for each message type is shown with the message name.

SOLICIT (1) The DHCP Solicit (or Solicit) message is used by clients to locate servers.

ADVERTISE (2) The DHCP Advertise (or Advertise) message is used by servers responding to Solicits.

REQUEST (3) The DHCP Request (or Request) message is used by clients to request configuration parameters from servers.

CONFIRM (4) The DHCP Confirm (or Confirm) message is used

by clients to confirm that the addresses
assigned to an IA and the lifetimes for
those addresses, as well as the current

configuration parameters assigned by the server to the client are still valid.

RENEW (5) The DHCP Renew (or Renew) message is used by clients to obtain the addresses assigned to an IA and the lifetimes for those addresses, as well as the current configuration parameters assigned by the server to the client. A client sends a Renew message to the server that originally assigned the IA when the lease on an IA is about to expire.

REBIND (6) The DHCP Rebind (or Rebind) message is used by clients to obtain the addresses assigned to an IA and the lifetimes for those addresses, as well as the current configuration parameters assigned by the server to the client. A clients sends a Rebind message to all available DHCP servers when the lease on an IA is about to expire.

REPLY (7) The DHCP Reply (or Reply) message is used by servers responding to Request, Confirm, Renew, Rebind, Release and Decline messages. In the case of responding to a Request, Confirm, Renew or Rebind message, the Reply contains configuration parameters destined for the client.

RELEASE (8) The DHCP Release (or Release) message is used by clients to return one or more IP addresses to servers.

DECLINE (9) The DHCP Decline (or Decline) message is used by clients to indicate that the client has determined that one or more addresses in an IA are already in use on the link to which the client is connected.

RECONFIG-INIT (10) The DHCP Reconfigure-init (or Reconfigure-init) message is sent by server(s) to inform client(s) that the server(s) has new or updated configuration parameters, and that the client(s) are to initiate a Request/Reply transaction with the server(s) in order to receive the updated information.

RELAY-FORW (11)   The DHCP Relay-forward (or Relay-forward) message is used by relays to forward client messages to servers. The client message is encapsulated in an option in the Relay-forward message.

RELAY-REPL (12) The DHCP Relay-reply (or Relay-reply) message is used by servers to send messages to clients through a relay. The server encapsulates the client message as an option in the Relay-reply message, which the relay extracts and forwards to the client.

7.4. Error Values

This section describes error values exchanged between DHCP implementations.

7.4.1. Generic Error Values

The following symbolic names are used between client and server implementations to convey error conditions. The following table contains the actual numeric values for each name. Note that the numeric values do not start at 1, nor are they consecutive. The errors are organized in logical groups.

Error Name	Error ID	Description
Success	00	Success
UnspecFail	16	Failure, reason unspecified
AuthFailed	17	Authentication failed or nonexistent
PoorlyFormed	18	Poorly formed message
Unavail	19	Addresses unavailable

7.4.2. Server-specific Error Values

The following symbolic names are used by server implementations to convey error conditions to clients. The following table contains the actual numeric values for each name.

Error Name	Error ID	Description
NoBinding	20	Client record (binding) unavailable
ConfNoMatch	21	Client record Confirm not match IA
RenwNoMatch	22	Client record Renew not match IA
RebdNoMatch	23	Client record Rebind not match IA
InvalidSource	24	Invalid Client IP address
NoServer	25	Relay cannot find Server Address
ICMPError	64	Server unreachable (ICMP error)

7.5. Configuration Variables

This section presents a table of client and server configuration variables and the default or initial values for these variables. The client-specific variables MAY be configured on the server and MAY be delivered to the client through the "DHCP Retransmission Parameter Option" in a Reply message.

Parameter	Default	Description
MIN_SOL_DELAY	1	MIN (secs) to delay 1st mesg
MAX_SOL_DELAY	5	MAX (secs) to delay 1st mesg
ADV_MSG_TIMEOUT	500	SOL Retrans timer (msecs)
ADV_MSG_MAX	30	MAX timer value (secs)
SOL_MAX_ATTEMPTS	-1	MAX attempts (-1 = infinite)
REP_MSG_TIMEOUT	250	Retrans timer (msecs) for Reply
QRY_MSG_ATTEMPTS	10	MAX Request/Confirm/Renew/Rebind attempts
REL_MSG_ATTEMPTS	5	MAX Release/Decline attempts
RECREP_MSG_TIMEOUT	2000	Retrans timer (msecs)
REC_MSG_ATTEMPTS	10	Reconfigure attempts
REC_THRESHOLD	100	% of required clients
SRVR_PREF_WAIT	2	Advertise Collect timer (secs)

8. Overview

This section provides a general overview of the interaction between the functional entities of DHCP. The overview is organized as a series of questions and answers. Details of DHCP such as message formats and retransmissions can be found in later sections of this document.

8.1. How does a node know to use DHCP?

An unconfigured node determines that it is to use DHCP for configuration of an interface by detecting the presence (or absence) of routers on the link. If router(s) are present, the node examines router advertisements to determine if DHCP should be used to configure the interface. If there are no routers present, then the node MUST use DHCP to configure the interface. Details of this process can be found in neighbor discovery [16] and stateless autoconfiguration [20].

8.2. What if the client and server(s) are on different links?

Use of DHCP in such environments requires one or more DHCP relays be set up on the client's link, because a client may only have a link-local address. Relays receive messages from the client and forward them to some set of servers within the DHCP domain. The client message is forwarded verbatim as an option in the message from the relay to the server. A relay will include one of its own addresses (of sufficient scope) from the interface on the same link as the client, as well as the prefix length of that address, in its message to the server. Servers receiving the forwarded traffic use this information to aid in selecting configuration parameters appropriate to the client's link.

Servers use relays to forward messages to clients. The message intended for the client is carried as an option in the message to the relay. The relay extracts the message from the option and forwards it to the client. Servers use the relay's address as the destination to forward client-destined messages for final delivery by the relay.

Relays forward client messages to servers using some combination of the All DHCP Servers site-local multicast address, some other (perhaps a combination) of site-local multicast addresses set up within the DHCP domain to include the servers in that domain, or a list of unicast addresses for servers. The network administrator makes relay configuration decisions based upon the topological requirements (scope) of the DHCP domain they are managing. Note that if the DHCP domain spans more than the site-local scope, then the relays MUST be configured with global addresses for the client's link so as to be reachable by servers outside the relays' site-local environment.

8.3. How does a client request configuration parameters from servers?

To request configuration parameters, the client forms a Request message, and sends it to the server either directly (the server is on the same link as the client) or indirectly (through the on-link relay). The client MAY include a Option Request Option 18.4 (ORO) along with other options to request specific information from the server. Note that the client MAY form multiple Request messages and send each of them to different servers to request potentially different information (perhaps based upon what was advertised) in order to satisfy its needs. As a client's needs may change over time (perhaps based upon an application's requirements), the client may form additional Request messages to request additional information as it is needed.

The server(s) respond with Reply messages containing the requested configuration parameters, which can include status information regarding the information requested by the client. The Reply MAY also include additional information.

8.4. How do clients and servers identify and manage addresses?

Servers and clients manage addresses in groups called "identity associations." Each identity association (IA) is identified using a unique identifier. An identity association may contain one or more IPv6 addresses. DHCP servers assign addresses to identity associations. DHCP clients use the addresses in an identity association to configure interfaces. There is always at least one identity association per interface that a client wishes to configure. Each address in an IA has its own preferred and valid lifetime. Over time, the server may change the characteristics of the addresses in an IA; for example, by changing the preferred or valid lifetime for an address in the IA. The server may also add or delete addresses from an IA; for example, deleting old addresses and adding new addresses to renumber a client. A client can request the current list of addresses assigned to an IA from a server through an exchange of protocol messages.

8.5. Can a client release its assigned addresses before the lease expires?

A client forms a Release message, including options identifying the IA to be released. The client sends the Release to the server which assigned the addresses to the client initially. If that server cannot be reached after a certain number of attempts (see section 7.5), the client can abandon the Release attempt. In this case, the address(es) in the IA will be reclaimed by the server(s) when the lifetimes on the addresses expire.

8.6. What if the client determines one or more of its assigned addresses are already being used by another client?

If the client determines through a mechanism like Duplicate Address Detection [20] that the address it was assigned by the server is already in use by another client, the client will send a Decline message to the server.

8.7. How are clients notified of server configuration changes?

There are two possibilities. Either the clients discover the new information when they revisit the server(s) to request additional configuration information/extend the lifetime on an address. or through a server-initiated event known as a reconfigure event.

The reconfiguration feature of DHCP offers network administrators the opportunity to update configuration information on DHCP clients whenever necessary. To signal the need for client reconfiguration, the server will unicast a Reconfigure-init message to each client individually. A Reconfigure-init is a trigger which will cause the client(s) to initiate a standard Request/Reply exchange with the server in order to acquire the new or updated addresses.

9. Message Formats

Each DHCP message has an identical fixed format header; some messages also allow a variable format area for options. Not all fields in the header are used in every message. In this section, every field is described for every message and fields that are not used in a message are marked as "unused". All unused fields in a message MUST be transmitted as zeroes and ignored by the receiver of the message.

The DHCP message header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    msg-type   |  preference   |         transaction-ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   client-link-local-address                   |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         server-address                        |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                            options                            .
|                          (variable)                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

9.1. DHCP Solicit Message Format

msg-type	SOLICIT
preference	(unused) MUST be 0
transaction-ID	An unsigned integer generated by the client used to identify this Solicit message.
client-link-local-address	The link-local address of the interface for which the client is using DHCP.
server-address	(unused) MUST be 0
options	See section 18.

