DHC Working Group
Internet Draft
Intended Status: Standards Track
Expires: April 18, 2011

Kim Kinnear
Bernie Volz
Mark Stapp
Cisco Systems, Inc.
D. Rao
B. Joshi
Infosys Technologies Ltd.
Neil Russell
Nokia
P. Kurapati
Juniper Networks Ltd.
October 18, 2010

Bulk DHCPv4 Lease Query
<draft-ietf-dhc-dhcpv4-bulk-leasequery-03.txt>

Status of this Memo

This Internet-Draft will expire on September 5, 2010

Copyright Notice

Abstract

The Dynamic Host Configuration Protocol for IPv4 (DHCPv4) Leasequery extension allows a requestor to request information about DHCPv4 bindings. This mechanism is limited to queries for individual bindings. In some situations individual binding queries may not be efficient, or even possible. This document extends the DHCPv4 Leasequery protocol to allow for bulk transfer of DHCPv4 address binding data via TCP.

Table of Contents

1. Introduction

The DHCPv4 protocol [RFC2131] [RFC2132] specifies a mechanism for the assignment of IPv4 address and configuration information to IPv4 nodes. DHCPv4 servers maintain authoritative binding information.

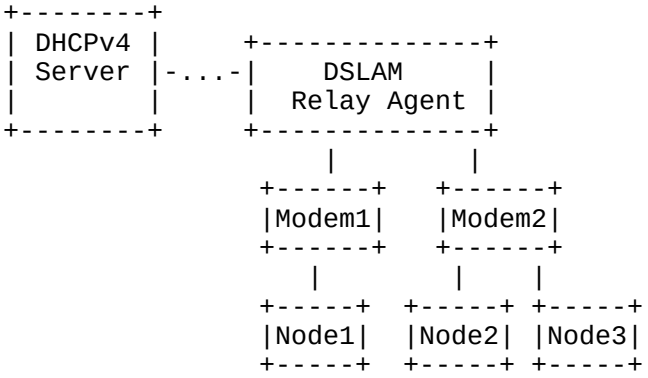


Figure 1: Example DHCPv4 configuration

DHCPv4 relay agents receive DHCPv4 messages and frequently append a relay agent information option [RFC3046] before relaying them to the configured DHCPv4 servers (see Figure 1). In this process, some relay agents also glean lease information sent by the server and cache it locally. This information is used for a variety of purposes. Two examples are prevention of spoofing attempts from the DHCPv4 clients, and installation of routes. When a relay agent reboots, this information is frequently lost.

The DHCPv4 Leasequery capability [RFC4388] extends the basic DHCPv4 capability to allow an external entity, such as a relay agent, to query a DHCPv4 server to recover lease state information about a particular IP address or client in near real-time.

The existing query types in Leasequery are typically data driven; the relay agent initiates the Leasequery when it receives data traffic from or to the client. This approach may not scale well when there are thousands of clients connected to the relay agent or when the relay agent has a need to rebuild its internal data store prior to processing traffic in one direction or another.

Some applications require the ability to query the server without waiting for traffic from or to clients. This query capability in turn requires an underlying transport more suitable to the bulk transmission of data.

This document extends the DHCPv4 Leasequery protocol to add support for queries that address these additional requirements. There may be many thousands of DHCPv4 bindings returned as the result of a single request, so TCP [RFC4614] is specified for efficiency of data transfer. We define several additional query types, each of which can return multiple responses, in order to meet a variety of requirements.

2. Terminology

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

This document uses the following terms:

- "absolute time"

  A 32-bit quantity containing the number of seconds since Jan 1, 1970.

- "access concentrator"

  An access concentrator is a router or switch at the broadband access provider’s edge of a public broadband access network. This document assumes that the access concentrator includes the DHCPv4 relay agent functionality. For example, a CMTS (Cable Modem Termination System) in Cable environment or a DSLAM (Digital Subscriber Line Access Multiplexer) in a DSL environment.

- "active binding"

  An IP address with an active binding refers to an IP address which is currently associated with a DHCPv4 client where that DHCPv4 client has the right to use the IP address.

- "Bulk Leasequery"

  Requesting and receiving the existing DHCPv4 address binding information in an efficient manner.

- "clock skew"

  The difference between the absolute time on a DHCPv4 server and the absolute time on the system where a requestor of a Bulk Leasequery is executing is termed the "clock skew" for that Bulk Leasequery connection. It is not absolutely constant but is likely to vary only slowly. It is possible that, when both systems run NTP, the clock skew is negligible, and this is not only acceptable, but desired.

  While it is easy to think that this can be calculated precisely after one message is received by a requestor from a DHCPv4 server, a more accurate value is derived from continuously examining the instantaneous value developed from each message received from a DHCPv4 server and using it to make small adjustments to the existing value held in the requestor.

- "DHCPv4 client"

  A DHCPv4 client is an Internet node using DHCPv4 to obtain configuration parameters such as a network address.

- "DHCPv4 relay agent"

  A DHCPv4 relay agent is a third-party agent that transfers BOOTP and DHCPv4 messages between clients and servers residing on different subnets, per [RFC951] and [RFC1542].

- "DHCPv4 server"

  A DHCPv4 server is an Internet node that returns configuration parameters to DHCPv4 clients.

- "DSLAM"

  Digital Subscriber Line Multiplexer.

- "downstream"

  Refers to a direction away from the central part of a network and toward the edge. In a DHCPv4 context, typically refers to a network direction which is away from the DHCPv4 server.

- "IP address"

  In this document, the term "IP address" refers to an IPv4 IP address.

- "IP address binding"

  The information that a DHCPv4 server keeps regarding the relationship between a DHCPv4 client and an IP address. This includes the identity of the DHCPv4 client and the expiration time, if any, of any lease that client has on a particular IP address. In some contexts, this may include information on IP addresses that are currently associated with DHCPv4 clients, and in others it may also include IP addresses with no current association to a DHCPv4 client.

- "MAC address"

  In the context of a DHCPv4 message, a MAC address consists of the fields: hardware type "htype", hardware length "hlen", and client hardware address "chaddr".

- "upstream"

  Refers to a direction toward the central part of a network and away from the edge. In a DHCPv4 context, typically refers to a network direction which is toward the DHCPv4 server.

- "stable storage"

  Stable storage is used to hold information concerning IP address bindings (among other things) so that this information is not lost in the event of a failure which requires restart of the network element. DHCPv4 servers are typically expected to have high speed access to stable storage, while relay agents and access concentrators usually do not have access to stable storage, although they may have periodic access to such storage.

- "xid"

  Transaction-id. The term "xid" refers to the DHCPv4 field containing the transaction-id of the message.

3. Design Goals

The goal of this document is to provide a lightweight mechanism for an Access Concentrator or other network element to retrieve IP address binding information available in the DHCPv4 server. The mechanism should also allow an Access Concentrator to retrieve consolidated IP address binding information for either the entire access concentrator or a single connection/circuit.

3.1. Information Acquisition before Data Starts

The existing data driven approach required by [RFC4388] means that the Leasequeries can only be performed after an Access Concentrator receives data. To implement antispoofing, the concentrator must drop packets for each client until it gets lease information from the DHCPv4 server for that client. If an Access Concentrator finishes the Leasequeries before it starts receiving data, then there is no need to drop legitimate packets. In this way, outage time may be reduced.

3.2. Lessen need for Caching and Negative Caching

The result of a single Leasequery should be cached, whether that results in a positive or negative cache, in order to remember that the Leasequery was performed. This caching is required to limit the traffic imposed upon a DHCPv4 server by Leasequeries for information already received.

These caches not only consume precious resources, they also need to be managed. Hence they should be avoided as much as possible.

3.3. Antispoofing in 'Fast Path'

If Antispoofing is not done in the fast path, it will become a bottleneck and may lead to denial of service of the access concentrator. The Leasequeries should make it possible to do antispoofing in the fast path.

3.4. Minimize data transmission

It may be that a network element is able to periodically save its entire list of assigned IP addresses to some form of stable storage. In this case, it will wish to recover all of the updates to this information without duplicating the information it has recovered from

its own stable storage.

Bulk Leasequery allows the specification of a query-start-time as well as a query-end-time. Use of query-times allows a network element that periodically commits information to stable storage to recover just what it lost since the last commit.

4. Protocol Overview

The DHCPv4 Bulk Leasequery mechanism is modeled on the existing individual DHCPv4 Leasequery protocol in [RFC4388] as well as related work on DHCPv6 Bulk Leasequery [RFC5460]. A Bulk Leasequery requestor opens a TCP connection to a DHCPv4 Server, using the DHCPv4 port 67. Note that this implies that the Leasequery requestor has server IP address(es) available via configuration or some other means, and that it has unicast IP reachability to the DHCPv4 server. No relaying of Bulk Leasequery messages is specified.

After establishing a connection, the requestor sends a DHCPBULKLEASEQUERY message over the connection.

The server uses the message type and additional data in the DHCPv4 DHCPBULKLEASEQUERY message to identify any relevant bindings.

In order to support some query types, servers may have to maintain additional data structures or otherwise be able to locate bindings that have been requested by the Leasequery requestor.

Relevant bindings are returned in DHCPv4 packets with either the DHCPLEASEACTIVE message type for an IP address with a currently active lease or, in some situations, a DHCPLEASEUNASSIGNED message type for an IP address which is controlled by the DHCPv4 server but which is not actively leased by a DHCPv4 client at the present time.

The Bulk Leasequery mechanism is designed to provide an external entity with information concerning existing DHCPv4 IPv4 address bindings managed by the DHCPv4 server. When complete, the DHCPv4 server will send a DHCPLEASEQUERYDONE message. If a connection is lost while processing a Bulk Leasequery, the Bulk Leasequery must be retried as there is no provision for determining the extent of data already received by the requestor for a Bulk Leasequery.

Bulk Leasequery supports queries by MAC address and by Client Identifier in a way similar to [RFC4388]. The Bulk Leasequery protocol also adds several new queries.

- Query by Relay Identifier

This query asks a server for the bindings associated with a specific relay agent; the relay agent is identified by a DUID carried in a Relay-ID sub-option [RelayId]. Relay agents can include this sub-option while relaying messages to DHCPv4 servers. Servers can retain the Relay-ID and associate it with bindings made on behalf of the relay agent's clients. The bindings returned are only those for DHCPv4 clients with a currently active binding.

- o Query by Remote ID

  This query asks a server for the bindings associated with a Relay Agent Remote-ID sub-option [RFC3046] value. The bindings returned are only those for DHCPv4 clients with a currently active binding.

- o Query for All Configured IP Addresses

  This query asks a server for information concerning all IP addresses configured in that DHCPv4 server, by specifying no other type of query. In this case, the bindings returned are for all configured IP addresses, whether or not they contain a currently active binding to a DHCPv4 client, since one point of this type of query is to update an existing database with changes after a particular point in time.

Any of the above queries can be qualified by the specification of a query-start-time or a query-end-time (or both). When these timers are used as qualifiers, they indicate that a binding should be included if it changed on or after the query-start-time and on or before the query-end-time.

In addition, any of the above queries can be qualified by the specification of a vpn-id option [VpnId] to select the VPN on which the query should be processed. The vpn-id option is also extended to allow queries across all available VPNs. By default, only the default VPN is used to satisfy the query.

5. Interaction Between UDP Leasequery and Bulk Leasequery

Bulk Leasequery can be seen as an extension of the existing UDP Leasequery protocol [RFC4388]. This section clarifies the relationship between the two protocols.

Only the DHCPBULKLEASEQUERY request is supported over the Bulk Leasequery connection. No other DHCPv4 requests are supported. The Bulk Leasequery connection is not an alternative DHCPv4 communication

option for clients seeking other DHCPv4 services.

Two of the query-types introduced in the UDP Leasequery protocol can be used in the Bulk Leasequery protocol -- query by MAC address and query by client-id.

The contents of the reply messages are similar between the existing UDP Leasequery protocol and the Bulk Leasequery protocol, though more information is returned in the Bulk Leasequery messages.

One change in behavior for these existing queries is required when Bulk Leasequery is used. [RFC4388], in sections 6.1, 6.4.1, and 6.4.2 specifies the use of an associated-ip option in DHCPLEASEACTIVE messages in cases where multiple bindings were found. When Bulk Leasequery is used, this mechanism is not necessary; a server returning multiple bindings simply does so directly as specified in this document. The associated-ip option MUST NOT appear in Bulk Leasequery replies.

Implementors should note that the TCP message framing defined in Section 4.1 is not compatible with the UDP message format. If a TCP-framed request is sent as a UDP message, it may not be valid, because protocol fields will be offset by the message-size prefix.

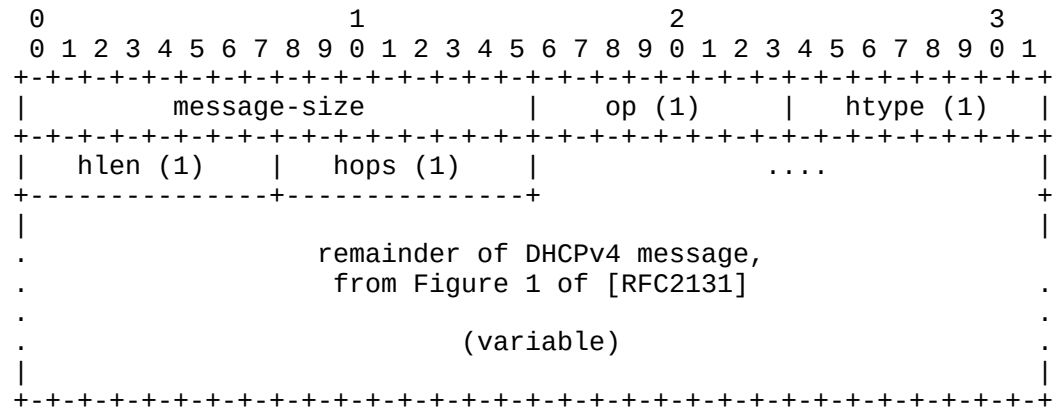
6. Message and Option Definitions

6.1. Message Framing for TCP

The use of TCP for the Bulk Leasequery protocol permits multiple messages to be sent from one end of the connection to the other without requiring a request/response paradigm as does UDP DHCPv4 [RFC2131]. The receiver needs to be able to determine the size of each message it receives. Two octets containing the message size in network byte-order are prepended to each DHCPv4 message sent on a Bulk Leasequery TCP connection. The two message-size octets ’frame’ each DHCPv4 message.

The maximum message size is 65535 octets.

DHCPv4 message framed for TCP:



message-size the number of octets in the message that follows, as a 16-bit integer in network byte-order.

All other fields are as specified in DHCPv4 [RFC2131].

Figure 2: Format of a DHCPv4 message in TCP

The intent in using this format is that code which currently knows how to deal with sending or receiving a message in [RFC2131] format will easily be able to deal with the message contained in the TCP framing.

6.2. New or Changed Options

The existing messages DHCPLEASEUNASSIGNED and DHCPLEASEACTIVE are used as the value of the dhcp-message-type option to indicate an IP address which is currently not leased or currently leased to a DHCPv4 client, respectively [RFC4388].

Additional options have also been defined to enable the Bulk Leasequery protocol to communicate useful information to the requestor.

6.2.1. dhcp-message-type

The dhcp-message-type option (option 53) from Section 9.6 of [RFC2132] requires new values. The values of these message types are

shown below in an extension of the table from Section 9.6 of [RFC2132]:

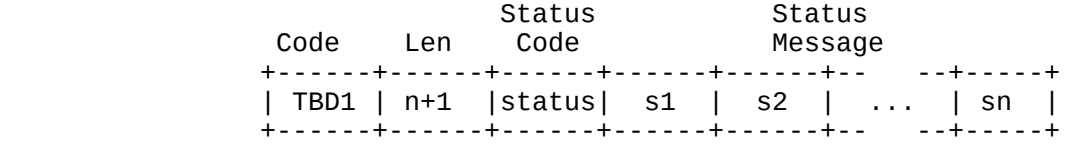
Value	Message Type
TBD8	DHCPBULKLEASEQUERY
TBD9	DHCPLEASEQUERYDONE

6.2.2. status-code

The status code option allows a machine readable value to be returned regarding the status of a DHCPBULKLEASEQUERY request.

This option has two possible scopes when used with Bulk Leasequery, depending on the context in which it appears. It refers to the information in a single Leasequery reply if the value of the dhcp-message-type is DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED. It refers to the message stream related to an entire request if the value of the dhcp-message-type is DHCPLEASEQUERYDONE.

The code for this option is TBD1. The length of this option is a minimum of 1 octet.



The status-code is an octet defined in the table below. The Status Message is an optional NVT ASCII encoded text string suitable for display to an end user, which MUST NOT be null-terminated.

Name	Status Code	Description
Success	000	Success. Also signaled by absence of a status-code option.
UnspecFail	001	Failure, reason unspecified.
QueryTerminated	002	Indicates that the server is unable to perform a query or has prematurely terminated the query for some reason (which should be communicated in the text message).
MalformedQuery	003	The query was not understood.
NotAllowed	004	The query or request was understood but was not allowed in this context.

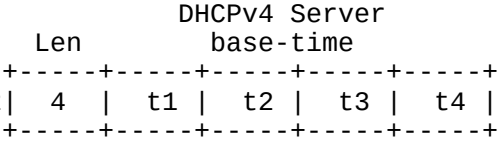
A status-code option MAY appear in the options field of a DHCPv4 message. If the status-code option does not appear, it is assumed that the operation was successful. The status-code option SHOULD NOT appear in a message which is successful unless there is some text string that needs to be communicated to the requestor.

6.2.3. base-time

The base-time option is the current time the message was created to be sent by the DHCPv4 server to the requestor of the Bulk Leasequery. This MUST be an absolute time. All of the other time based options in the reply message are relative to this time, including the dhcp-lease-time [RFC2132] and client-last-transaction-time [RFC4388]. This time is in the context of the DHCPv4 server who placed this option in a message.

This is an integer in network byte order.

The code for this option is TBD2. The length of this option is 4 octets.



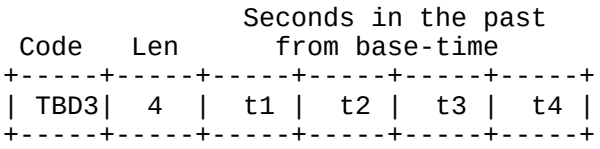
6.2.4. start-time-of-state

The start-time-of-state option allows the receiver to determine the time at which the IP address made the transition into its current state.

This MUST NOT be an absolute time, which is equivalent to saying that this MUST NOT be an absolute number of seconds since Jan 1, 1970. Instead, this MUST be the integer number of seconds from the time the IP address transitioned its current state to the time specified in the base-time option in the same message.

This is an integer in network byte order.

The code for this option is TBD3. The length of this option is 4 octets.



6.2.5. query-start-time

The query-start-time option specifies a start query time to the DHCPv4 server. If specified, only bindings that have changed on or after the query-start-time should be included in the response to the query.

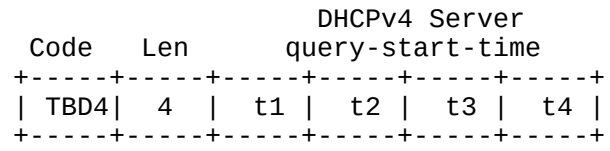
The requestor MUST determine the query-start-time using lease information it has received from the DHCPv4 server. This MUST be an absolute time in the DHCPv4 server's context (see Section 7.4).

Typically (though this is not a requirement) the query-start-time option will contain the value most recently received in a base-time option by the requestor, as this will indicate the last successful communication with the DHCP server.

This MUST be an absolute time.

This is an integer in network byte order.

The code for this option is TBD4. The length of this option is 4 octets.



6.2.6. query-end-time

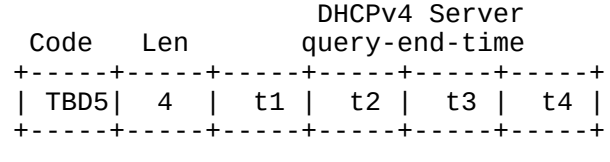
The query-end-time option specifies an end query time to the DHCPv4 server. If specified, only bindings that have changed on or before the query-end-time should be included in the response to the query.

The requestor MUST determine the query-end-time based on lease information it has received from the DHCPv4 server. This MUST be an absolute time in the context of the DHCPv4 server.

In the absence of information to the contrary, the requestor SHOULD assume that the time context of the DHCPv4 server is identical to the time context of the requestor (see Section 7.4).

This is an integer in network byte order.

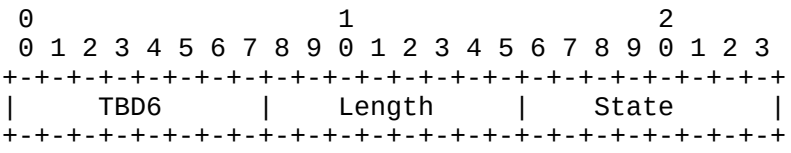
The code for this option is TBD5. The length of this option is 4 octets.



6.2.7. dhcp-state

The dhcp-state option allows greater detail to be returned than allowed by the DHCPLEASEACTIVE and DHCPLEASEUNASSIGNED message types.

The code for this option is TBD6. The length of this option is 1 octet.



TBD6 The option code.

Length The option length, 1 octet.

State The State of the IP address.

Value	State	
1	AVAILABLE	Address is available to local DHCPv4 server
2	ACTIVE	Address is assigned to a DHCPv4 client
3	EXPIRED	Lease has expired
4	RELEASED	Lease has been released by DHCPv4 client
5	ABANDONED	Server or client flagged address as unusable
6	RESET	Lease was freed by some external agent
7	REMOTE	Address is available to a remote DHCPv4 server
8	TRANSITIONING	Address is moving between states

Note that some of these states may be transient and may not appear in normal use. A DHCPv4 server MUST implement at least the AVAILABLE and ACTIVE states, and SHOULD implement at least the ABANDONED and RESET states.

Note the states AVAILABLE and REMOTE are relative to the current server. An address that is available to the current server should show AVAILABLE on that server, and if another server is involved with that address as well, on that other server it should show as REMOTE.

The dhcp-state option SHOULD contain ACTIVE when it appears in a DHCPLEASEACTIVE message. A DHCPv4 server MAY choose to not send a dhcp-state option in a DHCPLEASEACTIVE message, and a requestor SHOULD assume that the dhcp-state is ACTIVE if no dhcp-state option appears in a DHCPLEASEACTIVE message.

The reference to local and remote relate to possible use in an environment that includes multiple servers cooperating to provide an increased availability solution. In this case, an IP address with the state of AVAILABLE is available to the local server, while one with the state of REMOTE is available to a remote server. Usually, an IP address which is AVAILABLE on one server would be REMOTE on any remote server. The TRANSITIONING state is also likely to be useful in multiple server deployments, where sometimes one server must

interlock a state change with one or more other servers. Should a Bulk Leasequery need to send information concerning the state of the IP address during this period, it SHOULD use the TRANSITIONING state, since the IP address is likely to be neither ACTIVE or AVAILABLE.

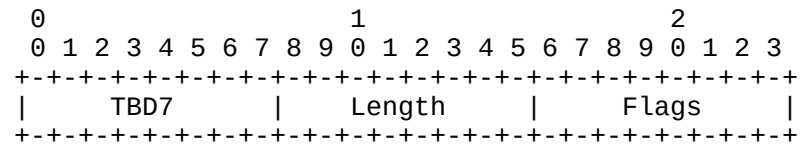
There is no requirement for the state of an IP address to transition in a well defined way from state to state. To put this another way, you cannot draw a simple state transition graph for the states of an IP address and the requestor of a Leasequery MUST NOT depend on one certain state always following a particular previous state. In general, every state can (at times) follow every other state.

6.2.8. data-source

The data-source option contains information about the source of the data in a DHCPLEASEACTIVE or a DHCPLEASEUNASSIGNED message. It is used when there are two or more servers who might have information about a particular IP address binding. Frequently two servers work together to provide an increased availability solution for the DHCPv4 service, and in these cases, both servers will respond to Bulk Leasequery requests for the same IP address.

The data contained in this option will allow an external process to better discriminate between the information provided by each of the servers servicing this IPv4 address.

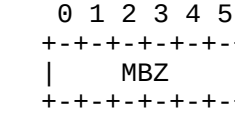
The code for this option is TBD7. The length of this option is 1 octet.



TBD7 The option code.

Length The option length, 1 octet.

Flags The Source information for this message.



R: REMOTE flag

remote = 1
local = 0

MBZ: MUST BE ZERO (reserved for future use)

The REMOTE flag is used to indicate where the most recent change of state (or other interesting change) concerning this IPv4 address took place. If the value is local, then the change took place on the server from which this message was transmitted. If the value is remote, then the change took place on some other server, and was made known to the server from which this message was transmitted.

If this option was requested and it doesn't appear, the requestor SHOULD consider that the data-source was local.

6.2.9. Virtual Subnet Selection Type and Information

All of the (sub)options defined in [VpnId] carry identical payloads, consisting of a type and additional VSS (Virtual Subnet Selection) information. The existing table is extended (see below) with a new type 254 to allow specification of a type code which indicates that all VPN's are to be used to process the Bulk Leasequery.

	Type	VSS Information format:
	0	NVT ASCII VPN identifier
	1	RFC 2685 VPN-ID
CHANGED ->	2-253	Not Allowed
NEW ->	254	All VPN's (wildcard)
	255	Global, default VPN

6.3. Connection and Transmission Parameters

DHCPv4 servers that support Bulk Leasequery SHOULD listen for incoming TCP connections on the DHCPv4 server port 67. Implementations MAY offer to make the incoming port configurable, but port 67 MUST be the default. Requestors SHOULD make TCP connections to port 67, and MAY offer to make the destination server port configurable.

This section presents a table of values used to control Bulk Leasequery behavior, including recommended defaults. Implementations MAY make these values configurable. However, configuring too-small timeout values may lead to harmful behavior both to this application as well as to other traffic in the network. As a result, timeout values smaller than the default values are NOT RECOMMENDED.

Parameter	Default	Description
BULK_LQ_DATA_TIMEOUT	300 secs	Bulk Leasequery data timeout for both client and server (see Sections 7 and 8)
BULK_LQ_MAX_CONNS	10	Max Bulk Leasequery TCP connections at the server side (see Section 8.1)

7. Requestor Behavior

7.1. Connecting and General Processing

A Requestor attempts to establish a TCP connection to a DHCPv4 Server in order to initiate a Leasequery exchange. If the attempt fails, the Requestor MAY retry.

If Bulk Leasequery is terminated prematurely by a DHCPLEASEQUERYDONE

with a status-code option with a status-code of QueryTerminated or by the failure of the connection over which it was being submitted, the requestor MAY retry the request after the creation of a new connection.

Messages from the DHCPv4 server come as multiple responses to a single DHCPBULKLEASEQUERY message. Thus, each DHCPBULKLEASEQUERY request MUST have an xid (transaction-id) unique on the connection on which it is sent. All of the messages which come as a response to that message will contain the same xid as the request. It is the xid which allows the data-streams of two different DHCPBULKLEASEQUERY requests to be demultiplexed by the requestor.

7.2. Forming a Bulk Leasequery

Bulk Leasequery is designed to create a connection which will transfer the state of some subset (or possibly all) of the IP address bindings from the DHCPv4 server to the requestor. The DHCPv4 server will send all of the requested IPv4 address bindings across this connection with minimal delay after it receives the request. In this context, "all IP address binding information" means information about all IPv4 addresses configured within the DHCPv4 server which meet the specified query criteria. For some query criteria, this may include IP address binding information for IP addresses which may not now have or ever had have an association with a specific DHCPv4 client.

To form the Bulk query, a DHCPv4 request is constructed with a dhcp-message-type of DHCPBULKLEASEQUERY. The query SHOULD have a dhcp-parameter-request-list to inform the DHCPv4 server which DHCPv4 options are of interest to the requestor sending the DHCPBULKLEASEQUERY message. The dhcp-parameter-request-list in a DHCPBULKLEASEQUERY message SHOULD contain the codes for base-time, dhcp-lease-time, start-time-of-state, and client-last-transaction-time.

A DHCPBULKLEASEQUERY request is constructed of one primary query and optionally one or more qualifiers for it.

The possible primary queries are listed below. Each DHCPBULKLEASEQUERY request MUST contain only one of these primary queries.

- Query by MAC address

  In a Query by MAC address, the chaddr, htype, and hlen of the DHCPv4 packet are filled in with the values requested.

- Query by Client-Id

In a Query by Client-Id, a dhcp-client-id option containing the requested value is included in the DHCPBULKLEASEQUERY request.

- Query by Remote-Id

  In a Query by Remote-Id, a remote-id sub-option containing the requested value is included in the relay-agent-information option of the DHCPBULKLEASEQUERY request.

- Query by Relay-Id

  In a Query by Relay-Id, a relay-id sub-option [RelayId] containing the requested value is included in the relay-agent-information option of the DHCPBULKLEASEQUERY request.

- Query for All Configured IP Addresses

  A Query for All Configured IP addresses is signaled by the absence of any other primary query.

There are three qualifiers which can be applied to any of the above primary queries. These qualifiers can appear individually or together in any combination, but only one of each can appear.

- Query Start Time

  Inclusion of a query-start-time option specifies that only IP address bindings which have changed on or after the time specified in the query-start-time option should be returned.