9.2. DHCP Advertise Message Format

msg-type	ADVERTISE
preference	An unsigned integer indicating a server's willingness to provide

service to the client.

transaction-ID	An unsigned integer used to identify this Advertise message. Copied from the client's Solicit message.
client-link-local-address	The IP link-local address of the client interface from which the client issued the Solicit message.
server-address	The IP address of the server that generated this message. If the DHCP domain crosses site boundaries, then this address MUST be globally-scoped.
options	See section 18.

9.3. DHCP Request Message Format

msg-type	REQUEST
preference	(unused) MUST be 0
transaction-ID	An unsigned integer generated by the client used to identify this Request message.
client-link-local-address	The link-local address of the client interface from which the client will issue the Request message.
server-address	The IP address of the server to which the this message is directed, copied from an Advertise message.
options	See section 18.

9.4. DHCP Confirm Message Format

msg-type	CONFIRM
preference	(unused) MUST be 0
transaction-ID	An unsigned integer generated by the client used to identify this Confirm message.
client-link-local-address	The link-local address of the client interface from which the client will

issue the Confirm message.

server-address MUST be zero.

options See section 18.

9.5. DHCP Renew Message Format

msg-type RENEW

preference (unused) MUST be 0

transaction-ID An unsigned integer generated by the client used to identify this Renew message.

client-link-local-address The link-local address of the client interface from which the client will issue the Renew message.

server-address The IP address of the server to which this Renew message is directed, which MUST be the address of the server from which the IAs in this message were originally assigned.

options See section 18.

9.6. DHCP Rebind Message Format

msg-type REBIND

preference (unused) MUST be 0

transaction-ID An unsigned integer generated by the client used to identify this Rebind message.

client-link-local-address The link-local address of the client interface from which the client will issue the Rebind message.

server-address MUST be zero.

options See section 18.

9.7. DHCP Reply Message Format

msg-type REPLY

preference An unsigned integer indicating a

server's willingness to provide
service to the client.

transaction-ID	An unsigned integer used to identify this Reply message. Copied from the client's Request, Confirm, Renew or Rebind message.
client-link-local-address	The link-local address of the interface for which the client is using DHCP.
server-address	The IP address of the server. If the DHCP domain crosses site boundaries, then this address MUST be globally-scoped.
options	See section 18.

9.8. DHCP Release Message Format

msg-type	RELEASE
preference	(unused) MUST be 0
transaction-ID	An unsigned integer generated by the client used to identify this Release message.
client-link-local-address	The client's link-local address for the interface from which the client will send the Release message.
server-address	The IP address of the server that assigned the IA.
options	See section 18.

9.9. DHCP Decline Message Format

msg-type	DECLINE
preference	(unused) MUST be 0
transaction-ID	An unsigned integer generated by the client used to identify this Decline message.
client-link-local-address	The client's link-local address for the interface from which the client

will send the Decline message.

server-address — The IP address of the server that assigned the addresses.

options See section 18.

9.10. DHCP Reconfigure-init Message Format

msg-type RECONFIG-INIT

preference (unused) MUST be 0

transaction-ID (unused) MUST be 0

client-link-local-address (unused) MUST be 0

server-address The IP address of the DHCP server issuing the Reconfigure-init message. MUST be of sufficient scope to be reachable by all clients.

options See section 18.

10. Relay messages

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

10.1. Relay-forward message

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    msg-type   | prefix length |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                                                               |
|                         relay-address                         |
|                                                               |
|                               |-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|            options (variable number and length)   ....        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

msg-type RELAY-FORW

prefix-length The length of the prefix in the address in the "relay-address" field.

relay-address An address assigned to the interface through which the message from the client was received.

options MUST include a "Client message option"; see section 18.5.

10.2. Relay-reply message

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    msg-type   | prefix length |                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
    |                                                               |
    |                         relay-address                         |
    |                                                               |
    |                               |-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
    |            options (variable number and length)   ....        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

msg-type   RELAY-REPL

prefix-length   The length of the prefix in the address in the "relay-address" field.

relay-address   An address identifying the interface through which the message from the server should be forwarded; copied from the "relay-forward" message.

options   MUST include a "Server message option"; see section 18.6.

11. DHCP unique identifier (DUID)

Each DHCP client has a DUID. DHCP servers use DUIDs to identify clients for the selection of configuration parameters and in the association of IAs with clients. See section 18.2 for the representation of a DUID in a DHCP message.

DISCUSSION:

The syntax, rules for selecting and requirements for gloabl uniqueness in DUIDs are TBD.

The DUID is carried in an option because it may be variable length and because it is not required in all DHCP options (e.g., messages sent by servers need not include a DUID).

12. Identity association

An "identity-association" (IA) is a construct through which a server and a client can identify, group and manage IPv6 addresses. Each IA consists of an IAID and a list of associated IPv6 addresses (the list may be empty). A client associates an IA with one of its interfaces and uses the IA to obtain IPv6 addresses for that interface from a server.

See section 18.3 for the representation of an IA in a DHCP message.

13. DHCP Server Solicitation

This section describes how a client locates servers. The behavior of client and server implementations is discussed, along with the messages they use.

13.1. Solicit Message Validation

Clients MUST silently discard any received Solicit messages.

Agents MUST silently discard any received Solicit messages if the "client-link-local-address" field does not contain a valid link-local address.

13.2. Advertise Message Validation

Servers MUST discard any received Advertise messages.

Clients MUST discard any Advertise messages that meet any of the following criteria:

- o The "Transaction-ID" field value does not match the value the client used in its Solicit message.

- o The "client-link-local-address" field value does not match the link-local address of the interface upon which the client sent the Solicit message.

13.3. Client Behavior

Clients use the Solicit message to discover DHCP servers configured to serve addresses on the link to which the client is attached.

13.3.1. Creation and sending of the Solicit message

The client sets the "msg-type" field to SOLICIT, and places the link-local address of the interface it wishes to configure in the "client-link-local-address" field.

The client generates a transaction ID inserts this value in the "transaction-ID" field.

The client includes a DUID option to identify itself to the server. The client MUST include options for any IAs to which the client is expecting to have the server assign addresses. Because the client does not have any IAs with addresses when sending a Solicit message, all of the IAs MUST be empty. The client MAY include an Option Request Option in the Solicit message. The client MUST NOT include any other options except those specifically allowed as defined by specific options.

The client sends the Solicit message to the All DHCP Agents multicast address, destination port 547. The source port selection can be arbitrary, although it SHOULD be possible using a client configuration facility to set a specific source port value.

13.3.2. Time out and retransmission of Solicit Messages

The client's first Solicit message on the interface MUST be delayed by a random amount of time between the interval of MIN_SOL_DELAY and MAX_SOL_DELAY. This random delay desynchronizes clients which start at the same time (e.g., after a power outage).

The client waits ADV_MSG_TIMEOUT, collecting Advertise messages. If no Advertise messages are received, the client retransmits the Solicit, and doubles the ADV_MSG_TIMEOUT value. This process continues until either one or more Advertise messages are received or ADV_MSG_TIMEOUT reaches the ADV_MSG_MAX value. Thereafter, Solicits are retransmitted every ADV_MSG_MAX until SOL_MAX_ATTEMPTS have been made, at which time the client MAY choose to stop trying to DHCP configure the interface. An event external to DHCP is required to restart the DHCP configuration process. A DHCP client MAY, alternatively, choose to continue sending Solicit messages at the ADV_MSG_MAX interval.

Default and initial values for MIN_SOL_DELAY, MAX_SOL_DELAY, ADV_MSG_TIMEOUT, AND ADV_MSG_MAX are documented in section 7.5.

13.3.3. Receipt of Advertise messages

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

- Those Advertise messages with the highest server preference value (see section 19.4) 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. For example, one server may be advertising the availability of IP addresses which have an address scope of interest to the client.

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. Depending on the requirements of the client's invoking user, the client MAY initiate a configuration exchange with the server(s) immediately, or MAY defer this exchange until later.

If the client needs to select an alternate server in the case that a chosen server does not respond, the client chooses the server with the next highest preference value.

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.

13.4. Server Behavior

For this discussion, the server is assumed to have been configured in an implementation specific manner. This configuration is assumed to contain all network topology information for the DHCP domain, as well as any necessary authentication information.

13.4.1. Receipt of Solicit messages

If the server receives a Solicit message, the client must be on the same link as the server. If the server receives a Relay-forward message containing a Solicit message, the client must be on the link to which the prefix identified by the "relay-address" and "prefix-length" fields in the Relay-forward message is assigned. The server records the "relay-address" field from the Relay-forward message and extracts the solicit message from the "client-message" option.

If administrative policy permits the server to respond to a client on that link, the server will generate and send an Advertise message to the client.

13.4.2. Creation and sending of Advertise messages

The server sets the "msg-type" field to ADVERTISE and copies the values of the following fields from the client's Solicit to the Advertise message:

o transaction-ID

o client-link-local-address

The server places one of its IP addresses (determined through administrator setting) in the "server-address" field of the Advertise message. The server sets the "preference" field according to its configuration information. See section 20.3 for a description of server preference.

The server MUST include options to the Advertise message containing any addresses that would be assigned to IAs contained in the Solicit message from the client. The server MAY include other options the server will return to the client in a subsequent Reply message. The information in these options will be used by the client in the selection of a server if the client receives more than one Advertise message.

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 "server-message" option. The server unicasts the Relay-reply message to the address in the "relay-address" field from the Relay-forward message.

If the Solicit message was received directly by the server, the server unicasts the Advertise message directly to the client using the "client-link-local-address" field value as the destination address. The Advertise message MUST be unicast through the interface on which the Solicit message was received.

14. 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 or when requested to do so by the application layer.

The client uses the following messages to initiate a configuration event:

Request   Obtain initial configuration information (from a server identified in a previously received Advertise message) when the client has no assigned addresses

Confirm   Confirm the validity of assigned addresses and other configuration changes through the server from which the configuration information was obtained when the client's assigned addresses may not be valid; for example, when the client reboots or loses its connection to a link

Renew   Extend the lease on an IA through the server that originally assigned the IA

Rebind   Extend the lease on an IA through any server willing to extend the lease

Release Release the lease on an IA and release all of the addresses contained in the IA,

Decline Decline the assignment of one or more addresses in an IA.

A client uses the Release/Reply message exchange to indicate to the DHCP server that the client will no longer be using the addresses in the released IA.

A client uses the Decline/Reply message exchange to indicate to the DHCP server that the client has detected that one or more addresses assigned by the server is already in use on the client's link.

14.1. Client Message Validation

Clients MUST silently discard any received client messages (Request, Confirm, Renew, Rebind, Release or Decline messages).

Agents MUST discard any received client messages in which the "client-link-local-address" field does not contain a valid link-local address.

Servers MUST discard any received client messages in which the "options" field contains an authentication option, and the server cannot successfully authenticate the client.

Servers MUST discard any received Request, Renew, Release or Decline message in which the "server-address" field value does not match any of the server's addresses.

14.2. Server Message Validation

Servers MUST silently discard any received server messages (Advertise, Reply or Reconfigure-init messages).

Clients MUST discard any server messages that meet any of the following criteria:

- o The "transaction-ID" field value in the server message does not match the value the client used in its Request or Release message.

- o The "client-link-local-address" field value in the server message does not match the link-local address of the interface from which the client sent in its Request, Confirm, Renew, Rebind, Release or Decline message.

- o The server message contains an authentication option, and the client's attempt to authenticate the message fails.

Relays MUST discard any Relay-reply message in which the "client-link-local-address" in the encapsulated Reply message does not contain a valid link-local address.

14.3. Client Behavior

A client will use Request, Confirm, Renew and Rebind messages to acquire and confirm the validity of configuration information. A client may initiate such an exchange automatically in order to acquire the necessary network parameters to communicate with nodes off-link. The client uses the server address information from previous Advertise message(s) for use in constructing Request and Renew message(s). Note that a client may request configuration information from one or more servers at any time.

A client uses the Release message in the management of IAs when the client has been instructed to release the IA prior to the IA expiration time since it is no longer needed.

A client uses the Decline message when the client has determined through DAD or some other method that one or more of the addresses assigned by the server in the IA is already in use by a different client.

14.3.1. Creation and sending of Request messages

If a client has no valid IPv6 addresses of sufficient scope to communicate with a DHCP server, it may send a Request message to obtain new addresses. The client includes one or more IAs in the Request message, to which the server assigns new addresses. The server then returns IA(s) to the client in a Reply message.

The client sets the "msg-type" field to REQUEST, and places the link-local address of the interface it wishes to acquire configuration information for in the "client-link-local-address" field.

The client generates a transaction ID inserts this value in the "transaction-ID" field.

The client places the address of the destination server in the "server-address" field.

The client adds a DUID option to identify itself to the server. The client adds any other appropppriate options, including one or more IA options (if the client is requesting that the server assign it some network addresses). The list of addresses in each included IA MUST be empty. If the client is not requesting that the server assign it any addresses, the client omits the IA option.

The client sends the Request message to the All DHCP Agents multicast address, destination port 547. The source port selection can be arbitrary, although it SHOULD be possible using a client configuration facility to set a specific source port value.

The server will respond to the Request message with a Reply message. If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Request with the same transaction-ID, and doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or REQUEST_MSG_ATTEMPTS unsuccessful attempts have been made, at which time the client MUST abort the configuration attempt. The client SHOULD report the abort status to the application layer.

Default and initial values for REP_MSG_TIMEOUT and REQ_MSG_ATTEMPTS are documented in section 7.5.

14.3.2. Creation and sending of Confirm messages

Whenever a client may have moved to a new link, its IPv6 addresses may no longer be valid. Examples of times when a client may have moved to a new link include:

- o The client reboots
- o The client is physically disconnected from a wired connection
- o The client returns from sleep mode
- o The client using a wireless technology changes cells

In any situation when a client may have moved to a new link, the client MUST initiate a Confirm/Reply message exchange. The client includes any IAs, along with the addresses associated with those IAs, in its Confirm message. Any responding servers will indicate the acceptability of the addresses with the status in the IA it returns to the client.

The client sets the "msg-type" field to CONFIRM, and places the link-local address of the interface it wishes to acquire configuration information for in the "client-link-local-address" field.

The client generates a transaction ID inserts this value in the "transaction-ID" field.

The client sets the "server-address" field to 0.

The client adds a DUID option to identify itself to the server. The client adds any appropriate options, including one or more IA options (if the client is requesting that the server confirm the validity of some network addresses). If the client does include any IA options, it MUST include the list of addresses the client currently has associated with that IA.

The client sends the Confirm message to the All DHCP Agents multicast address, destination port 547. The source port selection can be arbitrary, although it SHOULD be possible using a client configuration facility to set a specific source port value.

Servers will respond to the Confirm message with a Reply message. If no Confirm message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Confirm with the same transaction-ID, and doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or QRY_MSG_ATTEMPTS unsuccessful attempts have been made, at which time the client MUST abort the configuration attempt. The client SHOULD report the abort status to the application layer.

Default and initial values for REP_MSG_TIMEOUT and QRY_MSG_ATTEMPTS are documented in section 7.5.

If the client receives no response to its Confirm message, it MAY restart the configuration process by locating a DHCP server with an Advertise message and sending a Request to that server, as described in section 14.3.1.

14.3.3. Creation and sending of Renew messages

IPv6 addresses assigned to a client through an IA use the same preferred and valid lifetimes as IPv6 addresses obtained through stateless autoconfiguration. The server assigns preferred and valid lifetimes to the IPv6 addresses it assigns to an IA. To extend those lifetimes, the client sends a Request to the server containing an "IA option" for the IA and its associated addresses. The server determines new lifetimes for the addresses in the IA according to the server's administrative configuration. The server may also add new addresses to the IA. The server 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. If the server does not assign an explicit value to T1 or T2 for an IA, T1 defaults to 0.5 times the shortest preferred lifetime of any address assigned to the IA and T2 defaults to 0.875 times the shortest preferred lifetime of any address assigned to the IA.

At time T1 for an IA, the client initiates a Request/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 Request message. The client sends this Request message to the All DHCP Agents multicast address.

The client sets the "msg-type" field to RENEW, and places the link-local address of the interface it wishes to acquire configuration information for in the "client-link-local-address" field.

The client generates a transaction ID inserts this value in the "transaction-ID" field.

The client places the address of the destination server in the "server-address" field.

The client adds a DUID option to identify itself to the server. The client adds any appropriate options, including one or more IA options (if the client is requesting that the server extend the lease on some IAs; note that the client may check the status of other configuration parameters without asking for lease extensions). If the client does include any IA options, it MUST include the list of addresses the client currently has associated with that IA.

The client sends the Renew message to the All DHCP Agents multicast address, destination port 547. The source port selection can be arbitrary, although it SHOULD be possible using a client configuration facility to set a specific source port value.

The server will respond to the Renew message with a Reply message. If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Renew with the same transaction-ID, and doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or until time T2 is reached (see section 14.3.4).

Default and initial values for REP_MSG_TIMEOUT are documented in section 7.5.

14.3.4. Creation and sending 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. The client includes an IA option with all addresses currently assigned to the IA in its Rebind message. The client sends this message to the All DHCP Agents multicast address.

The client sets the "msg-type" field to REBIND, and places the link-local address of the interface it wishes to acquire configuration information for in the "client-link-local-address" field.

The client generates a transaction ID inserts this value in the "transaction-ID" field.

The client sets the "server-address" field to 0.

The client adds a DUID option to identify itself to the server. The client adds any appropriate options, including one or more IA options. If the client does include any IA options (if the client is requesting that the server extend the lease on some IAs; note that the client may check the status of other configuration parameters without asking for lease extensions), it MUST include the list of addresses the client currently has associated with that IA.

The client sends the Rebind message to the All DHCP Agents multicast address, destination port 547. The source port selection can be arbitrary, although it SHOULD be possible using a client configuration facility to set a specific source port value.

The server will respond to the Rebind message with a Reply message. If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Rebind with the same transaction-ID, and doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received.

Default and initial values for REP_MSG_TIMEOUT are documented in section 7.5.

The client has several alternatives to choose from if it receives no response to its Rebind message.

- When the lease on the IA expires, 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

- Some addresses in the IA may have lifetimes that extend beyond the lease of the IA, so the client may choose to continue to use those addresses; once all of the addresses have expired, the client may choose to locate a new DHCP 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

14.3.5. Receipt of Reply message in response to a Request, Confirm, Renew or Rebind message

Upon the receipt of a valid Reply message in response to a Request, Confirm, Renew or Rebind message, the client extracts the configuration information contained in the Reply. If the "status" field contains a non-zero value, the client reports the error status to the application layer.

The client records the T1 and T2 times for each IA in the Reply message. The client records any addresses included with IAs in the Reply message. The client updates the preferred and valid lifetimes for the addresses in the IA from the lifetime information in the IA option. The client leaves any addresses that the client has associated with the IA that are not included in the IA option unchanged.

Management of the specific configuration information is detailed in the definition of each option, in section 18.

When the client receives an Unavail error status in an IA from the server for a Request message the client will have to find a new server to create an IA.

When the client receives a NoBinding error status in an IA from the server for a Confirm message the client can assume it needs to send a Request to reestablish an IA with the server.