- Query End Time

  Inclusion of a query-end-time option specifies that only IP address bindings which have changed on or before the time specified in the query-end-time option should be returned.

- VPN Id

  If no vpn-id option appears in the DHCPBULKLEASEQUERY, the default VPN is used to search to satisfy the query specified by the DHCPBULKLEASEQUERY. Using the vpn-id option [VpnId] allows the requestor to specify a single VPN other than the default VPN. In addition, the vpn-id option has been extended as part of this document to allow specification that all configured VPN's be searched in order to satisfy the query specified in the DHCPBULKLEASEQUERY.

  In all cases, any message returned from a DHCPBULKLEASEQUERY

request containing information about an IP address for other than the default VPN MUST contain a vpn-id option in the message.

Use of the query-start-time or the query-end-time options or both can serve to reduce the amount of data transferred over the TCP connection by a considerable amount.

The TCP connection may become blocked or stop being writable while the requestor is sending its query. Should this happen, the implementation's behavior is controlled by the current value of BULK_LQ_DATA_TIMEOUT. The default value is given elsewhere in this document, and this value may be overridden by local configuration of the operator.

If this situation is detected, the requestor SHOULD start a timer using the current value of BULK_LQ_DATA_TIMEOUT. If that timer expires, the requestor SHOULD terminate the connection.

7.3. Processing Bulk Replies

The requestor attempts to read a DHCPv4 Leasequery reply message from the TCP connection.

The TCP connection may stop delivering reply data (i.e., the connection stops being readable). Should this happen, the implementation's behavior is controlled by the current value of BULK_LQ_DATA_TIMEOUT. The default value is given elsewhere in this document, and this value may be overridden by local configuration of the operator.

If this situation is detected, the requestor SHOULD start a timer using the current value of BULK_LQ_DATA_TIMEOUT. If that timer expires, the requestor SHOULD terminate the connection.

A single Bulk Leasequery can and usually will result in a large number of replies. The requestor MUST be prepared to receive more than one reply with an xid matching a single DHCPBULKLEASEQUERY message from a single DHCPv4 server. If the xid in the received message does not match an outstanding DHCPBULKLEASEQUERY message, the requestor MUST close the TCP connection.

The DHCPv4 server MUST send a server-identifier option (option 54) in the first response to any DHCPBULKLEASEQUERY message. The DHCPv4 server SHOULD NOT send server identifier options in subsequent responses to that DHCPBULKLEASEQUERY message. The requestor MUST cache the server-identifier option from the first response and apply it to any subsequent responses.

The response messages generated by a DHCPBULKLEASEQUERY request are:

- o DHCPLEASEACTIVE

  A Bulk Leasequery will generate DHCPLEASEACTIVE messages containing binding data for bound IP addresses which match the specified query criteria. The IP address which is bound to a DHCPv4 client will appear in the ciaddr field of the DHCPLEASEACTIVE message. The message may contain a non-zero chaddr, htype, and hlen and possibly additional options.

- o DHCPLEASEUNASSIGNED

  Some queries will also generate DHCPLEASEUNASSIGNED messages for IP addresses which match the query criteria. These messages indicate that the IP address is managed by the DHCPv4 server but is not currently bound to any DHCPv4 client. The IP address to which this message refers will appear in the ciaddr field of the DHCPLEASEUNASSIGNED message. A DHCPLEASEUNASSGINED message MAY also contain information about the last DHCPv4 client that was bound to this IP address. The message may contain a non-zero chaddr, htype, and hlen and possibly additional options in this case.

- o DHCPLEASEQUERYDONE

  A response of DHCPLEASEQUERYDONE indicates that the server has completed its response to the query, and that no more messages will be sent in response to the DHCPBULKLEASEQUERY. More details will sometimes be available in the received status-code option in the DHCPLEASEQUERYDONE message. If there is no status-code option in the DHCPLEASEQUERYDONE message, then the query completed successfully.

  Note that a query which returned no data, that is a DHCPBULKLEASEQUERY request followed by a DHCPLEASEQUERYDONE response, is considered a successful query in that no errors occurred during the processing. It is not considered an error to have no information to return to a DHCPBULKLEASEQUERY request.

The DHCPLEASEUNKNOWN message MUST NOT appear in a response to a Bulk Leasequery.

The requestor MUST NOT assume that there is any inherent order in the IP address binding information that is sent in response to a DHCPBULKLEASEQUERY. While the base-time will tend to increase monotonically (as it is the current time on the DHCPv4 server), the

actual time that any IP address binding information changed is unrelated to the base-time.

The DHCPLEASEQUERYDONE message always ends a successful DHCPBULKLEASEQUERY request and any unsuccessful DHCPBULKLEASEQUERY requests not terminated by a dropped connection. After receiving DHCPLEASEQUERYDONE from a server, the requestor MAY close the TCP connection to that server if no other DHCPBULKLEASEQUERY is outstanding on that TCP connection.

The DHCPv4 Leasequery protocol [RFC4388] uses the associated-ip option as an indicator that multiple bindings were present in response to a single DHCPv4 client based query. For Bulk Leasequery, a separate message is returned for each binding, and so the associated-ip option is not used.

7.4. Processing Time Values in Leasequery messages

Bulk Leasequery requests may be made to a DHCPv4 server whose absolute time may not be synchronized with the local time of the requestor. Thus, there are at least two time contexts in even the simplest Bulk Leasequery response, and in the situation where multiple DHCPv4 servers are queried, the situation becomes even more complex.

If the requestor of a Bulk Leasequery is saving the data returned in some form, it has a requirement to store a variety of time values, and some of these will be time in the context of the requestor and some will be time in the context of the DHCPv4 server.

When receiving a DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED message from the DHCPv4 server, the message will contain a base-time option. The time contained in this base-time option is in the context of the DHCPv4 server. As such, it is an ideal time to save and use as input to a DHCPBULKLEASEQUERY in the query-start-time or query-end-time options, should the requestor need to ever issue a DHCPBULKLEASEQUERY message using those options as part of a later query, since those options require a time in the context of the DHCPv4 server.

In addition to saving the base-time for possible future use in a query-start-time or query-end-time option, the base-time is used as part of the conversion of the other times in the Leasequery message to values which are meaningful in the context of the requestor.

In systems whose clocks are synchronized, perhaps using NTP, the clock skew will usually be zero, which is not only acceptable, but desired.

7.5. Querying Multiple Servers

A Bulk Leasequery requestor MAY be configured to attempt to connect to and query from multiple DHCPv4 servers in parallel. The DHCPv4 Leasequery specification [RFC4388] includes a discussion about reconciling binding data received from multiple DHCPv4 servers.

In addition, the algorithm in Section 7.6 should be used.

7.6. Making Sense Out of Multiple Responses Concerning a Single IPv4 Address

Any requestor of an Bulk Leasequery MUST be prepared for multiple responses to arrive for a particular IPv4 address from multiple different DHCPv4 servers. The following algorithm SHOULD be used to decide if the information just received is more up to date (i.e., better) than the best existing information. In the discussion below, the information that is received from a DHCPv4 server about a particular IPv4 address is termed a "record". The times used in the algorithm below SHOULD have been converted into the requestor's context and the time comparisons SHOULD be performed in a manner consistent with the information in Section 7.4.

- If both the existing and the new record contain client-last-transaction-time information, the record with the later client-last-transaction-time is considered better.

- If one of the records contains client-last-transaction-time information and the other one doesn't, then compare the client-last-transaction-time in the record that contains it against the other record's start-time-of-state. The record with the later time is considered better.

- If neither record contains client-last-transaction-time information, compare their start-time-of-state information. The record with the later start-time-of-state is considered better.

- If none of the comparisons above yield a clear answer as to which record is later, then compare the value of the REMOTE flag from the data-source option for each record.

  If the values of the REMOTE flag are different between the two records, the record with the REMOTE flag value of local is considered better.

The above algorithm does not necessarily determine which record is better. In the event that the algorithm is inconclusive with regard to a record which was just received by the requestor, the requestor

SHOULD use additional information in the two records to make a determination as to which record is better.

7.7. Multiple Queries to a Single Server over One Connection

Bulk Leasequery requestors may need to make multiple queries in order to recover binding information. A requestor MAY use a single connection to issue multiple queries to a server willing to support them. Each query MUST have a unique xid.

A server SHOULD allow configuration of the number of queries that can be processed simultaneously over a single connection. A server SHOULD read the number of queries it is configured to process simultaneously and only read any subsequent queries as current queries are processed.

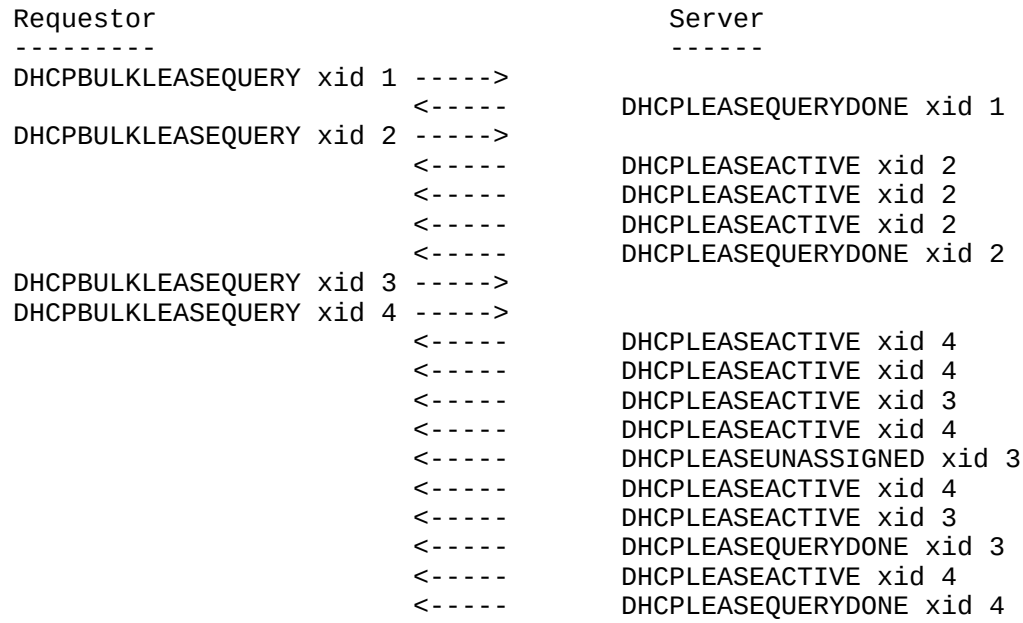
A server that is processing multiple queries simultaneously MUST interleave replies to the multiple queries within the stream of reply messages it sends. Requestors need to be aware that replies for multiple queries may be interleaved within the stream of reply messages. Requestors that are not able to process interleaved replies (based on xid) MUST NOT send more than one query over a single connection prior to the completion of the previous query.

Requestors should be aware that servers are not required to process more than one query over a connection at a time (the limiting case for the configuration described above), and that servers are likely to limit the rate at which they process queries from any one requestor.

7.7.1. Example

This example illustrates what a series of queries and responses might look like. This is only an example - there is no requirement that this sequence must be followed, or that requestors or servers must support parallel queries.

In the example session, the client sends four queries after establishing a connection. Query 1 returns no results; query 2 returns 3 messages and the stream of replies concludes before the client issues any new query. Query 3 and query 4 overlap, and the server interleaves its replies to those two queries.



7.8. Closing Connections

The requestor SHOULD close the connection after the DHCPLEASEQUERYDONE message is received for the last outstanding query that it has sent, if it has no more queries to send.

8. Server Behavior

8.1. Accepting Connections

Servers that implement DHCPv4 Bulk Leasequery listen for incoming TCP connections. Port numbers are discussed in Section 6.3. Servers MUST be able to limit the number of concurrently accepted and active connections. The value BULK_LQ_MAX_CONNS SHOULD be the default; implementations MAY permit the value to be configurable. Connections SHOULD be accepted and, if the number of connections is over BULK_LQ_MAX_CONNS, they SHOULD be closed immediately.

Servers MAY restrict Bulk Leasequery connections and DHCPBULKLEASEQUERY messages to certain requestors. Connections not from permitted requestors SHOULD be closed immediately, to avoid server connection resource exhaustion. Servers MAY restrict some requestors to certain query types. Servers MAY reply to queries that

are not permitted with the DHCPLEASEQUERYDONE message with a status-code option status of NotAllowed, or MAY simply close the connection.

If the TCP connection becomes blocked while the server is accepting a connection or reading a query, it SHOULD be prepared to terminate the connection after an BULK_LQ_DATA_TIMEOUT. We make this recommendation to allow servers to control the period of time they are willing to wait before abandoning an inactive connection, independent of the TCP implementations they may be using.

8.2. Replying to a Bulk Leasequery

If the connection becomes blocked while the server is attempting to send reply messages, the server SHOULD be prepared to terminate the TCP connection after BULK_LQ_DATA_TIMEOUT.

Every Bulk Leasequery request MUST be terminated by sending a final DHCPLEASEQUERYDONE message if such a message can be sent. The DHCPLEASEQUERYDONE message MUST have a status-code option status if the termination was other than successful, and SHOULD NOT contain a status-code option status if the termination was successful.

If the DHCPv4 server encounters an error during processing of the DHCPBULKLEASEQUERY message, either during initial processing or later during the message processing, it SHOULD send a DHCPLEASEQUERYDONE containing a status-code option. It MAY close the connection after this error is signaled, but that is not required.

If the server does not find any bindings satisfying a query, it MUST send a DHCPLEASEQUERYDONE. It SHOULD NOT include a status-code option with a Success status unless there is a useful string to include in the status-code option. Otherwise, the server sends each binding’s data in a DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED message.

The response to a DHCPBULKLEASEQUERY may involve examination of multiple DHCPv4 IP address bindings maintained by the DHCPv4 server. The Bulk Leasequery protocol does not require any ordering of the IP addresses returned in DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED messages.

When responding to a DHCPBULKLEASEQUERY message, the DHCPv4 server MUST NOT send more than one message for each applicable IP address, even if the state of some of those IP addresses changes during the processing of the message. Updates to such IP address state are already handled by normal protocol processing, so no special effort is needed here.

If the ciaddr, yiaddr, or siaddr is non-zero in a DHCPBULKLEASEQUERY

request, the request must be terminated immediately by a DHCPLEASEQUERYDONE message with a status-code status of MalformedQuery.

Any DHCPBULKLEASEQUERY which has more than one of the following primary query types specified MUST be terminated immediately by a DHCPLEASEQUERYDONE message with a status-code option status code of NotAllowed.

The allowable queries in a DHCPBULKLEASEQUERY message are processed as follows. Note that the descriptions of the primary queries below must be constrained by the actions of any of the three qualifiers described subsequently as well.