When the client receives a Conf_NoMatch error status in an IA from the server for a Confirm message the client can send a Renew message to the server to extend the lease for the addresses.

When the client receives a NoBinding error status in an IA from the server for a Renew message the client can assume it needs to send a Request to reestablish an IA with the server.

When the client receives a Renw_NoMatch error status in an IA from the server for a Renew message the client can assume it needs to send a Request to reestablish an IA with the server.

When the client receives an Unavail error status in an IA from the server for a Renew message the client can assume it needs to send a Request to reestablish an IA with the server.

When the client receives a NoBinding error status in an IA from the server for a Rebind message the client can assume it needs to send a Request to reestablish an IA with the server or try another server.

When the client receives a Rebd_NoMatch error status in an IA from the server for a Rebind message the client can assume it needs to send a Request to reestablish an IA with the server or try another server.

When the client receives an Unavail error status in an IA from the server for a Rebind message the client can assume it needs to send a Request to reestablish an IA with the server or try another server.

14.3.6. Creation and sending of Release messages

The client sets the "msg-type" field to RELEASE, and places the link-local address of the interface associated with the configuration information it wishes to release in the "client-link-local-address" field.

The client generates a transaction ID and places this value in the "transaction-ID" field.

The client places the IP address of the server that allocated the address(es) in the "server-address" field.

The client adds a DUID option to identify itself to the server. The client includes options containing the IAs it is releasing in the "options" field. The addresses to be released MUST be included in the IAs. The appropriate "status" field in the options MUST be set to indicate the reason for the release.

If the client is configured to use authentication, the client generates the appropriate authentication option, and adds this option to the "options" field. Note that the authentication option MUST be the last option in the "options" field. See section 18.9 for more details about the authentication option.

The client sends the Release message to the All DHCP Agents multicast address.

14.3.7. Time out and retransmission of Release Messages

A client MAY choose to wait for a Reply message from the server in response to the Release message. If the client does wait for a Reply, the client MAY choose to retransmit the Release message.

If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Release, doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or REL_MSG_ATTEMPTS unsuccessful attempts have been made, at which time the client SHOULD abort the release attempt. The client SHOULD return the abort status to the application, if an application initiated the release.

Default and initial values for REP_MSG_TIMEOUT and REL_MSG_ATTEMPTS are documented in section 7.5.

Note that if the client fails to release the IA, the addresses assigned to the IA will be reclaimed by the server when the lease associated with it expires.

14.3.8. Receipt of Reply message in response to a Release message

Upon receipt of a valid Reply message, the client can consider the Release event successful, and SHOULD return the successful status to the application layer, if an application initiated the release.

14.3.9. Creation and sending of Decline messages

The client sets the "msg-type" field to DECLINE, and places the link-local address of the interface associated with the configuration information it wishes to decline in the "client-link-local-address" field.

The client generates a transaction ID and places this value in the "transaction-ID" field.

The client places the IP address of the server that allocated the address(es) in the "server-address" field.

The client adds a DUID option to identify itself to the server. The client includes options containing the IAs it is declining in the "options" field. The addresses to be released MUST be included in the IAs. The appropriate "status" field in the options MUST be set to indicate the reason for declining the address.

If the client is configured to use authentication, the client generates the appropriate authentication option, and adds this option to the "options" field. Note that the authentication option MUST be the last option in the "options" field. See section 18.9 for more details about the authentication option.

The client send the Decline message to the All DHCP Agents multicast address.

14.3.10. Time out and retransmission of Decline Messages

If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Decline, doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or REL_MSG_ATTEMPTS unsuccessful attempts have been made, at which time the client SHOULD abort the attempt to decline the address. The client SHOULD return the abort status to the application, if an application initiated the release.

Default and initial values for REP_MSG_TIMEOUT and REL_MSG_ATTEMPTS are documented in section 7.5.

14.3.11. Receipt of Reply message in response to a Release message

Upon receipt of a valid Reply message, the client can consider the Release event successful, and SHOULD return the successful status to the application layer, if an application initiated the release.

14.4. Server Behavior

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

14.4.1. Receipt of Request messages

Upon the receipt of a valid Request message from a client the server can respond to, (implementation-specific administrative policy satisfied) the server scans the options field.

The server then constructs a Reply message and sends it to the client.

The server SHOULD process each option for the client in an implementation-specific manner. The server MUST construct a Reply message containing the following values:

msg-type	REPLY
preference	Enter the server's preference to provide services to the client.
transaction-ID	Enter the transaction-ID from the Request message.
client-link-local address	Enter the client-link-local address from the Request message.
server address	Enter the IP address of the server.

When the server receives a Request and IA option is included the client is requesting the configuration of a new IA by the server. The server MUST take the clients IA and associate a binding for that client in an implementation-specific manner within the server's configuration parameter database for DHCP clients.

If the server cannot provide addresses to the client it SHOULD send back an empty IA to the client with the status field set to Unavail.

If the server can provide addresses to the client it MUST send back the IA to the client with all fields entered and a status of Success, and add the IA as a new client binding.

The server adds options to the Reply message for any other configuration information to be assigned to the client.

14.4.2. Receipt of Confirm messages

Upon the receipt of a valid Confirm message from a client the server can respond to, (implementation-specific administrative policy satisfied) the server scans the options field.

The server then constructs a Reply message and sends it to the client.

The server SHOULD process each option for the client in an implementation-specific manner. The server MUST construct a Reply message containing the following values:

msg-type	REPLY
preference	Enter the server's preference to provide services to the client.
transaction-ID	Enter the transaction-ID from the Confirm message.
client-link-local address	Enter the client-link-local address from the Confirm message.
server address	Enter the server's address.

When the server receives a Confirm and an IA option is included the client is requesting confirmation that the addresses in the IA are valid. The server SHOULD locate the clients binding and verify the information in the IA from the client matches the information stored for that client.

If the server cannot find a client entry for this IA the server SHOULD return an empty IA with status set to NoBinding.

If the server finds that the information for the client does not match what is in the server's records for that client the server should send back an empty IA with status set to Conf_NoMatch.

If the server finds a match to the Confirm then the server should send back the IA to the client with status set to success.

14.4.3. Receipt of Renew messages

Upon the receipt of a valid Renew message from a client the server can respond to, (implementation-specific administrative policy satisfied) the server scans the options field.

The server then constructs a Reply message and sends it to the client.

The server SHOULD process each option for the client in an implementation-specific manner. The server MUST construct a Reply message containing the following values:

msg-type	REPLY

preference	Enter the server's preference to provide services to the client.

transaction-ID	Enter the transaction-ID from the Confirm message.
client-link-local address	Enter the client-link-local address from the Confirm message.
server address	Enter the server's address.

When the server receives a Renew and IA option from a client it SHOULD locate the clients binding and verify the information in the IA from the client matches the information stored for that client.

If the server cannot find a client entry for this IA the server SHOULD return an empty IA with status set to NoBinding.

If the server finds that the addresses in the IA for the client do not match the clients binding the server should return an empty IA with status set to Renw_NoMatch.

If the server cannot Renew addresses for the client it SHOULD send back an empty IA to the client with the status field set to Unavail.

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 lease times and T1/T2 times if the default is not being used, and set status to Success.

14.4.4. Receipt of Rebind messages

Upon the receipt of a valid Rebind message from a client the server can respond to, (implementation-specific administrative policy satisfied) the server scans the options field.

The server then constructs a Reply message and sends it to the client.

The server SHOULD process each option for the client in an implementation-specific manner. The server MUST construct a Reply message containing the following values:

msg-type	REPLY
preference	Enter the server's preference to provide services to the client.
transaction-ID	Enter the transaction-ID from the Confirm message.

client-link-local address	Enter the client-link-local address from the Confirm message.
server address	Enter the server's address.

When the server receives a Rebind and IA option from a client it SHOULD locate the clients binding and verify the information in the IA from the client matches the information stored for that client.

If the server cannot find a client entry for this IA the server SHOULD return an empty IA with status set to NoBinding.

If the server finds that the addresses in the IA for the client do not match the clients binding the server should return an empty IA with status set to Rebd_NoMatch.

If the server cannot Rebind addresses for the client it SHOULD send back an empty IA to the client with the status field set to Unavail.

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 lease times and T1/T2 times if the default is not being used, and set status to Success.

DISCUSSION:

There is a significant difference between Renew and Rebind messages: Because the Rebind message is processed by a single server, the respnding server can actually change the addresses in the IA. However, because multiple servers may repsond to a Rebind, all they can safely do is update T1, T2 (for the IA) and lifetimes (for individual addresses).

14.4.5. Receipt of Release messages

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 IA, the server deletes the addresses from the IAs and makes the addresses available for assignment to other clients.

The server then generates a Reply message. If all of the IAs were valid and the addresses successfully released,, the server sets the "status" field to "Success". If any of the IAs were invalid or if any of the addresses were not successfully released, the server releases none of the addresses in the message and sets the "status" field to "NoBinding"(section 7.4).

If the client successfully releases some but not all of the addresses in an IA, the IA continues to exist and holds the remaining, unreleased addresses.

A client can send an option containing an IA with no listed addresses to release implicitly all of the addresses in the IA.

A server is not required to (but may choose to as an implementation strategy) retain any record of an IA from which all of the addresses have been released.

14.4.6. Sending of Reply messages

If the Request, Confirm, Renew, Rebind or Release message from the client was originally received by the server, the server unicasts the Reply message to the link-local address in the "client-link-local-address" field.

If the message was originally received in a Forward-request or Forward-release message from a relay, the server places the Reply message in the options field of a Response-reply message and unicasts the message to the relay's address from the original message.

15. DHCP Server-Initiated Configuration Exchange

A server initiates a configuration exchange to force 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 (system or application).