The following table discusses how to process the various queries. For information on how to identify the query, see the information in Section 7.2.

- o Query by MAC address

  Every IP address that has a current binding to a DHCPv4 client matching the chaddr, htype, and hlen in the DHCPBULKLEASEQUERY request MUST be returned in a DHCPLEASEACTIVE message.

- o Query by Client-Id

  Every IP address that has a current binding to a DHCPv4 client matching the client-id option in the DHCPBULKLEASEQUERY request MUST be returned in a DHCPLEASEACTIVE message.

- o Query by Remote-Id

  Every IP address that has a current binding to a DHCPv4 client matching the remote-id sub-option of the relay-agent-information option in the DHCPBULKLEASEQUERY request MUST be returned in a DHCPLEASEACTIVE message.

- o Query by Relay-Id

  Every IP address that has a current binding to a DHCPv4 client matching the relay-id sub-option of the relay-agent-information option in the DHCPBULKLEASEQUERY request MUST be returned in a DHCPLEASEACTIVE message.

- o Query for All Configured IP Addresses

  A Query for All Configured IP addresses is signaled by the

absence of any other primary query. That is, if there is no value in the chaddr, hlen, htype, no client-id option, no remote-id sub-option or relay-id sub-option of the relay-agent-information option, then the request is a query for information concerning all configured IP addresses. In this case, every configured IP address that has a current binding to a DHCPv4 client MUST be returned in a DHCPLEASEACTIVE message. In addition, every configured IP address that does not have a current binding to a DHCPv4 client MUST be returned in a DHCPLEASEUNASSIGNED message.

In this form of query, each configured IP address MUST be returned at most one time. If the absence of qualifiers restricting the number of IP addresses returned, every configured IP address MUST be returned exactly once.

There are three qualifiers that can be applied to any of the above primary queries. These qualifiers can appear individually or together in any combination, but only one of each can appear.

o Query Start Time

If a query-start-time option appears in the DHCPBULKLEASEQUERY request, only IP address bindings that have changed on or after the time specified in the query-start-time option should be returned.

o Query End Time

If a query-end-time option appears in the DHCPBULKLEASEQUERY request, only IP address bindings that have changed on or before the time specified in the query-end-time option should be returned.

o VPN Id

If no vpn-id option appears in the DHCPBULKLEASEQUERY, the default VPN is used to satisfy the query. A vpn-id option [VpnId] value other than the wildcard value (254) allows the requestor to specify a single VPN other than the default VPN. In addition, the vpn-id option has been extended as part of this document to allow specification of a type 254 which indicates that all configured VPN’s be searched in order to satisfy the primary query.

In all cases, if the information returned in a DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED message is for a VPN other than the default, a vpn-id option MUST appear in the packet.

The query-start-time and query-end-time qualifiers are used to constrain the amount of data returned by a Bulk Leasequery request by

returning only IP addresses whose address bindings have changed in some way during the time window specified by the query-start-time and query-end-time.

A DHCPv4 server SHOULD consider an address binding to have changed during a specified time window if either the client-last-transaction-time or the start-time-of-state of the address binding changed during that time window.

The DHCPv4 server MAY return address binding data in any order, as long as binding information for any given IP address is not repeated. When all binding data for a given DHCPBULKLEASEQUERY has been sent, the DHCPv4 server MUST send a DHCPBULKLEASEQUERYDONE message.

8.3. Building a Single Reply for Bulk Leasequery

The DHCPv4 Leasequery [RFC4388] specification describes the initial construction of DHCPLEASEQUERY reply messages using the DHCPLEASEACTIVE and DHCPLEASEUNASSIGNED message types in Section 6.4.2. All of the reply messages in Bulk Leasequery are similar to the reply messages for an IP address query. Message transmission and framing for TCP is described in this document in Section 6.1.

[RFC2131] and [RFC4388] specify that every response message MUST contain the server-identifier option. However, that option will be the same for every response from a particular DHCPBULKLEASEQUERY request. Thus, the DHCPv4 server MUST include the server-identifier option in the first message sent in response to a DHCPBULKLEASEQUERY. It SHOULD NOT include the server-identifier in later messages.

The message type of DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED is based on the value of the dhcp-state option. If the dhcp-state option value is ACTIVE, then the message type is DHCPLEASEACTIVE, otherwise the message type is DHCPLEASEUNASSIGNED.

In addition to the basic message construction described in [RFC4388], the following guidelines exist:

1. If the dhcp-state option code appears in the dhcp-parameter-request-list, the DHCPv4 server SHOULD include a dhcp-state option whose value corresponds most closely to the state held by the DHCPv4 server for the IP address associated with this reply. If the state is ACTIVE and the message being returned is DHCPLEASEACTIVE, then the DHCPv4 server MAY choose to not send the dhcp-state option. The requestor SHOULD assume that any DHCPLEASEACTIVE message arriving without a requested dhcp-state option has a dhcp-state of ACTIVE.

2. If the base-time option code appears in the dhcp-parameter-request-list, the DHCPv4 server MUST include a base-time option, which is the current time in the DHCPv4 server's context and the time from which the start-time-of-state, dhcp-lease-time, client-last-transaction-time, and other duration-style times are based upon.

3. If the start-time-of-state option code appears in the dhcp-parameter-request-list, the DHCPv4 server MUST include a start-time-of-state option whose value represents the time at which the dhcp-state option's state became valid.

4. If the dhcp-lease-time option code appears in the dhcp-parameter-request-list, the DHCPv4 server MUST include a dhcp-lease-time option for any state that has a time-out value associated with it.

5. If the data-source option code appears in the dhcp-parameter-request-list, the DHCPv4 server MUST include the data-source option in any situation where any of the bits would be non-zero. Thus, in the absence of the data-source option, the assumption is that all of the flags were zero.

6. If the client-last-transaction-time option code appears in the dhcp-parameter-request-list, The DHCPv4 server MUST include the client-last-transaction-time option in any situation where the information is available.

7. If there is a dhcp-parameter-request-list in the initial DHCPBULKLEASEQUERY request, then it should be used for all of the replies generated by that request. Some options can be sent from a DHCPv4 client to the server or from the DHCPv4 server to a DHCPv4 client. Option 125 is such an option. If the option code for one of these options appears in the dhcp-parameter-request-list, it SHOULD result in returning the value of the option sent by the DHCPv4 client to the server if one exists.

Note that there may be other requirements for a reply to a DHCPBULKLEASEQUERY request discussed in Section 8.2.

8.4. Multiple or Parallel Queries

As discussed in Section 7.3, requestors may want to use a connection that has already been established when they need to make additional queries. Servers SHOULD support reading and processing multiple queries from a single connection and SHOULD allow configuration of the number of simultaneous queries it may process. A server MUST NOT

read more query messages from a connection than it is prepared to process simultaneously.

This SHOULD be a feature that is administratively controlled. Servers SHOULD offer configuration that limits the number of simultaneous queries permitted from any one requestor, in order to control resource use if there are multiple requestors seeking service.

8.5. Closing Connections

The DHCPv4 server SHOULD start a timer for BULK_LQ_DATA_TIMEOUT seconds for a particular connection after it sends a DHCPLEASEQUERYDONE message over that connection and if there is no current query outstanding for that connection. It should clear this timer if a query arrives over that connection. If the timer expires, the DHCPv4 server should close the connection.

The server MUST close its end of the TCP connection if it encounters an error sending data on the connection. The server MUST close its end of the TCP connection if it finds that it has to abort an in-process request. A server aborting an in-process request SHOULD attempt to signal that to its requestors by using the QueryTerminated status code in the status-code option in a DHCPLEASEQUERYDONE message, including a message string indicating details of the reason for the abort. If the server detects that the requesting end of the connection has been closed, the server MUST close its end of the connection.

9. Security Considerations

The "Security Considerations" section of [RFC2131] details the general threats to DHCPv4. The DHCPv4 Leasequery specification [RFC4388] describes recommendations for the Leasequery protocol, especially with regard to authentication of LEASEQUERY messages, mitigation of packet-flooding DOS attacks, and restriction to trusted requestors.

The use of TCP introduces some additional concerns. Attacks that attempt to exhaust the DHCPv4 server’s available TCP connection resources, such as SYN flooding attacks, can compromise the ability of legitimate requestors to receive service. Malicious requestors who succeed in establishing connections, but who then send invalid queries, partial queries, or no queries at all also can exhaust a server’s pool of available connections. We recommend that servers offer configuration to limit the sources of incoming connections, that they limit the number of accepted connections and the number of

in-process queries from any one connection, and that they limit the period of time during which an idle connection will be left open.

[RFC4388] discusses security concerns and potential solutions for DHCPLEASEQUERY message exchanges in its Section 7, and all of the solutions discussed there are applicable to the DHCPLEASEQUERY message exchanges described in this document.

10. IANA Considerations

This document defines two new name spaces associated with DHCPv4 options:

1. Status code values for the status-code option, TBD1.

2. DHCP state values for the dhcp-state option, TBD6.

IANA has established a registry of values for these two name spaces. These name spaces will be managed by IANA. New values for these name spaces may only be defined by IETF Consensus, as described in [RFC5226]. Basically, this means that they are defined by RFCs approved by the IESG.

IANA is requested to assign the following new values for this document. See Section 6.2 for details.

1. A dhcp-message-type of TBD8 for DHCPBULKLEASEQUERY.

2. A dhcp-message-type of TBD9 for DHCPLEASEQUERYDONE.

3. An option code of TBD1 for status-code.

4. An option code of TBD2 for base-time.

5. An option code of TBD3 for start-time-of-state.

6. An option code of TBD4 for query-start-time.

7. An option code of TBD5 for query-end-time.

8. An option code of TBD6 for dhcp-state.

9. An option code of TBD7 for data-source.

10. Values for status code in a status-code option (option TBD1):

Name	status-code
Success	000
UnspecFail	001
QueryTerminated	002
MalformedQuery	003
NotAllowed	004

11. Values for dhcp-state (option TBD6):

State	
1	AVAILABLE
2	ACTIVE
3	EXPIRED
4	RELEASED
5	ABANDONED
6	RESET
7	REMOTE
8	TRANSITIONING

12. Additional type field values for the Virtual Subnet Selection Type and Information [VpnId]:

Type	VSS Information format:
0	NVT ASCII VPN identifier
1	RFC2685 VPN-ID
2-253	Not Allowed
254	All VPN's. (wildcard; only allowed in DHCPBULKLEASEQUERY messages)
255	Global, default VPN.

11. Acknowledgements

This draft is a collaboration between the authors of draft-dtv-dhc-dhcpv4-bulk-leasequery-00.txt and draft-kkinnear-dhc-dhcpv4-bulk-leasequery-00.txt. Both documents acknowledged that significant text as well as important ideas were borrowed in whole or in part from the DHCPv6 Bulk Leasequery RFC, [RFC5460] written by Mark Stapp. Further suggestions and improvements were made by participants in the DHC working group, including Alfred Hoenes.

12. References

12.1. Normative References

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2132] Alexander, S., Droms, R., "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.

[RFC3046] Patrick, M., "DHCP Relay Agent Information Option", RFC 3046, January 2001.

[RFC4388] Woundy, R., K. Kinnear, "Dynamic Host Configuration Protocol (DHCP) Leasequery", RFC 4388, February 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RelayId] Stapp, M., "The DHCPv4 Relay Agent Identifier Suboption", draft-ietf-dhc-relay-id-suboption-07.txt, (work in progress) July 2009.

[VpnId] Kinnear, K., R. Johnson, M. Stapp and J. Kumarasamy, "Virtual Subnet Selection Options for DHCPv4 and DHCPv6" draft-ietf-dhc-vpn-option-11.txt, (work in progress) March 2009.

12.2. Informative References

[RFC951] Croft, B., Gilmore, J., "Bootstrap Protocol (BOOTP)", RFC 951, September 1985.

[RFC1542] Wimer, W., "Clarifications and Extensions for the Bootstrap Protocol", RFC 1542, October 1993.

[RFC4614] Duke, M., R. Braden, W. Eddy, and E. Blanton, "A Roadmap for Transmission Control Protocol (TCP) Specification Documents", RFC 4614, September 2006.

[RFC5460] Stapp, M., "DHCPv6 Bulk Leasequery", RFC 5460, February 2009.

Authors' Addresses

Kim Kinnear
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: kkinnear@cisco.com

Bernie Volz
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: volz@cisco.com

Neil Russell
Nokia
5 Wayside Drive
Burlington, MA 01803

EMail: neil.russell@nokia.com

Mark Stapp
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: mjs@cisco.com

Ramakrishna Rao DTV
Infosys Technologies Ltd.
44 Electronics City, Hosur Road

Bangalore 560 100
India

EMail: ramakrishnadtv@infosys.com
URI: http://www.infosys.com/

Bharat Joshi
Infosys Technologies Ltd.
44 Electronics City, Hosur Road
Bangalore 560 100
India

EMail: bharat_joshi@infosys.com
URI: http://www.infosys.com/

Pavan Kurapati
Juniper Networks Ltd.
Embassy Prime Buildings, C.V.Raman Nagar
Bangalore 560 093
India

Email: kurapati@juniper.net
URI: http://www.juniper.net/

dhc
Internet-Draft
Intended status: Standards Track
Expires: April 19, 2011

T. Lemon
Nominum, Inc.
H. Deng
China Mobile

October 16, 2010

Relay Agent Encapsulation for DHCPv4
draft-ietf-dhc-dhcpv4-relay-encapsulation-00

Abstract

This document describes a general mechanism whereby DHCP relay agents can encapsulate DHCP packets that they are forwarding in the direction of DHCP servers, and decapsulate packets that they they are forwarding toward DHCP clients, so that more than one relay agent can insert relay agent suboptions into the forwarding chain.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 19, 2011.

described in the Simplified BSD License.

Table of Contents

1. Introduction

In some networking environments, it is useful to be able to configure relay agents in a hierarchy, so that information from a relay agent close to the client can be combined with information from one or more relay agents closer to the server, particularly in cases where the relay agents may be in separate administrative domains.

The current Relay Agent Information Option (RAIO) specification [RFC3046] specifies that when a relay agent receives a packet containing an RAIO, it must not add an RAIO. This prevents chaining of RAIOs, and hence prohibits the hierarchical use case.

DHCP version 6 [RFC3315] provides a much cleaner technique for providing RAIO suboptions to the DHCP server. Rather than appending an information option to the DHCP client's message, the relay agent encapsulates the DHCP client message in a new DHCP message which it sends to the DHCP server along with any options it chooses to add to provide information to the DHCP server.

This document specifies a mechanism for providing the same functionality in DHCPv4.