15.1. Reconfigure-init Message Validation

Agents MUST silently discard any received Reconfigure-init messages.

Clients MUST discard any Reconfigure-init messages that do not contain an authentication option or that fail the client's authentication check.

15.2. Server Behavior

A server sends a Reconfigure-init message to cause a client to initiate immediately a Request/Reply message exchange with the server.

15.2.1. Creation and sending of Reconfigure-init messages

The server sets the "msg-type" field to RECONFIG-INIT. The server generates a transaction-ID and inserts it in the "transaction-ID" field. The server places its address (of appropriate scope) in the "server-address" field.

The server MAY include an ORO 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 ORO if the server wants the client to obtain new address information.

The server MUST include an authentication option with the appropriate settings and add that option as the last option in the "options" field of the Reconfigure-init message.

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

The server unicasts the Reconfigure-init message to one client. The server may unicast Reconfigure-init messages to more than one client concurrently; for example, to reliably reconfigure all known clients, the server will unicast a Reconfigure-init message to each client.

After the server sends the Reconfigure-init message, it waits for a Request message from those clients confirming that each client has received the Reconfigure-init and are thus initiating a Request/Reply transaction with the server.

15.2.2. Time out and retransmission of Reconfigure-init messages

If the server does not receive a Request message from the client in RECREP_MSG_TIMEOUT milliseconds, the server retransmits the Reconfigure-init message, doubles the RECREP_MSG_TIMEOUT value and waits again. The server continues this process until REC_MSG_ATTEMPTS unsuccessful attempts have been made, at which point the server SHOULD abort the reconfigure process.

Default and initial values for RECREP_MSG_TIMEOUT and REC_MSG_ATTEMPTS are documented in section 7.5.

15.2.3. Receipt of Request messages

The server generates and sends Reply message(s) to the client as described in section 14.4.6, including in the "options" field new values for configuration parameters.

It is possible that the client may send a Request message after the server has sent a Reconfigure-init but before the Reconfigure-init is received by the client. In this case, the client's Request message may not include all of the IAs and requests for parameters to be reconfigured by the server. To accommodate this scenario, the server MAY choose to send a Reply with the IAs and other parameters to be reconfigured, even if those IAs and parameters were not in the Request message from the client.

15.3. Client Behavior

A client MUST always monitor UDP port 546 for Reconfigure-init messages on interfaces upon which it has acquired DHCP parameters. 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.

15.3.1. Receipt of Reconfigure-init messages

Upon receipt of a valid Reconfigure-init message, the client initiates a Request/Reply transaction with the server. While the Request/Reply transaction is in progress, the client silently discards any Reconfigure-init messages it receives.

DISCUSSION:

The Reconfigure-init message acts as a trigger that signals the client to complete a successful Request/Reply message exchange. Once the client has received a Recongfigure-init, the client proceeds with the Request/Reply message exchange (retransmitting the Request if necessary); the client ignores any additional Reconfigure-init messages (regardless of the transaction ID in the Reconfigure-init message) until the Request/Reply exchange is complete. Subsequent Reconfigure-init messages (again independent of the transaction ID) cause the client to initiate a new Request/Reply exchange.

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

It might be possible for a duplicate or retransmitted Reconfigure-init to be sufficiently delayed (and delivered out of order) to arrive at the client after the Request/Reply exchange (initiated by the original Reconfigure-init) has been completed. In this case, the client would initiate a redundant Request/Reply 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.

15.3.2. Creation and sending of Request messages

When responding to a Reconfigure-init, the client creates and sends the Request message in exactly the same manner as outlined in section 14.3.1 with the following difference:

IAs The client includes IA options containing the addresses the client currently has assigned to those IAs for the interface through which the Reconfigure-init message was received.

15.3.3. Time out and retransmission of Request messages

The client uses the same variables and retransmission algorithm as it does with Request messages generated as part of a client-initiated configuration exchange. See section 14.3.1 for details.

15.3.4. Receipt of Reply messages

Upon the receipt of a valid Reply message, the client extracts the contents of the "options" field, and sets (or resets) configuration parameters appropriately. The client records and updates the lifetimes for any addresses specified in IAs in the Reply message. If the configuration parameters changed were requested by the application layer, the client notifies the application layer of the changes using an implementation-specific interface.

As discussed in section 15.2.3, the Reply from the server may include IAs and parameters that were not included in the Request message from the client. The client MUST configure itself with all of the IAs and parameters in the Reply from the server.

16. Relay Behavior

For this discussion, the Relay may be configured to use a list of server destination addresses, which may include unicast addresses, the All DHCP Servers multicast address, or other multicast addresses selected by the network administrator. If the Relay has not been explicitly configured, it will use the All DHCP Servers multicast address as the default.

16.1. Relaying of client messages

When a Relay receives a valid client message, it constructs a Relay-forward message. The relay places an address from the interface on which the client message was received in the "relay-address" field and the prefix length for that address in the "prefix-length" field. This address will be used by the server to identify the link to which the client is connected and will be used by the relay to forward the Advertise message from the server back to the client.

The relay constructs a "client-message" option 18.5 that contains the entire message from the client in the data field of the option. The relay places the "relay-message" option along with any "relay-specific" options in the options field of the Relay-forward message. The Relay then sends the Relay-forward message to the list of server destination addresses that it has been configured with.

16.2. Relaying of server messages

When the relay receives a Relay-reply message, it extracts the server message from the "server-message" option and forwards the message to the address in the client-link-local-address field in the server message. The relay forwards the server message through the interface identified in the "relay-address" field in the Relay-reply message.

17. 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.

Because of the risk of denial of service attacks against DHCP clients, the use of authentication is mandated in Reconfigure-init messages. A DHCP server MUST include an authentication option in Reconfigure-init messages sent to clients.

The DHCP authentication mechanism is based on the design of authentication for DHCP for IPv4 [8].

17.1. DHCP threat model

The threat to DHCP is inherently an insider threat (assuming a properly configured network where DHCPv6 ports are blocked on the enterprise's perimeter gateways.) Regardless of the gateway configuration, however, the potential attacks by insiders and outsiders are the same.

The attack specific to a DHCP client is the possibility of the establishment of a "rogue" server with the intent of providing incorrect configuration information to the client. The motivation for doing so may be to establish a "man in the middle" attack or it may be for a "denial of service" attack.

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

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 valid addresses, or the exhaustion of CPU or network bandwidth, and are present anytime there is a shared resource. In current practice, redundancy mitigates DoS attacks the best.

17.2. Summary of DHCP authentication

Authentication of DHCP messages is accomplished through the use of the Authentication option. 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.

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.

17.3. 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 monotonically increasing counter. Using a counter value such as the current time of day (e.g., an NTP-format timestamp [12]) can reduce the danger of replay attacks. This method MUST be supported by all protocols.

17.4. Configuration token protocol

If the protocol field is 0, the authentication information field holds a simple configuration token. The configuration token is an opaque, unencoded value known to both the sender and receiver. The sender inserts the configuration token in the DHCP message and the receiver matches the token from the message to the shared token. If the configuration option is present and the token from the message does not match the shared token, the receiver MUST discard the message.

Configuration token may be used to pass a plain-text configuration token and provides only weak entity authentication and no message authentication. This protocol is only useful for rudimentary protection against inadvertently instantiated DHCP servers.

DISCUSSION:

The intent here is to pass a constant, non-computed token such as a plain-text password. Other types of entity authentication using computed tokens such as Kerberos tickets or one-time passwords will be defined as separate protocols.

17.5. Delayed authentication protocol

If the protocol field is 1, 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.

The use of a particular technique based on the HMAC protocol [10] using the MD5 hash [19] is defined here.

17.5.1. Management issues in the delayed authentication protocol

The "delayed authentication" protocol does not attempt to address situations where a client may roam from one administrative domain to another, i.e. interdomain roaming. This protocol is focused on solving the intradomain problem where the out-of-band exchange of a shared secret is feasible.

17.5.2. Use of the Authentication option in the delayed authentication protocol

In a Solicit message, the Authentication option carries the Protocol, Algorithm, RDM and Replay detection fields, but no Authentication information.

In an Advertise, Request, Renew, Rebind or Confirm message, the Authentication option carries the Protocol, Algorithm, RDM and Replay detection fields and Authentication information. The format of the Authentication information 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Secret ID (32 bits)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                      HMAC-MD5 (128 bits)                      |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The following definitions will be used in the description of the authentication information for delayed authentication, algorithm 1:

```
Replay Detection  - as defined by the RDM field
K                 - a secret value shared between the source and
                    destination of the message; each secret has a
                    unique identifier (secret ID)
secret ID         - the unique identifier for the secret value
                    used to generate the MAC for this message
HMAC-MD5          - the MAC generating function.
```

The sender computes the MAC using the HMAC generation algorithm [10] and the MD5 hash function [19]. The entire DHCP message (except as noted below), including the DHCP message header and the options field, is used as input to the HMAC-MD5 computation function. The 'secret ID' field MUST be set to the identifier of the secret used to generate the MAC.

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.

Delayed authentication requires a shared secret key for each client on each DHCP server with which that client may wish to use the DHCP protocol. Each secret key has a unique identifier that can be used by a receiver to determine which secret was used to generate the MAC in the DHCP message. Therefore, delayed authentication may not scale well in an architecture in which a DHCP client connects to multiple administrative domains.

17.5.3. Message validation

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. Next, the receiver computes the MAC as described in [10]. The receiver MUST set the 'MAC' field of the authentication option to all 0s for computation of the MAC, and because a DHCP relay agent may alter the values of the 'giaddr' and 'hops' fields in the DHCP message, the contents of those two fields MUST also be set to zero for the computation of the MAC. If the MAC computed by the receiver does not match the MAC contained in the authentication option, the receiver MUST discard the DHCP message.

17.5.4. Key utilization

Each DHCP client has a key, K. The client uses its key to encode any messages it sends to the server and to authenticate and verify any messages it receives from the server. The client's key SHOULD be initially distributed to the client through some out-of-band mechanism, and SHOULD be stored locally on the client for use in all authenticated DHCP messages. Once the client has been given its key, it SHOULD use that key for all transactions even if the client's configuration changes; e.g., if the client is assigned a new network address.

Each DHCP server MUST know, or be able to obtain in a secure manner, the keys for all authorized clients. If all clients use the same key, clients can perform both entity and message authentication for all messages received from servers. However, the sharing of keys is strongly discouraged as it allows for unauthorized clients to masquerade as authorized clients by obtaining a copy of the shared key. To authenticate the identity of individual clients, each client MUST be configured with a unique key.

17.5.5. Client considerations for delayed authentication protocol

17.5.5.1. 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, RDM and replay detection field as described in section 17.5. The client does not include any authentication information in the Authentication option.

17.5.6. 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 17.5.3.

Client behavior if no Advertise messages include authentication information or pass the validation test is controlled by local policy on the client. According to client policy, the client MAY choose to respond to a Advertise message that has not been authenticated.

The decision to set local policy to accept unauthenticated messages should be made with care. Accepting an unauthenticated Advertise message can make the client vulnerable to spoofing and other attacks. If local users are not explicitly informed that the client has accepted an unauthenticated Advertise message, the users may incorrectly assume that the client has received an authenticated address and is not subject to DHCP attacks through unauthenticated messages.

A client MUST be configurable to discard unauthenticated messages, and SHOULD be configured by default to discard unauthenticated messages. 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.

17.5.6.1. Sending Request, Confirm, Renew, Rebind 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 17.5. When the client sends a subsequent message, it MUST use the same secret used by the server to generate the authentication information.

17.5.6.2. 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 discard the Reply if the message fails to pass validation and MAY log the validation failure. If the Reply fails to pass validation, the client MUST restart the DHCP configuration process by sending a Solicit message. The client MAY choose to remember which server replied with a Reply message that failed to pass validation and discard subsequent messages from that server.

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.

17.5.7. Server considerations for delayed authentication protocol

17.5.7.1. Receiving Solicit messages and Sending Advertise messages

The server selects a secret for the client and includes authentication information in the Advertise message returned to the client as specified in section 17.5. The server MUST record the identifier of the secret selected for the client and use that same secret for validating subsequent messages with the client.

17.5.7.2. Receiving Request, Confirm, Renew, Rebind or Release messages and Sending Reply messages

The server uses the secret identified in the message and validates the message as specified in section 17.5.3. If the message fails to pass validation or the server does not know the secret identified by the 'secret ID' field, the server MUST discard the message and MAY choose to log the validation failure.

If the message passes the validation procedure, the server responds to the specific message as described in section 14.4. The server MUST include authentication information generated using the secret identified in the received message as specified in section 17.5.

17.5.7.3. Sending Reconfigure-Init messages

The server MUST include authentication information in a Reconfigure-Init message, generated as specified in section 17.5 using the secret the server initially selected for the client to which the Reconfigure-Init message is to be sent.

18. 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 18.1.

This document describes the DHCP options defined as part of the base DHCP specification. Other options may be defined in the future in a separate document.

18.1. Format of DHCP options

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |          option-code          |           option-len          |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                          option-data                          |
     |                      (option-len octets)                      |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code An unsigned integer identifying the specific option type carried in this option.

option-len An unsigned integer giving the length of the data in this option in octets.

option-data The data for the option; the format of this data depends on the definition of the option.

18.2. DHCP unique identifier option

The DHCP unique identifier option is used to carry a DUID. The format for the DUID is keyed to mark the type of identifier and is of variable length. The format of the DUID 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 DUID          |          option-len           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |           DUID type           |           DUID len            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                             DUID                              |
     .                                                               .
     .                                                               .
     .                                                               .
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

18.3. Identity association option

The identity association option is used to carry an identity association, the parameters associated with the IA and the addresses assigned to the IA.

The format of the IA 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 IA           |          option-len           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        IAID (4 octets)                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                              T1                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                              T2                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   IA status   |   num-addrs   |T| addr status | prefix length |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                          IPv6 address                         |
    |                           (16 octets)                         |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                       preferred lifetime                      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                         valid lifetime                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |T| addr status | prefix length |                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
    |                          IPv6 address                         |
    |                           (16 octets)                         |
    |                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                               |      preferred lifetime       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | pref. lifetime (cont.)        |        valid lifetime         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | valid lifetime (cont.)        |T| addr status | prefix length |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                          IPv6 address                         |
    |                           (16 octets)                         |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                               ...                             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code   OPTION_IA (1)

option-len   Variable; equal to 24 + num-addrs*26

IA ID — The unique identifier for this IA; chosen by the client

T1 — The time at which the client contacts the server from which the addresses in the IA were obtained to extend the lifetimes of the addresses assigned to the IA.

T2 — The time at which the client contacts any available server to extend the lifetimes of the addresses assigned to the IA.

T — When set to 1, indicates that this address is a "temporary address" [15]; when set to 0, the address is not a temporary address.

IA status — Status of the IA in this option.

num-addrs — An unsigned integer giving the number of addresses carried in this IA option (MAY be zero).

addr status — Status of the addresses in this IA.

prefix length — Prefix length for this address.

IPv6 address — An IPv6 address assigned to this IA.

preferred lifetime — The preferred lifetime for the associated IPv6 address.

valid lifetime — The valid lifetime for the associated IPv6 address.

The "IPv6 address", "preferred lifetime" and "valid lifetime" fields are repeated for each address in the IA option (as determined by the "num-addrs" field).

Note that an IA has no explicit "lifetime" or "lease length" of its own. When the lifetimes of all of the addresses in an IA have expired, the IA 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.

The 'T' bit identifies the associated address as a temporary address. If the server is configured to assign temporary addresses to the client, the server marks those temporary addresses with the 'T' bit. The number of temporary addresses assigned to the client and the lifetimes of those addresses is determined by the administrative configuration of the server. The 'T' bit only identifies an address as a temporary address; identification of an address as ``temporary'' has no implication on the lifetime of the extensibility of the lifetime of the address.

18.4. Option request option

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           OPTION_ORO          |           option-len          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    requested-option-code-1    |    requested-option-code-2    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                              ...                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code   OPTION_ORO (2)

option-len   Variable; equal to twice the number of option codes carried in this option.

option-data   A list of the option codes for the options requested in this option.

18.5. Client message option

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        OPTION_CLIENT_MSG      |           option-len          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                       DHCP client message                     |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code   OPTION_CLIENT_MSG (3)

option-len   Variable; equal to the length of the forwarded DHCP client message.

option-data   The message received from the client; forwarded verbatim to the server.

18.6. Server message option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       OPTION_SERVER_MSG       |          option-len           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      DHCP server message                      |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code   OPTION_SERVER_MSG (4)

option-len   Variable; equal to the length of the forwarded DHCP server message.

option-data   The message received from the server; forwarded verbatim to the client.

18.7. Retransmission parameter option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      OPTION_RETRANS_PARM      |          option-len           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          option-data                          |
|                      (option-len octets)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code   OPTION_RETRANS_PARM (5)

option-len   An unsigned integer giving the length of the data in this option in octets.

option-data   TBD - The details of the operational parameters to be set in the client

18.8. DSTM Global IPv4 Address Option

The DSTM Global IPv4 Address Option informs a client or server that the Identity Association Option (IA) following this option will contain an IPv4-Mapped IPv6 Address [9] in the case of a Client receiving the option, or is a Request for an IPv4-Mapped IPv6 Address from a client in the case of a DHCPv6 Server receiving the option. The option can also provide a set of IPv6 addresses to be used as the Tunnel Endpoint (TEP) to encapsulate an IPv6 packet within IPv6.

This option can be used with the Request, Reply, and Reconfigure-Init Messages for cases where a server wants to assign to clients IPv4-Mapped IPv6 Addresses, thru the Option Request Option (ORO).

```
 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_DSTM          |            option-length      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Tunnel End Point (TEP)               |
|                           (If Present)                        |
|                            (16 octets)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option code   OPTION_DSTM (7)

option length   Variable: 0 or multiple of 16

tunnel end point   IPv6 Address or addresses if Present

A DSTM IPv4 Global Address Option MUST only apply to the IA following this option.

18.9. 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 17.

The format of the Authentication 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_AUTH          |        option-length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Protocol    |   Algorithm   |      RDM      | Replay detect.|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Replay Detection (64 bits)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Replay cont.                  | Auth. Info    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|           Authentication Information                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
      option-code                  OPTION_AUTH (TBD)
```

```
   option-length              Variable

   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
```

18.10. Server unicast option

This option is used by a server to send to a client to inform the client it can send a Request, Renew, Confirm, Release, and Decline by unicasting directly to the server instead of the ALL-DHCPv6-Agents Multicast address as an optimization, when the client as an address of sufficient scope to reach the server.

```
 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_UNICAST       |           option-length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   option-code     OPTION_UNICAST (TBD)

   option-length   0
```

This option only applies to the server address that sends this to the client.