1.1. Requirements Language

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

1.2. Terminology

The following terms and acronyms are used in this document:

BOOTREPLY message	Any DHCP or BOOTP message in which the 'op' field is set to BOOTREPLY.
BOOTREQUEST message	Any DHCP or BOOTP message in which the 'op' field is set to BOOTREQUEST.
DHCP	Dynamic Host Configuration Protocol Version 4 [RFC2131]
decapsulate	the act of de-encapsulating DHCP packets being relayed from DHCP servers or relay agents in the direction of DHCP clients, according to this specification.

encapsulate	the act of encapsulating DHCP packets that are being relayed from DHCP clients or relay agents toward DHCP servers, according to the method described in this specification.
encapsulating relay agent	a relay agent that implements this specification and is configured to encapsulate.
L2RA	Layer 2 Relay Agent--a relay agent that doesn't have an IP address reachable by the DHCP server.
L3RA	Layer 3 Relay Agent--a relay agent that has an IP address reachable by the DHCP server.
option buffer	the portion of the DHCP packet following the magic cookie in the 'vend' field of the DHCP Packet.
RAIO	Relay Agent Information Option [RFC3046]. Also commonly referred to as "Option 82."
RAIO suboption	a DHCP suboption that has been defined for encapsulation in the Relay Agent Information Option
relay message	a RELAYFORWARD or RELAYREPLY message.
RELAYFORWARD message	Any DHCP or BOOTP message in which the 'op' field is set to RELAYFORWARD.
RELAYREPLY message	Any DHCP or BOOTP message in which the 'op' field is set to RELAYREPLY.
silently discard	in many places in this specification, the implementation is required to silently discard erroneous messages. What is meant by 'silently discard' is that the implementation MUST NOT send any ICMP message indicating that the delivery was in error. It may be desirable for the implementation to keep a count of messages that have been discarded, either by message type or by reason for discarding, or some combination. Nothing in this specification should be construed to forbid such data collection.

2. Protocol Summary

This document specifies two new BOOTP message types: the RELAYFORWARD message, and the RELAYREPLY message. These messages are analogous to the Relay Forward and Relay Reply messages in DHCPv6 [RFC3315].

Although this specification is generally aimed at DHCP implementations, it is not specifically restricted to DHCP, and is applicable to BOOTP in cases where the BOOTP server is a DHCP server that implements this specification, or the less likely case that the BOOTP server only supports the BOOTP protocol, but still implements this specification.

In general, when the term "DHCP" appears in this specification, the reader should not read this as intending to exclude BOOTP.

2.1. RELAYFORWARD Message

Conforming relay agents encapsulate messages being sent toward DHCP servers in RELAYFORWARD messages.

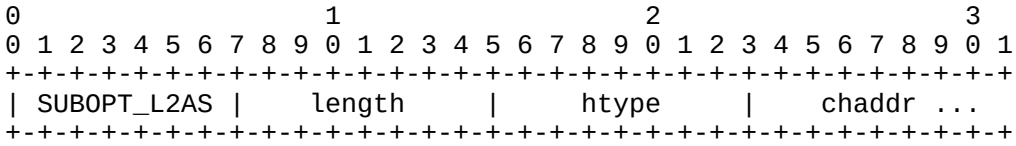
2.2. RELAYREPLY Message

A conforming DHCP server encapsulates any message being sent toward a DHCP client in a RELAYREPLY message, if the message being responded to was encapsulated.

A conforming relay agent, when it receives a RELAYREPLY message, decapsulates the message contained in the RELAYREPLY message and sends it to the next relay or to the client.

2.3. Layer Two Address suboption

In cases where the closest relay agent to the client is an L2RA, but where there is an L3RA on the path to the client, the DHCP server will encode the link layer address that would normally go in the chaddr field of the DHCP packet into a Layer Two Address suboption.



The Layer Two Address suboption has four fields:

SUBOPT_L2AS One octet: the suboption code for the Layer Two Address suboption (TBD).

length One octet: the length of the Layer Two Address suboption.

htype One octet: the type of the Layer Two Address suboption-- corresponds to the 'htype' field in a non-relay DHCP or BOOTP message.

chaddr One or more octets: the layer two address of the client, from the 'chaddr' field of the DHCP or BOOTP message.

3. Encoding

RELAYFORWARD and RELAYREPLY messages are distinguished through the use of the 'op' field of the DHCP packet.

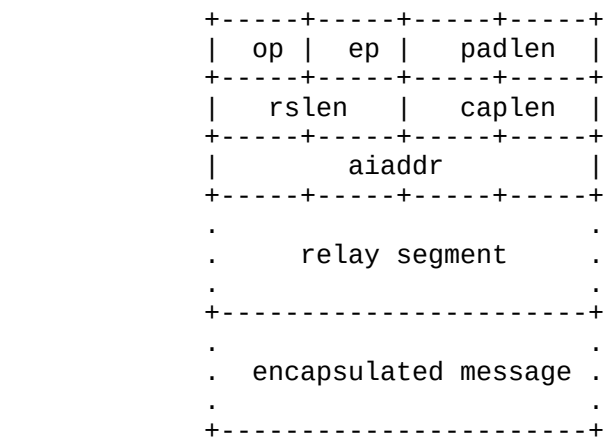
In non-relay DHCP packets, the 'op' field either contains BOOTREQUEST, for any DHCP message from the client to the server, or BOOTREPLY, for any DHCP message from the server to the client.

This document defines two additional codes, RELAYFORWARD and RELAYREPLY. Conforming DHCP servers and DHCP relay agents MUST support these two new values for the 'op' field. DHCP clients should never see either value.

code	meaning
1	BOOTREQUEST
2	BOOTREPLY
3	RELAYFORWARD
4	RELAYREPLY

Values for the 'op' field

RELAYFORWARD and RELAYREPLY messages share only the 'op' field in common with other DHCP and BOOTP messages. The remainder of the message consists of a series of fixed-length fields followed by two variable-length fields: the relay segment, and the encapsulated message.



3.1. The fixed-length header

The fixed-length header of the relay message contains a series of fields that perform two purposes: to provide enough information that the DHCP server can reconstruct the original packet sent by the DHCP client, and to establish the lengths of the two variable-length segments.

To satisfy the first of these requirements, two fields in the fixed-length header report the amount of padding stripped from the client message, if any, and whether or not an end option was stripped from the client message. Except for a relay message that immediately encapsulates a message from a DHCP client, these fields are always zero. Using these two fields, the DHCP server can reconstruct the client packet exactly, and this allows the DHCP server to validate any signature [RFC3118] that may be present.

The fixed-length header consists of five fields:

op The BOOTP 'op' field, which, for a relay message, MUST contain the RELAYFORWARD or RELAYREPLY code.

ep If an End option was present in the option buffer prior to encapsulation, this field is set to 1; otherwise, it is set to 0. This field is a single byte.

padlen The length of any padding that was removed from the option buffer prior to encapsulation: two bytes in network byte order.

rslen The length of the relay segment: two byte in network byte order.

caplen The length of the encapsulation segment: two byte in network byte order.

aiaddr Relay agent IP address.

3.2. Relay Segment

The relay segment contains any RAIO suboptions that the encapsulating agent (the relay agent or the DHCP server) wishes to send. End and Pad options MUST NOT appear within the relay segment.

3.3. Encapsulation Segment

The encapsulation segment contains the entire DHCP message being encapsulated, with four exceptions:

- The encapsulating agent MUST omit the IP and UDP headers, as well as any layer two header, from the encapsulated message.
- The encapsulating agent MUST omit any options following the first End option in the option buffer. These options are assumed to be garbage, and are not covered by any signature [RFC3118].
- The encapsulating MUST omit any Pad options present either at the end of the option buffer, or prior to the first End packet, that are followed only by other Pad options or a single End option. The encapsulating agent MUST record number of Pad options that were omitted in the 'padlen' field of the message header.
- The encapsulating agent MUST omit the End option, if present. The encapsulating agent MUST set the 'ep' field in the message header to 1 if an End option was present in the option buffer, and to zero if no End option was present.

These exceptions apply only to the option buffer. The encapsulating agent MUST NOT modify the contents of the 'file' and 'sname' fields. The encapsulating agent MUST NOT count End or Pad options that appear in these fields.

4. DHCP Relay Agent Behavior

DHCP Relay agents implementing this specification MUST have a configuration parameter controlling relay encapsulation. By default, relay encapsulation MUST be disabled.

Relay agents with encapsulation disabled MUST NOT encapsulate. Relay agents with encapsulation disabled MUST NOT decapsulate.

In any case where a relay agent implementing this specification does not encapsulate or decapsulate, it MUST behave exactly as a relay agent would that did not implement this specification at all.

DHCP relay agents that are configured with encapsulation enabled, but which have no agent-specific options to send to the DHCP server, MUST encapsulate. Relay agents that are configured with encapsulation enabled MUST decapsulate.

Layer two relay agents MUST silently discard any messages that contains an IPsec authentication header [RFC4302]. This is because they cannot modify such packets, but also cannot detect that a message from the DHCP server is in response such a message, since it might not contain an IPsec authentication header.

4.1. Packet processing

Relay agents implementing this specification may receive packets directed toward DHCP servers with a source port of 67 (BOOTPS). Therefore, the source port cannot be used to determine whether the packet is traveling toward a DHCP server or toward a DHCP client.

In order to determine whether a message is traveling toward a DHCP client or toward a DHCP server, the relay agent must check the 'op' field of the DHCP message. If the 'op' field is set to BOOTREQUEST or RELAYFORWARD, the message is traveling toward a DHCP server. If the 'op' field is set to BOOTREPLY or RELAYREPLY, the message is traveling toward a DHCP client. At the time of the writing of this specification, no other value is meaningful in the 'op' field.

Relay agents implementing this specification MUST NOT encapsulate or decapsulate messages with other values in the 'op' field. It is assumed that if meanings are defined for additional values, the document that specifies the meaning of those values will update this document; in the absence of such an update, the behavior specified here will remain in effect.

Relay agents implementing this specification MAY differentiate between DHCP and BOOTP messages. Under normal circumstances, BOOTP and DHCP messages are forwarded to the same server, which should be able to successfully decapsulate both DHCP and BOOTP messages. However, some relay agents may send BOOTP and DHCP packets to different servers; this document should not be construed to require that such a relay agent should encapsulate all messages regardless of protocol.

4.1.1. Packets traveling toward DHCP servers

Any DHCP or BOOTP packet with an 'op' value of BOOTREQUEST or RELAYFORWARD is traveling toward a DHCP server. When a DHCP relay agent that is configured to encapsulate receives such a packet, the relay agent MUST encapsulate that packet into a RELAYFORWARD message and send it to the address or addresses with which it is configured to forward messages intended for DHCP servers.

4.1.2. Packets traveling toward DHCP clients

Any DHCP or BOOTP packet with an 'op' value of BOOTREPLY or RELAYREPLY is traveling toward a DHCP client. When a DHCP relay agent that is configured to encapsulate recieves a RELAYREPLY message that is traveling toward a DHCP or BOOTP client, the relay agent MUST decapsulate that message and forward the decapsulated message toward the client.

4.1.3. Anti-spoofing

Because this specification allows for chaining of relay agent-supplied information, it is now possible for a relay agent outside of the trusted portion of a network to communicate relay agent information to the DHCP server without preventing the legitimate relay from communicating return path information to the DHCP server, as is the case with RFC3046.

In order to prevent this sort of spoofing, relay agents implementing this specification MUST be configurable to discard all RELAYFORWARD messages that they receive. Administrators relying on a trusted network architecture to control the flow of information to the DHCP server SHOULD configure relay agents on the edge of their networks to discard RELAYFORWARD messages.

4.2. Constructing RELAYFORWARD messages

4.2.1. Initializing the fixed-length header

The relay agent constructs the RELAYFORWARD message by constructing the fixed-length header as specified in the earlier section titled 'Encoding'. The relay agent MUST set the 'op' field to a value of RELAYFORWARD.

If the relay agent is not a layer two relay agent [I-D.ietf-dhc-l2ra], it MUST store one of its own IP addresses in the 'aiaddr' field. This address MUST be a valid IP address that is reachable by the next hop relay(s) or DHCP server(s) to which the relay agent is configured to forward.

DHCP servers normally use the relay agent IP address to determine on what link the DHCP client's IP address should be allocated. In some cases, the value stored in the 'aiaddr' field will not be a valid IP address on the link on which the source message was received. In this case, the relay agent MUST include a link selection suboption [RFC3527] that identifies that link in the relay segment.

If the relay agent is a layer two relay agent, it MAY include a link selection suboption in the relay segment.

If the message being encapsulated is a BOOTREQUEST, L2RAs MUST store a value of zero in the 'aiaddr' field. Otherwise, the L2RA MUST copy the value of the 'aiaddr' field in the RELAYFORWARD message being encapsulated into the 'aiaddr' field of the RELAYFORWARD message that it generates.

The 'rslen' field depends on the length of the relay segment. The 'caplen', 'padlen' and 'ep' values in the fixed header are initialized differently depending on whether the message being encapsulated is a BOOTREQUEST or a RELAYFORWARD message.

4.2.2. Initializing the relay segment

Following the fixed header, the relay agent MUST append any RAIO suboptions it wishes to send to the DHCP server; this is the 'relay segment'. It MUST store the length of the relay segment in the 'rslen' field of the fixed header.

The relay agent SHOULD include a Relay Agent ID suboption [I-D.ietf-dhc-relay-id-suboption] in the relay segment to identify itself to the DHCP server.

4.2.3. Fixed header settings for RELAYFORWARD messages

If the message being encapsulated is a RELAYFORWARD message, the relay agent MUST initialize the 'caplen' field of the fixed header to the length of the source message, excluding any layer 2, IP and UDP headers. It MUST append the contents of the message, again excluding any layer 2, IP or UDP headers, to the new message. It MUST initialize the 'ep' and 'padlen' fields in the fixed header of the new message to zero.

4.2.4. Fixed header settings for BOOTREQUEST messages

If the message being encapsulated is a BOOTREQUEST message, the relay agent determines the length of the encapsulation segment by scanning forward across the option buffer of the source message, beginning with the first option in the option buffer, until an End option is

reached, or the end of the buffer is reached. The difference between the offset of this location in the message and the offset of the first location following the UDP header of the message is the length of the message to be relayed.

If an End option terminated the scan, the relay agent MUST set the value of the 'ep' field in the fixed header to one. Otherwise, the relay agent MUST set the value of the 'ep' field to zero.

The relay agent MUST count all of the Pad options that follow the last option in the option buffer that is neither a Pad nor an End option. The relay agent MUST store this count in the 'padlen' field of the fixed header.

The relay agent MUST store the difference between the value stored in 'padlen' and the length of the message to be relayed in the 'caplen' field of the fixed header.

4.2.5. Initializing the encapsulation segment

The relay agent MUST copy the portion of the message being encapsulated that immediately follows the UDP header into the RELAYFORWARD message being generated. The length of the data being copied is the value that was stored in 'caplen'.