18.11. Domain Search Option

This option provides a list of domain names a client can use to resolve DNS names.

```
 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_DOMAIN_SEARCH_LIST   |         option-length         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   |                       Domain Search List                      |
   |                              ...                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

msg type   OPTION_DOMAIN_SEARCH_LIST (TBD)

option-length   variable

Domain Search List   The DNS domain search list the client should use to resolve names.

So that the search list may be encoded compactly and uniformly, search strings in the search list are concatenated and encoded using the technique described in section 4.1 of [13].

For use in this specification, the compression pointer (see section 4.1.4 of [13]) refers to the offset within the SearchString portion of the option.

18.12. Domain Name Server Option

This option provides a list of Domain Name System [13] that a client name resolver can use to access DNS services. There must be at least 1 server listed in this option.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      OPTION_DNS_SERVERS       |          option_length        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                   DNS server (IP address)                     |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                   DNS server (IP address)                     |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                              ...                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

msg-type   OPTION_DNS_SERVERS (TBD) option-length   variable

DNS server   IPv6 address of a DNS name server for the client to use. The DNS servers are listed in the order of preference for use by the client resolver.

19. DHCP Client Implementor Notes

This section provides helpful information for the client implementor regarding their implementations. The text described here is not part of the protocol, but rather a discussion of implementation features we feel the implementor should consider during implementation.

19.1. Primary Interface

Since configuration parameters acquired through DHCP can be interface-specific or more general, the client implementor SHOULD provide a mechanism by which the client implementation can be configured to specify which interface is the primary interface. The client SHOULD always query the DHCP data associated with the primary interface for non-interface specific configuration parameters. An implementation MAY implement a list of interfaces which would be scanned in order to satisfy the general request. In either case, the first interface scanned is considered the primary interface.

By allowing the specification of a primary interface, the client implementor identifies which interface is authoritative for non-interface specific parameters, which prevents configuration information ambiguity within the client implementation.

19.2. Advertise Message and Configuration Parameter Caching

If the hardware the client is running on permits it, the implementor SHOULD provide a cache for Advertise messages and a cache of configuration parameters received through DHCP. Providing these caches prevents unnecessary DHCP traffic and the subsequent load this generates on the servers. The implementor SHOULD provide a configuration knob for setting the amount of time the cache(s) are valid.

19.3. Time out and retransmission variables

Note that the client time out and retransmission variables outlined in section 7.5 can be configured on the server and sent to the client through the use of the "DHCP Retransmission Parameter Option", which is documented in section 18.7. A client implementation SHOULD be able to reset these variables using the values from this option.

19.4. Server Preference

A client MUST wait for SRVR_PREF_WAIT seconds after sending a DHCP Solicit message to collect Advertise messages and compare their preferences (see section 20.3), unless it receives an Advertise message with a preference of 255. If the client receives an Advertise message with a preference of 255, then the client MAY act immediately on that Advertise without waiting for any more additional Advertise messages.

20. DHCP Server Implementor Notes

This section provides helpful information for the server implementor.

20.1. Client Bindings

A server implementation MUST use the IA's DUID and the prefix specification from which the client sent its Request message(s) as an index for finding configuration parameters assigned to the client. While it isn't critical to keep track of the other parameters assigned to a client, the server MUST keep track of the addresses it has assigned to an IA.

The server should periodically scan its bindings for addresses whose leases have expired. When the server finds expired addresses, it MUST delete the assignment of those addresses, thereby making these addresses available to other clients.

The client bindings MUST be stored in non-volatile storage.

The server implementation should provide policy knobs to control whether or not the lifetimes on assigned addresses are renewable, and by how long.

20.2. Reconfigure-init Considerations

A server implementation MUST provide an interface to the administrator for initiating reconfigure-init events.

20.3. Server Preference

The server implementation SHOULD allow the setting of a server preference value by the administrator. The server preference variable is an unsigned single octet value (0--255), with the lowest preference being 0 and the highest 255. Clients will choose higher preference servers over those with lower preference values. If you don't choose to implement this feature in your server, you MUST set the server preference field to 0 in the Advertise messages generated by your server.

20.4. Request Message Transaction-ID Cache

In order to improve performance, a server implementation MAY include an in memory transaction-ID cache. This cache is indexed by client binding and transaction-ID, and enables the server to quickly determine whether a Request is a retransmission or a new Request without the cost of a database lookup. If an implementor chooses to implement this cache, then they SHOULD provide a configuration knob to tune the lifetime of the cache entries.

21. DHCP Relay Implementor Notes

A relay implementation SHOULD allow the specification of a list of destination addresses for forwarded messages. This list MAY contain any mixture of unicast addresses and multicast addresses.

If a relay receives an ICMP message in response to a DHCP message it has forwarded, it SHOULD log this event.

22. Security

Section 17 describes a threat model and an option that provides an authentication framework to defend against that threat model.

23. Year 2000 considerations

Since all times are relative to the current time of the transaction, there is no problem within the DHCPv6 protocol related to any hardcoded dates or two-digit representation of the current year.

24. IANA Considerations

This document defines several new name spaces associated with DHCPv6 and DHCPv6 options. IANA is requested to manage the allocation of values from these name spaces.

New values in each of these name spaces should be approved by the process of IETF Consensus [14].

24.1. DHCPv6 options

This document defines message types TBD to be received by UDP at port numbers 546 and 547. Additional message types may be defined in the future.

24.2. Multicast addresses

Section 7.1 lists several multicast addresses used by DHCP. Additional multicast addresses may be defined in the future.

24.3. Status codes

Section 9.7 defines several status codes that are to be returned with the Reply message. The non-zero values for these status codes that are currently specified are shown in the table in section 7.4.

24.4. Retransmission parameter option

There is a DHCPv6 option described in section 18.7, which allows clients and servers to exchange values for some of the timing and retransmission parameters defined in section 7.5. Adding new parameters in the future would require extending the values by which the parameters are indicated in the DHCP option. Since there needs to be a list kept, the default values for each parameter should also be stored as part of the list.

24.5. Authentication option

Section 17 defines three new name spaces associated with the Authentication Option (section 18.9), which are to be created and maintained by IANA: Protocol, Algorithm and RDM.

Initial values assigned from the Protocol name space are 0 (for the configuration token Protocol in section 17.4) and 1 (for the delayed authentication Protocol in section 17.5). Additional protocols may be defined in the future.

The Algorithm name space is specific to individual Protocols. That is, each Protocol has its own Algorithm name space. The guidelines for assigning Algorithm name space values for a particular protocol should be specified along with the definition of a new Protocol.

For the configuration token Protocol, the Algorithm field MUST be 0, as described in section 17.4. For the delayed authentication Protocol, the Algorithm value 1 is assigned to the HMAC-MD5 generating function as defined in section 17.5. Additional algorithms for the delayed authentication protocol may be defined in the future.

The initial value of 0 from the RDM name space is assigned to the use of a monotonically increasing value as defined in section 17.3.

Additional replay detection methods may be defined in the future.

25. Acknowledgments

Thanks to the DHC Working Group for their time and input into the specification. Ralph Droms and Thomas Narten have had a major role in shaping the continued improvement of the protocol by their careful reviews. Many thanks to Matt Crawford, Erik Nordmark, Gerald Maguire, and Mike Carney for their studied review as part of the Last Call process. Thanks also for the consistent input, ideas, and review by (in alphabetical order) Brian Carpenter, Francis DuPont, Ted Lemon, Jack McCann, Yakov Rekhter, Matt Thomas, Sue Thomson, Bernie Volz and Phil Wells.

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

Bill Arbaugh reviewed the authentication mechanism described in section 17.

The Domain Search option described in section 18.11 is based on the DHCPv4 domain search option, [1], and was reviewed by Bernard Aboba.

A. Comparison between DHCPv4 and DHCPv6

This appendix is provided for readers who will find it useful to see a model and architecture comparison between DHCPv4 [7, 2] and DHCPv6. There are three key reasons for the differences:

- o IPv6 inherently supports a new model and architecture for communications and autoconfiguration of addresses.
- o DHCPv6 benefits from the new IPv6 features.
- o New features were added to support the expected evolution and the existence of more complicated Internet network service requirements.

IPv6 Architecture/Model Changes:

- o The link-local address permits a node to have an address immediately when the node boots, which means all clients have a source IP address at all times to locate an on-link server or relay.
- o The need for BOOTP compatibility and the broadcast flag have been removed.
- o Multicast and address scoping in IPv6 permit the design of

discovery packets that would inherently define their range by the multicast address for the function required.

- o Stateful autoconfiguration has to coexist and integrate with stateless autoconfiguration supporting Duplicate Address Detection and the two IPv6 lifetimes, to facilitate the dynamic renumbering of addresses and the management of those addresses.

- o Multiple addresses per interface are inherently supported in IPv6.

- o Some DHCPv4 options are unnecessary now because the configuration parameters are either obtained through IPv6 Neighbor Discovery or the Service Location protocol [21].

DHCPv6 Architecture/Model Changes:

- o The message type is the first octet in the packet.

- o IPv6 Address allocations are now handled in a message option as opposed to the message header.