4.3. Decapsulating RELAYREPLY messages

To decapsulate a RELAYREPLY message, the relay agent creates a decapsulated message, processes any RAIO suboptions it recognizes in the relay segment, and forwards the decapsulated message to its destination.

4.3.1. Processing relay agent suboptions

The suboptions parsed from the relay segment are processed by the relay agent as specified in the Relay Agent Information Option specification [RFC3046], as well as in any documents that define suboptions to the Relay Agent Information Option. A current list of DHCP Relay Agent suboptions and the documents that define them can be located in the IANA protocol registry for Bootp and DHCP parameters, in the section titled "DHCP Relay Agent Sub-Option Codes."

4.3.2. Constructing the decapsulated message

To construct a decapsulated message, the relay agent copies the portion of the RELAYREPLY message following the relay segment, with a length specified in the 'caplen' field of the fixed-length header, into the new message.

4.4. Retransmitting modified messages

If the relay agent did not modify the message either by encapsulating or decapsulating it, it retransmits the message according to existing practice.

Otherwise, how the modified message is transmitted to its next destination depends on two factors. First, is the relay agent that modified the message a layer two [I-D.ietf-dhc-l2ra] relay agent or a layer three [RFC1542] relay agent? Second, is the modified message a BOOTREPLY message, a RELAYREPLY message, or some other message?

4.4.1. Layer two relay agents

There are two special aspects to the handling of relayed packets in Layer Two Relay Agents (L2RAs). The first is the construction of the layer two, IP and UDP headers on the packet. The second is how the L2RA makes the forwarding decision.

4.4.1.1. Constructing the headers

The L2RA constructs the headers on the relayed packet by copying and, as necessary, modifying the headers from the original packet.

If there is a layer two header, the L2RA MUST copy this header in accordance with the needs of the particular layer two implementation it is using. If the header contains a packet length field, the L2RA MUST adjust the value in the packet length field. If the header contains a non-secure integrity check such as a CRC or checksum that covers the entire packet, the L2RA MUST recompute this value.

L2RA encapsulation in cases where the layer two contains a secure integrity check must either construct a new integrity signature, or else remove the integrity signature. If neither of these is possible, the L2RA MUST silently discard the packet.

The L2RA MUST copy the IP header without modification. If the IP header contains any sort of secure integrity check on the packet, the L2RA MUST silently discard the packet.

The L2RA MUST copy the UDP header and adjust the ’Length’ field [RFC0768]. If the contents of the ’Checksum’ field are not zero, the L2RA MUST compute a new checksum according to the algorithm specified in User Datagram Protocol. [RFC0768]

4.4.1.2. Forwarding the modified packet

Ordinarily when a layer two device forwards a packet, it simply copies that packet from the interface on which it was received and transmits it, unchanged, on one or more interfaces other than that interface. The mechanism used to choose which interfaces it forwards the packet to is beyond the scope of this document.

Once a DHCP packet has been modified by the L2RA either as an encapsulation or a decapsulation, the L2RA must forward it either toward the DHCP server or the DHCP client. The implementation has two options: transmit the packet as it would transmit any other packet, or use its configuration and/or information in the relay header to direct the packet out a particular interface.

The details of ordinary packet forwarding in devices that implement L2RA is beyond the scope of this document.

When processing a RELAYREPLY message, the L2RA MAY use information in the relay segment of the RELAYREPLY to determine on which network interface the RELAYREPLY should be forwarded.

When processing any other message, the L2RA MAY use configuration information to direct the packet out a specific port or ports that have been marked as reaching DHCP servers. The L2RA MUST NOT forward any packet on the interface on which it was received, even if that interface is so marked.

4.5. Layer Three Relay Agents

4.5.1. Transmitting a decapsulated RELAYREPLY message

When the decapsulated message is itself a RELAYREPLY message, the relay agent MUST forward the decapsulated message to the IP address specified in the ’aiaddr’ field of the fixed-length header.

If the relay segment of the packet that was decapsulated contains a Link Layer Address suboption, the relay agent MUST transmit the packet to that link layer address without attempting to use Address Resolution Protocol (ARP) to translate the address contained in ’aiaddr’ to a layer two address.

4.5.2. Transmitting a decapsulated BOOTREPLY message

When transmitting a decapsulated BOOTREPLY message, the relay agent transmits the message as specified in Bootstrap Protocol, Section 4 [RFC0951].

4.5.3. Transmitting other messages

When transmitting RELAYFORWARD and BOOTREQUEST messages, the relay agent simply sends the message to the IP address or addresses configured as DHCP servers for that relay agent.

5. DHCP Server Behavior

A DHCP server which receives a RELAYREPLY message MUST silently discard that message.

5.1. Receiving RELAYFORWARD messages

DHCP servers that implement this specification MUST decapsulate RELAYFORWARD messages.

5.1.1. Decapsulation

By design, this specification supports multiple layers of encapsulation. The DHCP server implementation therefore MUST decapsulate each layer and retain the information in that layer, organized so that the ordering of the encapsulation is available both during packet processing, and when constructing the reply.

Aside from the necessity of handling an RAIO differently than an encapsulation when constructing a RELAYREPLY message, DHCP servers MUST NOT, by default, treat relay suboptions received in an RAIO any differently than relay suboptions received in an encapsulation.

It is not the goal of this specification to define a particular implementation strategy or data structure for representing the encapsulation, so long as the ability to process the options and suboptions as required below is preserved. Implementations MAY provide means for addressing the contents of relay segments and of the inner encapsulation segment in any convenient form, as long as it conforms generally to the requirements of this document.

5.1.2. Processing of decapsulated suboptions

This section specifies requirements for the processing of decapsulated relay suboptions in the default case, where no custom processing has been specified. This should not be construed to forbid implementations for providing mechanisms for customizing the processing of these suboptions.

This document does not specify special handling for DHCP options. Only the inner encapsulation--the encapsulation of the original

message sent from the client, which is done by the first encapsulating relay--can ever contain DHCP Options; hence, whatever normal mechanisms a DHCP server provides for processing these options should suffice.

Some relay agent suboptions, such as the Relay Agent Subnet Selection suboption [RFC3527], are intended to be processed by DHCP servers. Others, like the Circuit ID and Remote ID [RFC3046] suboptions, were not intended to be processed by the DHCP server, but are often used by the DHCP server for identification purposes.

In cases where more than one relay agent sends the same suboption, the DHCP server must either factor all such suboptions into its processing, or choose one such suboption and use that exclusively for processing.

By default, DHCP servers MUST use the outermost suboption (the one added by the relay agent closest to the DHCP server) for every suboption that was sent by more than one relay agent.

5.1.3. Address allocation

During normal processing, DHCP servers use a variety of data to determine where the DHCP client is in the network topology. These data are provided by relay agents. In the absence of any relay-agent-provided topology data, the DHCP server assumes that the client is connected to the link on which the message was received.

One datum used by DHCP servers in locating the client is the value of the 'giaddr' field of the BOOTP header. This specification eliminates the use of giaddr; hence, it cannot be used in the address allocation decision.

The functionality provided by giaddr is replaced in this specification by the 'aiaddr' field in the fixed-length relay header.

5.1.3.1. Default link selection algorithm

DHCP servers MUST implement a default algorithm for determining the link to which the client is attached. This algorithm is to first search the client message for a subnet selection option [RFC3011].

The server next searches for link-identifying data in each RELAYFORWARD encapsulation, starting from the inner most and ending at the outermost, until a RELAYFORWARD is found that identifies the client's location.

A RELAYFORWARD encapsulation contains link-identifying data if the

value of the 'aiaddr' field of the fixed-length header is not zero, or if the relay segment contains a Link Selection suboption [RFC3527].

If a Link Selection suboption is present in the innermost RELAYFORWARD message containing link-identifying data, the DHCP server MUST use the contents of that option to identify the link to which the client is connected.

Otherwise, if a Subnet Selection option was found in the client message, the DHCP server MUST use the contents of that option to identify the link to which the client is connected.

Otherwise, the DHCP server MUST use the contents of the 'aiaddr' field in the RELAYFORWARD encapsulation that was identified as being the innermost RELAYFORWARD containing link-identifying data.

5.1.3.2. Other link selection algorithms

DHCP servers implementing this specification MAY implement link selection algorithms other than the one described in the preceding section. DHCP servers MUST NOT use any link selection algorithm other than the one described in the preceding section unless specially configured to do so.

5.2. Responding to RELAYFORWARD messages

Once the DHCP server has processed the encapsulated message from the DHCP client and constructed a response to the DHCP client, it constructs a RELAYREPLY message and sends it to the next hop on the way to the client.

5.2.1. Constructing a RELAYREPLY encapsulation

The server MUST encapsulate any response to a client message contained in one or more RELAYFORWARD encapsulations in a set of corresponding RELAYREPLY encapsulations. Each RELAYREPLY is nested in the same way that the corresponding RELAYFORWARD was nested, so that the innermost RELAYREPLY corresponds to the innermost RELAYFORWARD, and the outermost RELAYREPLY corresponds to the outermost RELAYFORWARD.

5.2.1.1. Constructing the relay segments

The server MUST copy every suboption that appeared in the relay segment of the RELAYFORWARD encapsulation into corresponding outgoing RELAYREPLY encapsulation's relay segment. The server SHOULD NOT preserve the order of options from the incoming relay segment to the

outgoing relay segment.

If the message encapsulated by a particular RELAYREPLY encapsulation is not a RELAYREPLY, or if the message encapsulated by that RELAYREPLY is a RELAYREPLY, but the 'aiaddr' field of the fixed-length header of the inner RELAYREPLY has a value of zero, the DHCP server MUST include a Layer Two Address suboption in the relay segment. The DHCP server MUST set the 'htype' field of the Layer Two Address suboption to the value of 'htype' in the client message. The DHCP server MUST set the 'chaddr' field in the Layer Two Address suboption to the value of the 'chaddr' field in the client message.

5.2.1.2. Constructing the fixed-length header

The server MUST set the values of 'ep' and 'padlen' in the fixed-length header to zero. The server MUST set the value of 'caplen' to the length of the message encapsulated in the RELAYREPLY encapsulation being constructed. The server MUST set the value of 'rslen' to the length of the relay segment of the RELAYREPLY encapsulation being constructed.

If the message encapsulated in the RELAYREPLY being constructed is a RELAYREPLY, and the 'aiaddr' field of the RELAYFORWARD encapsulation corresponding to the inner RELAYREPLY is not zero, the DHCP server MUST copy the value from that 'aiaddr' field to the 'aiaddr' field of the RELAYREPLY being constructed.

Otherwise, if the BROADCAST bit in the 'flags' field of the inner message is set to 1, or 'ciaddr' is zero and 'yiaddr' is also zero, the DHCP server MUST set the value of 'aiaddr' to 255.255.255.255. If 'ciaddr' is not zero, the DHCP server must copy that value to 'aiaddr'. If 'ciaddr' is zero and 'yiaddr' is not, the DHCP server MUST copy the value of 'yiaddr' to 'aiaddr'.

5.2.2. Transmission of RELAYREPLY messages

If the value of 'aiaddr' in the outermost RELAYFORWARD encapsulation to which the RELAYREPLY is a response is nonzero, the DHCP server MUST transmit the RELAYREPLY to that IP address.

Otherwise, if 'ciaddr' and 'yiaddr' are both zero, or the BROADCAST bit in the 'flags' field is set to 1, or the DHCP server implementation is not able to perform the ARP-cache preloading trick described in Bootstrap Protocol, Section 4 [RFC0951], the DHCP server MUST broadcast the RELAYREPLY message to the 255.255.255.255 broadcast address.

Otherwise, the DHCP server MUST use the value of 'ciaddr', if

nonzero, or 'yiaddr', otherwise, as the destination address for the message, and MUST preload its ARP cache (or otherwise arrange to transmit the message without using ARP) to the layer two address provided by the client in 'htype' and 'chaddr' and 'hlen'.

5.3. Responding to messages other than RELAYFORWARD

When a DHCP server constructs a response to a DHCP client message that did not arrive encapsulated in a RELAYFORWARD message, the DHCP server MUST NOT encapsulate the response in a RELAYREPLY message. DHCP server implementors should be careful that their servers do not respond to an incoming RAIO from a non-conforming relay agent with a RELAYREPLY message.

6. DHCP Client Behavior

A DHCP client that receives either a RELAYFORWARD message or a RELAYREPLY message MUST silently discard that message.

7. Security Considerations

DHCP Relay Information Option [RFC3046] limits relay agent information to a single relay agent, and provides some minimal anti-spoofing. By supporting relay agent chaining, it is now possible for a relay agent downstream of the trusted portion of a provider network to communicate relay agent information options to a DHCP server.

This specification defines a much more robust spoofing rejection mechanism, by allowing the administrator to configure the relay to discard RELAYFORWARD messages originating from outside of the trusted portion of the network. Administrators of networks that rely on this trusted/untrusted configuration are encouraged to configure edge relays to discard RELAYFORWARD messages.

In networks where trusted relay agents may be separated from the DHCP server by transit networks that are not trusted, it is possible to modify the message in transit. This possibility exists with normal DHCP relays as well. Administrators of such networks should consider using relay agent authentication [RFC4030] to prevent modification of relay agent communications in transit.

8. IANA Considerations

We request that IANA assign one new suboption code from the registry of DHCP Agent Sub-Option Codes maintained in

http://www.iana.org/assignments/bootp-dhcp-parameters. This suboption code will be assigned to the Layer Two Address Suboption.

9. References

9.1. Normative References

[I-D.ietf-dhc-relay-id-suboption]
Stapp, M., "The DHCPv4 Relay Agent Identifier Suboption", draft-ietf-dhc-relay-id-suboption-07 (work in progress), July 2009.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC0951] Croft, B. and J. Gilmore, "Bootstrap Protocol", RFC 951, September 1985.

[RFC1542] Wimer, W., "Clarifications and Extensions for the Bootstrap Protocol", RFC 1542, October 1993.

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC3011] Waters, G., "The IPv4 Subnet Selection Option for DHCP", RFC 3011, November 2000.

[RFC3046] Patrick, M., "DHCP Relay Agent Information Option", RFC 3046, January 2001.

[RFC3118] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.

[RFC3527] Kinnear, K., Stapp, M., Johnson, R., and J. Kumarasamy, "Link Selection sub-option for the Relay Agent Information Option for DHCPv4", RFC 3527, April 2003.