- o Client/Server bindings are now mandatory and take advantage of the client's link-local address to always permit communications either directly from an on-link server, or from a off-link server through an on-link relay.

- o Servers are discovered by a client Solicit, followed by a server Advertise message

- o The client will know if the server is on-link or off-link.

- o The on-link relay may locate off-link server addresses from system configuration or by the use of a site-wide multicast packet.

- o ACKs and NAKs are not used.

- o The server assumes the client receives its responses unless it receives a retransmission of the same client request. This permits recovery in the case where the network has faulted.

- o Clients can issue multiple, unrelated Request messages to the same or different servers.

- o The function of DHCPINFORM is inherent in the new packet design; a client can request configuration parameters other than IPv6 addresses in the optional option headers.

- o Clients MUST listen to their UDP port for the new Reconfigure-init message from servers.

o New options have been defined.

With the changes just enumerated, we can support new user features, including

- o Configuration of Dynamic Updates to DNS
- o Address deprecation, for dynamic renumbering.
- o Relays can be preconfigured with server addresses, or use of multicast.
- o Authentication
- o Clients can ask for multiple IP addresses.
- o Addresses can be reclaimed using the Reconfigure-init message.
- o Integration between stateless and stateful address autoconfiguration.
- o Enabling relays to locate off-link servers.

B. Full Copyright Statement

C. Changes in this draft

This section describes the changes between this version of the DHCPv6 specification and draft-ietf-dhc-dhcpv6-19.txt.

C.1. Reconfigure-init

The client behavior in response to a Reconfigure-init message described in section 15 has been changed. When the client receives a Reconfigure-init message, the client goes into "Reconfigure" mode. The client initiates a Request/Reply exchange in which the XID in client Request is independent of server Reconfigure-init XID. The server waits for the next Request message from the client to determine if the client has received the Reconfigure-init.

To avoid redundant Request/Reply messages exchanges, the client ignores subsequent Reconfigure-init messages until it completes the Request/Reply exchange.

Use of multicast for Reconfigure-init message delivery has been removed:

- Multicast only saves, at most, 1/3 of the messages when reconfiguring multiple clients

- Multicast might cause an implosion of Request messages; additional complexity in the client and protocol messages would be required to add delay to spread out Request messages

- Authentication of multicast Reconfigure-init messages (where a single message must be authenticated by multiple clients) is an open problem

Text has been added clarifying that the ORO option applies to IAs as well as other options. The server may choose to omit the IA option from the ORO in the Reconfigure-init message.

The Reconfigure-delay option (used only by multicast Reconfigure-init) has been removed.

The transaction ID feild in the Reconfigure-init message header is now marked as "(unused) MUST be zero".

C.2. Authentication

DHCPv4-style authentication has been added to this draft in section 17.

C.3. Confirm message

The following DISCUSSION was removed from the description of the Confirm message:

DISCUSSION:

This section used to allow servers to change the addresses in an IA. Without some additional mechanism, servers responding to Confirm messages can't change safely change the addresses in IAs (although they can change the lifetimes), because servers may send back different addresses.

C.4. Failure of Rebind message

In section 14.3.4, the alternatives for client behavior in the case that the client receives no response to a Rebind message were taken out of a DISCUSSION section and made part of the spec. These alternatives are really an implementation issue and not part of the DHCPv6 spec.

C.5. Server behavior in response to Release message

The following DISCUSSION was merged into the text describing server behavior in response to a Release message in section 14.4.5:

DISCUSSION:

What is the behavior of the server relative to a "partially released" IA; i.e., an IA for which some but not all addresses are released?

Can a client send an empty IA to release all addresses in the IA?

If the IA becomes empty - all addresses are released - can the server discard any record of the IA?

C.6. Client behavior when sending a Release message

Text has been added to section 14.3.6 clarifying that a client MAY (but not MUST) wait for a Reply to a Release message.

C.7. IA option

The format diagram has been corrected to include the prefix length and address status with each address. PROPOSAL - use left-most bit in address status to indicate whether an address is "temporary".

C.8. DSTM option

Definition of DSTM option has been updated to carry multiple IPv6 addresses as tunnel endpoints.

C.9. Server unicast option

An option to allow clients to use unicast where possible has been added in section 18.10.

C.10. Domain search option

An option to pass a domain name search list to a client has been added in section 18.11.

C.11. DNS servers option

An option to pass a list of DNS options to a client has been added in section 18.12.

C.12. DUID and IAID

The "DHCP unique identifier" is defined as a typed, variable length value (see section 18.2). The DUID is carried in an option. The details of the DUID are TBD.

The "IA identifier" is defined as a 4 octet identifier, unique among all IAIDs for IAs from a client.

C.13. Continuing to poll with Solicit

Text has been added to section 13.3.2 allowing a client to continue to send Solicit messages at low frequency indefinitely.

C.14. Using DHCPv6 without address assignment

Text has been added to section 14.3.1 allowing a client to send a Solicit message containing no IAs to request other configuration information without address assignment (equivalent to DHCPv4 DHCPINFORM).

C.15. Potential crossing in flight of Request and Reconfigure-init messages

Text has been added to section 15 addressing the case in which the client sends a Request after a server has sent a Reconfigure-init but before the client receives the Reconfigure-init.

D. Open Issues for Working Group Discussion

This section contains some items for discussion by the working group.

D.1. Generation and use of DUID and IAID

Details for generation and use of DUID and IA identifiers is TBD.

D.2. Address registration

Should there be a way for a DHCP client to register stateless autoconfig addresses with the server?

D.3. Prefix advertisement

Can a DHCP server advertise prefixes? This function might be used to provide managed temporary addresses - the server advertises a prefix and the client then registers selected addresses with the DHCP server.

D.4. DHCP-DNS interaction

Interaction among DHCP servers, clients and DNS servers should be discussed in this document.

What is relationship between DHCP-DNS for IPv4 (work-in-progress) and DHCP-DNS interaction requirements for IPv6?

D.5. Use of term "agent"

The term "agent", taken to mean "relay agent or server", may be confusing. "relay agent or server" might be clearer.

D.6. Additional options

Which additional options should be included in this base spec document? How should we reserve space for "local options" (as in DHCPv4)?

D.7. Operational parameters

Should servers have an option to set operational parameters - retransmission timeouts, number of retries - in clients?

References

[1] B. Aboba. DHCP Domain Search Option. Internet Draft, Internet

Engineering Task Force, December 2000. Work in progress.

[2] S. Alexander and R. Droms. DHCP Options and BOOTP Vendor Extensions. Request for Comments (Draft Standard) 2132, Internet Engineering Task Force, March 1997.

[3] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. Request for Comments (Best Current Practice) 2119, Internet Engineering Task Force, March 1997.

[4] S. Bradner and A. Mankin. The Recommendation for the IP Next Generation Protocol. Request for Comments (Proposed Standard) 1752, Internet Engineering Task Force, January 1995.

[5] W. J. Croft and J. Gilmore. Bootstrap Protocol. Request for Comments 951, Internet Engineering Task Force, September 1985.

[6] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. Request for Comments (Draft Standard) 2460, Internet Engineering Task Force, December 1998.

[7] R. Droms. Dynamic Host Configuration Protocol. Request for Comments (Draft Standard) 2131, Internet Engineering Task Force, March 1997.

[8] R. Droms and W. Arbaugh. Authentication for DHCP Messages. Internet Draft, Internet Engineering Task Force, January 2001. Work in progress.

[9] R. Hinden and S. Deering. IP Version 6 Addressing Architecture. Request for Comments (Proposed Standard) 2373, Internet Engineering Task Force, July 1998.

[10] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. Request for Comments (Informational) 2104, Internet Engineering Task Force, February 1997.

[11] J. McCann, S. Deering, and J. Mogul. Path MTU Discovery for IP version 6. Request for Comments (Proposed Standard) 1981, Internet Engineering Task Force, August 1996.

[12] David L. Mills. Network Time Protocol (Version 3) Specification, Implementation. Request for Comments (Draft Standard) 1305, Internet Engineering Task Force, March 1992.

[13] P. V. Mockapetris. Domain names - implementation and specification. Request for Comments (Standard) 1035, Internet Engineering Task Force, November 1987.

[14] T. Narten and H. Alvestrand. Guidelines for Writing an IANA

Considerations Section in RFCs. Request for Comments (Best Current Practice) 2434, Internet Engineering Task Force, October 1998.

[15] T. Narten and R. Draves. Privacy Extensions for Stateless Address Autoconfiguration in IPv6. Request for Comments (Proposed Standard) 3041, Internet Engineering Task Force, January 2001.

[16] T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). Request for Comments (Draft Standard) 2461, Internet Engineering Task Force, December 1998.

[17] D. C. Plummer. Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware. Request for Comments (Standard) 826, Internet Engineering Task Force, November 1982.

[18] J. Postel. User Datagram Protocol. Request for Comments (Standard) 768, Internet Engineering Task Force, August 1980.

[19] R. Rivest. The MD5 Message-Digest Algorithm. Request for Comments (Informational) 1321, Internet Engineering Task Force, April 1992.

[20] S. Thomson and T. Narten. IPv6 Stateless Address Autoconfiguration. Request for Comments (Draft Standard) 2462, Internet Engineering Task Force, December 1998.

[21] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. Service Location Protocol. Request for Comments (Proposed Standard) 2165, Internet Engineering Task Force, June 1997.

[22] P. Vixie, Ed., S. Thomson, Y. Rekhter, and J. Bound. Dynamic Updates in the Domain Name System (DNS UPDATE). Request for Comments (Proposed Standard) 2136, Internet Engineering Task Force, April 1997.

Chair's Address

The working group can be contacted via the current chair:

Ralph Droms
Cisco Systems
300 Apollo Drive
Chelmsford, MA 01824

Phone: (978) 244-4733
E-mail: rdroms@cisco.com

Author's Address

Questions about this memo can be directed to:

Jim Bound
Compaq Computer Corporation
ZK3-3/W20
110 Spit Brook Road
Nashua, NH 03062-2698
USA
Phone: +1 603 884 0062
Email: Jim.Bound@compaq.com

Mike Carney
Sun Microsystems, Inc
Mail Stop: UMPK17-202
901 San Antonio Road
Palo Alto, CA 94303-4900
USA
Phone: +1-650-786-4171
Email: mwc@eng.sun.com

Charles E. Perkins
Communications Systems Lab
Nokia Research Center
313 Fairchild Drive
Mountain View, California 94043
USA
Phone: +1-650 625-2986
Email: charliep@iprg.nokia.com
Fax: +1 650 625-2502

Ralph Droms
Cisco Systems
300 Apollo Drive
Chelmsford, MA 01824
USA
Phone: +1 978 244 4733
Email: rdroms@cisco.com