[RFC4030] Stapp, M. and T. Lemon, "The Authentication Suboption for the Dynamic Host Configuration Protocol (DHCP) Relay Agent Option", RFC 4030, March 2005.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

9.2. Informative References

[I-D.ietf-dhc-l2ra]
Joshi, B. and P. Kurapati, "Layer 2 Relay Agent Information", draft-ietf-dhc-l2ra-04 (work in progress), April 2009.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

Authors' Addresses

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

Phone: +1 650 381 6000
Email: mellon@nominum.com

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

Email: denghui@chinamobile.com

Dynamic Host Configuration (DHC)
Internet-Draft
Updates: 3633 (if approved)
Intended status: Standards Track
Expires: April 11, 2011

J. Korhonen, Ed.
Nokia Siemens Networks
T. Savolainen
Nokia
S. Krishnan
Ericsson
O. Troan
Cisco Systems, Inc

October 8, 2010

Prefix Exclude Option for DHCPv6-based Prefix Delegation
draft-ietf-dhc-pd-exclude-00.txt

Abstract

This specification defines an optional mechanism to allow exclusion of one specific prefix from a delegated prefix set when using DHCPv6-based prefix delegation.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 11, 2011.

Copyright Notice

Table of Contents

1. Introduction

This specification defines an optional mechanism and the related DHCPv6 option to allow exclusion of one specific prefix from a delegated prefix set when using DHCPv6-based prefix delegation.

The prefix exclusion mechanism is targeted to deployments where DHCPv6-based prefix delegation is used but a single aggregatable route/prefix has to represents one customer, instead of using one prefix for the link between the delegating router and the requesting router and another prefix for the customer network. The mechanism defined in this specification allows a delegating router to use a prefix out of the delegated prefix set on the link through which it exchanges DHCPv6 messages with the requesting router.

2. Requirements and Terminology

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

3. Prefix Delegation with Excluded Prefixes

3.1. Problem Background

DHCPv6 Prefix Delegation (DHCPv6-PD) [RFC3633] has an explicit limitation described in Section 12.1 of [RFC3633] that a prefix delegated to a requesting router cannot be used by the delegating router. This restriction implies that the delegating router will have two (non aggregatable) routes towards a customer, one for the link between the requesting router and the delegating router and one for the customer site behind the requesting router. This approach works well with the unnumbered router model (i.e. routers on the link have no globally scoped prefixes). Also the same approach applies to the case where the prefix assigned to the requesting router link through which it received DHCP messages does not in any way need to be associated to the delegated prefixes.

There are architectures and link models, where a host (e.g. a mobile router, also acting as a requesting router) always has a single (/64) prefix configured on its uplink interface and the delegating router is also requesting router’s first hop router. Furthermore, it may be required that the prefix configured on the uplink interface has to be aggregatable with the delegated prefixes. This introduces a problem in how to use DHCPv6-PD together with stateless [RFC4862] or stateful [RFC3315] address autoconfiguration on a link, where the /64

advertised on the link is also part of the prefix delegated (e.g /56) to the requesting router.

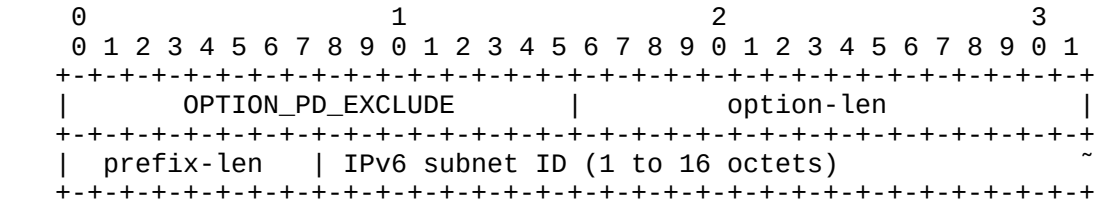
3.2. Proposed Solution

This specification defines a new DHCPv6 option, OPTION_PD_EXCLUDE (TBD1), that is used to exclude exactly one prefix from a delegated prefix. The OPTION_PD_EXCLUDE MUST only be included in the OPTION_IAPREFIX IAprefix-options field. There can be at most one OPTION_PD_EXCLUDE option in one OPTION_IAPREFIX option. The OPTION_PD_EXCLUDE option allows prefix delegation where a requesting router is delegated a prefix (e.g. /56) and the delegating router uses one prefix (e.g. /64) on the link through which it exchanges DHCPv6 messages with the requesting router with a prefix out of the same delegated prefix set.

A requesting router SHOULD include an OPTION_ORO option with the OPTION_PD_EXCLUDE option code in a Solicit, Request, Renew, Rebind or Confirm message to inform the delegating router about the support for the prefix delegation functionality defined in this specification. A delegating router MAY include the OPTION_PD_EXCLUDE option code in an OPTION_ORO option in a Reconfigure message for indicating that the requesting router should request OPTION_PD_EXCLUDE from the delegating router.

The delegating router includes the prefix in the OPTION_PD_EXCLUDE option that is excluded from the delegated prefix set. The requesting router MUST NOT assign the excluded prefix to any of its downstream interfaces.

4. Prefix Exclude Option



Prefix Exclude Option

- option-code: OPTION_PD_EXCLUDE (TBD1).
- option-len: 1 + length of IPv6 subnet ID in octets. A valid option-len is between 2 and 17.

- o prefix-len: The length of the excluded prefix in bits. The prefix-len MUST be between 'OPTION_IAPREFIX prefix-length'+1 and 128.

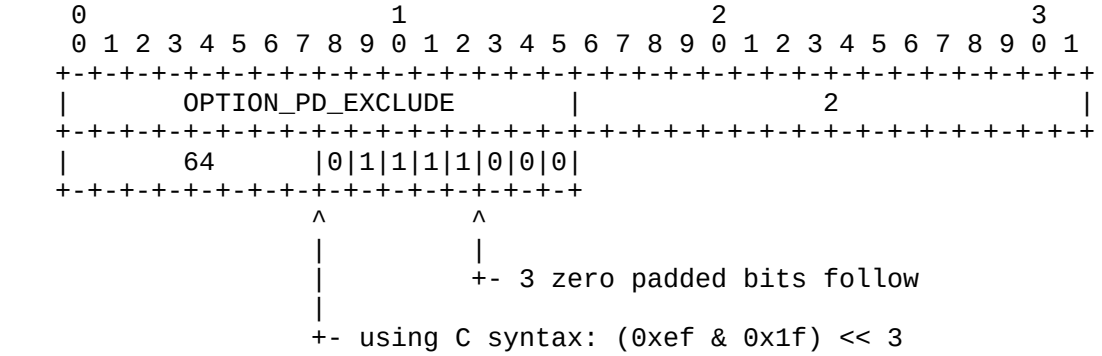
- o IPv6 subnet ID: A variable length IPv6 subnet ID up to 128 bits. The subnet ID contains prefix-len minus 'OPTION_IAPREFIX prefix-length' bits extracted from the excluded prefix starting from the bit position 'OPTION_IAPREFIX prefix-length'. The extracted subnet ID MUST be left shifted to start from a full octet boundary, i.e. left shift of 'OPTION_IAPREFIX prefix-length' mod 7 bits. The subnet ID MUST be zero padded to the next full octet boundary.

The OPTION_PD_EXCLUDE option MUST only be included in the OPTION_IAPREFIX IAprefix-options [RFC3633] field. The OPTION_PD_EXCLUDE option MUST be located before the possible Status Code option in the IAprefix-options field.

Any prefix excluded from the delegated prefix MUST be contained in OPTION_PD_EXCLUDE options within the corresponding OPTION_IAPREFIX.

The prefix included in the OPTION_PD_EXCLUDE option share the same preferred-lifetime and valid-lifetime as the delegated prefix in the encapsulating OPTION_IAPREFIX option.

The prefix in the OPTION_PD_EXCLUDE option MUST be part of the delegated prefix in the OPTION_IAPREFIX. For example, the requesting router has earlier been assigned a 2001:db8:dead:beef::/64 prefix by the delegating router, and the delegated prefix in the OPTION_IAPREFIX is 2001:db8:dead:bee0::/59. In this case, 2001:db8:dead:beef::/64 is a valid prefix to be used in the OPTION_PD_EXCLUDE option. The OPTION_PD_EXCLUDE option would be encoded as follows:



5. Delegating Router Solicitation

The requesting router locates and selects a delegating router in the same way as described in Section 11 [RFC3633]. This specification only describes the additional steps required by the use of OPTION_PD_EXCLUDE option.

5.1. Requesting Router

If the requesting router implement the solution described in Section 3.2 then the requesting router MUST include the OPTION_PD_EXCLUDE option code in the OPTION_ORO option in the Solicit message.

Once receiving Advertise message, the requesting router uses the prefix(es) received in OPTION_PD_EXCLUDE in addition to the advertised prefixes to choose the delegating router to respond to. If Advertise message did not include OPTION_PD_EXCLUDE option, then the requesting router MUST fall back to normal [RFC3633] behavior.

> Editor's Note: is there actually deployment case when multiple delegating routers would respond?

5.2. Delegating Router

If the OPTION_ORO option in the Solicit message includes the OPTION_PD_EXCLUDE option code, then the delegating router knows that the requesting router supports the solution defined in this specification. If the Solicit message also contains an IA_PD option, the delegating router can delegate to the requesting router a prefix which includes the prefix already assigned to the requesting router's uplink interface. The delegating router includes the prefix originally or to be assigned to the requesting router in the OPTION_PD_EXCLUDE option within the OPTION_IAPREFIX IAprefix-option in the Advertise message.

If the OPTION_ORO option in the Solicit message does not include the OPTION_PD_EXCLUDE option code, then the delegating router MUST fall back to normal [RFC3633] behavior.

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

6. Requesting Router Initiated Prefix Delegation

The procedures described in the following sections are aligned with Section 12 of [RFC3633]. In this specification we only describe the additional steps required by the use of OPTION_PD_EXCLUDE option.

6.1. Requesting Router

The requesting router behavior regarding the use of the OPTION_PD_EXCLUDE option is more or less identical to step described in Section 5.1. The only difference really is different used DHCPv6 messages.

The requesting router uses a Release message to return the delegated prefix(es) to a delegating router. The prefix(es) to be released MUST be included in the IA_PDs along with the excluded prefix included in the OPTION_PD_EXCLUDE option. The requesting router MUST NOT use the OPTION_PD_EXCLUDE option to introduce additional excluded prefix in the Release message that it originally got a valid binding for.

The requesting router must create sink routes for the delegated prefixes minus the excluded prefixes. This may be done by creating sink routes for delegated prefixes and more specific routes for the excluded prefixes.

6.2. Delegating Router

The delegating router behavior regarding the use of the OPTION_PD_EXCLUDE option is more or less identical to step described in Section 5.2. The only difference really is DHCPv6 messages used to carry the OPTION_PD_EXCLUDE option.

The delegating router may mark any prefix(es) in IA_PD Prefix options in a Release message from a requesting router as 'available' excluding the prefix included in the OPTION_PD_EXCLUDE options. If the Release message contains 'new' excluded prefix in any OPTION_PD_EXCLUDE option, the delegating router MUST send a Reply message with Status Code set to NoBinding for that IA_PD option.

7. Security Considerations

Security considerations in DHCPv6 are described in Section 23 of [RFC3315], and for DHCPv6 Prefix Delegation in Section 12 of [RFC3633].

8. IANA Considerations

A new DHCPv6 Option Code is reserved from DHCPv6 registry for DHCP Option Codes.

OPTION_PD_EXCLUDE is set to TBD1

9. Acknowledgements

Authors would like to thank Ralph Droms, Frank Brockners, Ted Lemon, Julien Laganier, Fredrik Garneij, Sri Gundavelli, Mikael Abrahamsson, Behcet Sarikaya, Jyrki Soini and Deng Hui for their valuable comments and discussions.

10. References

10.1. Normative References

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

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

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

10.2. Informative References

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

Authors' Addresses

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

Email: jouni.nospam@gmail.com

Teemu Savolainen
Nokia
Hermiankatu 12 D
FI-33720 Tampere
Finland

Email: teemu.savolainen@nokia.com

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

Email: suresh.krishnan@ericsson.com

Ole Troan
Cisco Systems, Inc
Veversmauet 8
N-5017 BERGEN
Norway

Email: ot@cisco.com

DHC Working Group
Internet Draft
Intended Status: Standards Track
Expires: April 22, 2011

Kim Kinnear
Richard Johnson
Mark Stapp
Cisco Systems
Jay Kumarasamy
October 22, 2010

Virtual Subnet Selection Options for DHCPv4 and DHCPv6
<draft-ietf-dhc-vpn-option-12.txt>

Status of this Memo

Copyright Notice

Abstract

This memo defines a Virtual Subnet Selection (VSS) option for each of DHCPv4 and DHCPv6, and a VSS sub-option carried in the DHCPv4 relay-agent-information option. These are intended for use by DHCP clients, relay agents, and proxy clients in situations where VSS information needs to be passed to the DHCP server for proper address or prefix allocation to take place.

For the DHCPv4 option and relay-agent-information sub-option, this memo documents existing usage as per RFC 3942 [RFC3942].

Table of Contents

1. Introduction

There is a growing use of Virtual Private Network (VPN) configurations. The growth comes from many areas; individual client systems needing to appear to be on the home corporate network even when traveling, ISPs providing extranet connectivity for customer companies, etc. In some of these cases there is a need for the DHCP server to know the VPN (hereafter called a "Virtual Subnet Selector" or "VSS") from which an address, and other resources, should be allocated.

This memo defines a Virtual Subnet Selection (VSS) option for each of DHCPv4 and DHCPv6, and a VSS sub-option carried in the DHCPv4 relay-agent-information option. These are intended for use by DHCP clients, relay agents, and proxy clients in situations where VSS information needs to be passed to the DHCP server for proper address or prefix allocation to take place. If the receiving DHCP server understands the VSS option or sub-option, this information may be used in conjunction with other information in determining the subnet on which to select an address as well as other information such as DNS server, default router, etc.

If the allocation is being done through a DHCPv4 relay, then the relay sub-option defined here should be included. In some cases, however, an IP address is being sought by a DHCPv4 proxy on behalf of a client (which may be assigned the address via a different protocol). In this case, there is a need to include VSS information relating to the client as a DHCPv4 option.

If the allocation is being done through a DHCPv6 relay, then the DHCPv6 VSS option defined in this document should be included in the Relay-forward and Relay-reply message going between the DHCPv6 relay and server. In some cases, addresses or prefixes are being sought by a DHCPv6 proxy on behalf of a client. In this case, there is a need for the client itself to supply the VSS information using the DHCPv6 VSS option in the messages that it sends to the DHCPv6 server.

In the remaining text of this document, when a DHCPv6 address is indicated the same information applies to DHCPv6 Prefix Delegation [RFC3633] as well.

In the remaining text of this document, when the term VSS sub-option is used, it refers to the VSS sub-option carried in the DHCPv4 relay-agent-information option.

2. Terminology

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

This document uses the following terms:

- "DHCP client"

  A DHCP client is a host using DHCP to obtain configuration parameters such as a network address.

- "DHCP proxy"

  A DHCP proxy is a DHCP client which acquires IP addresses not for its own use, but rather on behalf of another entity. There are a variety of ways that a DHCP proxy can supply the addresses it acquires to other entities that need them.

- "DHCP relay agent"

  A DHCP relay agent is an agent that transfers BOOTP and DHCP messages between clients and servers residing on different subnets, per [RFC951], [RFC1542], and [RFC3315].

- "DHCP server"

  A DHCP server is a host that returns configuration parameters to DHCP clients.

- "DHCPv4 option"

  An option used to implement a capability defined by the DHCPv4 RFCs [RFC2131][RFC2132]. These options have one-octet code and size fields.

- "DHCPv4 sub-option"

  As used in this document, a DHCPv4 sub-option refers to a sub-option of the relay-agent-information option [RFC3046]. These sub-options have one-octet code and size fields.

- "DHCPv6 option"

  An option used to implement a capability defined by the DHCPv6 RFC [RFC3315]. These options have two-octet code and size fields.

- "Global VPN"

  Indicates that the address being described belongs to the set of addresses not part of any VPN. In other words, the normal address space operated on by DHCP. This includes private addresses, for example the 10.x.x.x addresses as well as the other private subnets that are not routed on the open internet.

- "VSS information"

  Information about a VPN necessary to allocate an address to a DHCP client on that VPN and necessary to forward a DHCP reply packet to a DHCP client on that VPN.

- "VPN"

  Virtual private network. A network which appears to the client to be a private network.

- "VPN Identifier"

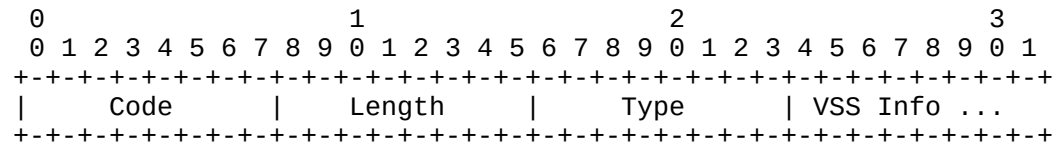
  The VPN-ID is defined by [RFC2685] to be a sequence of 7 octets.

3. Virtual Subnet Selection Option and Sub-Options Definitions

The Virtual Subnet Selection options and sub-options contain a generalized way to specify the VSS information about a VPN. There are two options and one sub-option defined in this section. The actual VSS information is identical in each.

3.1. DHCPv4 Virtual Subnet Selection Option

The format of the option is:



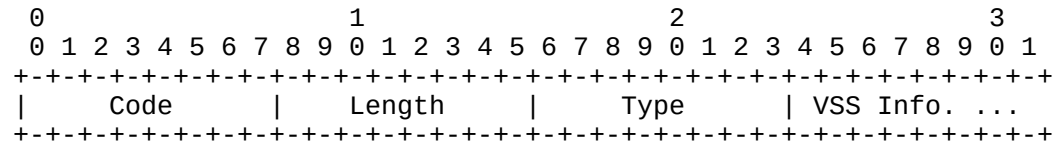
Code The option code (221).

Length The option length, minimum 1 octets.

Type and VSS Information -- see Section 3.4

3.2. DHCPv4 Virtual Subnet Selection Sub-Option

This is a sub-option of the relay-agent-information option [RFC3046]. The format of the sub-option is:



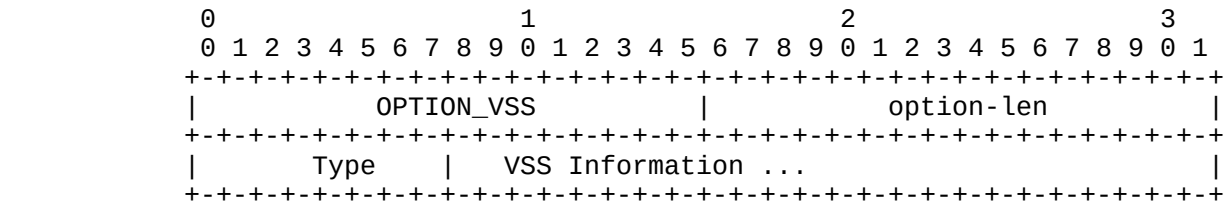
Code The sub-option code (151).

Length The option length, minimum 1 octets.

Type and VSS Information -- see Section 3.4

3.3. DHCPv6 Virtual Subnet Selection Option

The format of the DHCPv6 Virtual Subnet Selection option is shown below. This option may be included by a client or relay-agent (or both).



option-code OPTION_VSS (TBD).

option-len The number of octets in the option, minimum 1.

Type and VSS Information -- see Section 3.4

3.4. Virtual Subnet Selection Type and Information

All of the (sub)options defined above carry identical payloads, consisting of a type and additional VSS information as follows:

Type	VSS Information format:
0	NVT ASCII VPN identifier
1	RFC2685 VPN-ID
2-252	Reserved
253	CONTROL (DHCPv4 VSS sub-option only)
254	Reserved
255	Global, default VPN.

- Type 0 -- NVT ASCII VPN identifier

  Indicates that the VSS information consists of a NVT ASCII string. It MUST NOT be terminated with a zero byte.

- Type 1 -- RFC2685 VPN-ID

  Indicates that the VSS information consists of an RFC2685 VPN-ID [RFC2685], which is defined to be 7 octets in length.

- Type 253 -- CONTROL

  This is only valid for the DHCPv4 relay-agent-information option

sub-option. It indicates that another DHCPV4 VSS sub-option is present in the relay-agent-information option. The sub-option with type CONTROL MUST be removed by any DHCPv4 server which successfully processes the information in the other DHCPv4 sub-option with valid VSS information. In this case, there MUST NOT be any VSS Information included in the sub-option, and the length of the VSS sub-option MUST be 1.

- o Type 255 -- Global, default VPN

  Indicates that there is no explicit, non-default VSS information but rather that this option references the normal, global, default address space. In this case, there MUST NOT be any VSS Information included in the VSS option or sub-option and the length of the MUST be 1.

All other values of the Type field are reserved.

4. Overview of Virtual Subnet Selection Usage

At the highest level, the VSS option or sub-option determines the VPN on which a DHCP client is supposed to receive an IP address. How the option or sub-option is entered and processed is discussed below, but the point of all of the discussion is to determine the VPN on which the DHCP client resides. This will affect a relay agent, in that it will have to ensure that DHCP packets sent to and received from the DHCP client flow over the correct VPN. This will affect the DHCP server in that it determines the IP address space used for the IP address allocation.

A DHCP server has as part of its configuration some IP address space from which it allocates IP addresses to DHCP clients. These allocations are typically for a limited time, and thus the DHCP client gets a lease on the IP address. In the absence of any VPN information, the IP address space is in the global or default VPN used throughout the Internet. When a DHCP server deals with VPN information, each VPN defines a new address space inside the server, one distinct from the global or default IP address space. A server which supports the VSS option or sub-option thereby supports allocation of IP addresses from multiple different VPNs. Supporting IP address allocation from multiple different VPNs means that the DHCP server must be prepared to configure multiple different address spaces (one per distinct VPN) and allocate IP addresses from these different address spaces.

These address spaces are typically independent, so that the same IP address (consisting of the same string of bytes) could be allocated to one client in the global, default VPN, and to a different client

residing in a different VPN. There is no conflict in this allocation, since the clients have essentially different addresses, even though these addresses consist of the same string of bytes, because the IPv4 or IPv6 address is qualified by the VPN.

Thus a VSS option or sub-option is a way of signaling the use of a VPN other than the global or default VPN. The next question is: who decides what VPN a DHCP client should be using?

There are three entities which can either insert a VSS option or sub-option into a DHCPv4 packet or DHCPv6 message; a DHCP client, a relay agent, or a DHCPv4 or DHCPv6 server. While all of these entities could include a different VSS option or sub-option in every request or response, this situation is neither typical nor useful. There are two known paradigms for use of the VSS option or sub-option, which are discussed below.

4.1. VPN assignment by the DHCP relay agent

The typical use of the VSS option or sub-option is for the relay agent to know the VPN on which the DHCP client is operating. The DHCP client itself does not, in this approach, know the VPN on which it resides. The relay agent is responsible for mediating the access between the VPN on which the DHCP client resides and the DHCP server. In this situation, the relay agent will insert two DHCPv4 VSS sub-options (one with valid VSS information, and one with type CONTROL) into the relay-agent-information option or a DHCPv6 VSS option into the Relay-forward message of every request it forwards from the DHCP client. The server will use the VSS option or sub-option to determine the VPN on which the client resides, and use that VPN information to select the address space within its configuration from which to allocate an IP address to the DHCP client.

When, using this approach, a DHCPv4 relay agent inserts a VSS sub-option with containing VSS information it MUST also insert a VSS sub-option containing type CONTROL, no additional VSS information, and a length of 1. This is to allow determination of whether or not the DHCPv4 server actually processes the VSS information provided by the DHCPv4 relay agent. If the DHCPv4 server supports the VSS capabilities described in this document, it will remove the VSS sub-option with type CONTROL from the relay-agent-information option that it returns to the DHCPv4 relay agent. See Section 5 for more information.

In this approach, the relay agent might also send a VSS option or sub-option in either a DHCPv4 or DHCPv6 Leasequery request, but in this case, it would use the VSS option in the Leasequery request to select the correct address space for the Leasequery. In this

approach, the relay agent would be acting as a DHCP client from a Leasequery standpoint, but it would not be as if a DHCP client were sending in a VSS option in a standard DHCP address allocation request, say a DHCPDISCOVER.

In this approach, only one relay agent would mediate the VPN access for the DHCP client to the DHCP server, and it would be the relay agent which inserts the VSS information into the request packet and would remove it prior to forwarding the response packet on.

In the diagram below is an example of a DHCPv4 client, DHCPv4 relay agent, and DHCPv4 server. The DHCPv6 situation is similar, but uses the DHCPv6 VSS option.

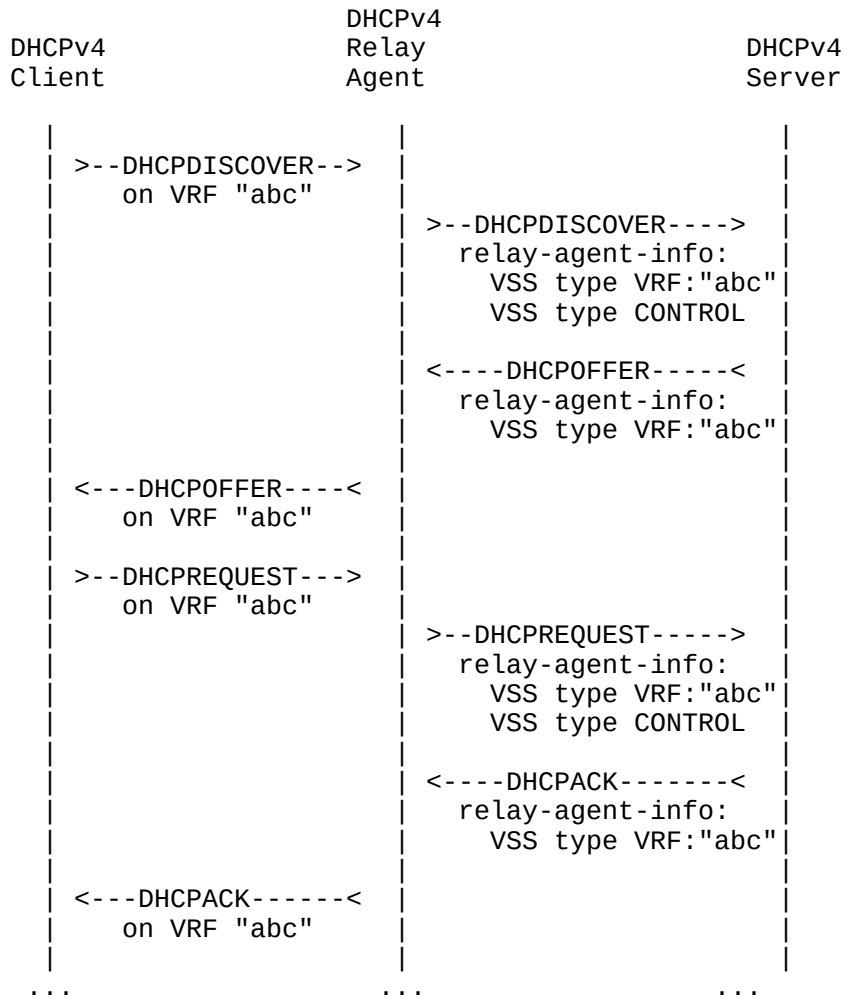


Figure 4.1-1: DHCPv4 - Relay Agent knows VPN

The DHCP server would know that it should respond to VPN information specified in a VSS option or sub-option, and it would be configured with appropriate VPN address spaces to service the projected client requirements. Thus, in this common approach, the DHCP client knows nothing of any VPN access, the relay agent has been configured in some way that allows it to determine the VPN of the DHCP client and transmit that using a VSS option or sub-option to the DHCP server, and the DHCP server responds to the VPN specified by the relay agent. There is no conflict between different entities trying to specify

different VSS information -- each entity knows its role through policy or configuration external to this document.

If any mis-configuration exists, it SHOULD result in a DHCP client being unable to acquire an IP address. For instance, a relay agent which supports VPN access SHOULD couple transmission of VSS options or sub-options to the configuration of VPN support, and not allow one without the other.

It is important to ensure that the relay agent and DHCP server both support the VSS option and sub-option (for DHCPv4) or the VSS option (for DHCPv6). Deploying DHCPv4 relay agents which support and emit VSS sub-options in concert with DHCPv4 servers which do not support the VSS option or sub-option as defined in this document SHOULD NOT be done, as such an ensemble will not operate correctly. Should this situation occur, however, the relay agent can detect the problem (since the VSS sub-option with type CONTROL will appear in the packets it receives from the DHCPv4 server), and it can issue appropriate diagnostic messages.

4.2. VPN assignment by the DHCP server

In this approach, the DHCP server would be configured in some way to know the VPN on which a particular DHCP client should be given access. The DHCP server would in this case include the VSS sub-option in the relay-agent-information option for DHCPv4 or the VSS option in the Relay-reply message for DHCPv6. The relay agent responsible for mediating VPN access would use this information to select the correct VPN for the DHCP client. In the unusal event that there were more than one relay agent involved in this transaction, some external configuration or policy would be needed to inform the DHCPv6 server into which Relay-reply message the VSS option should go.

Once the relay agent has placed the DHCP client into the proper VPN, it SHOULD begin including VSS information in requests that it forwards to the DHCP server. Since this information does not conflict with the DHCP server’s idea of the proper VPN for the client, everything works correctly.

The diagram below shows this approach using DHCPv4. The DHCPv6 situation is similar, but uses the DHCPv6 VSS option instead.

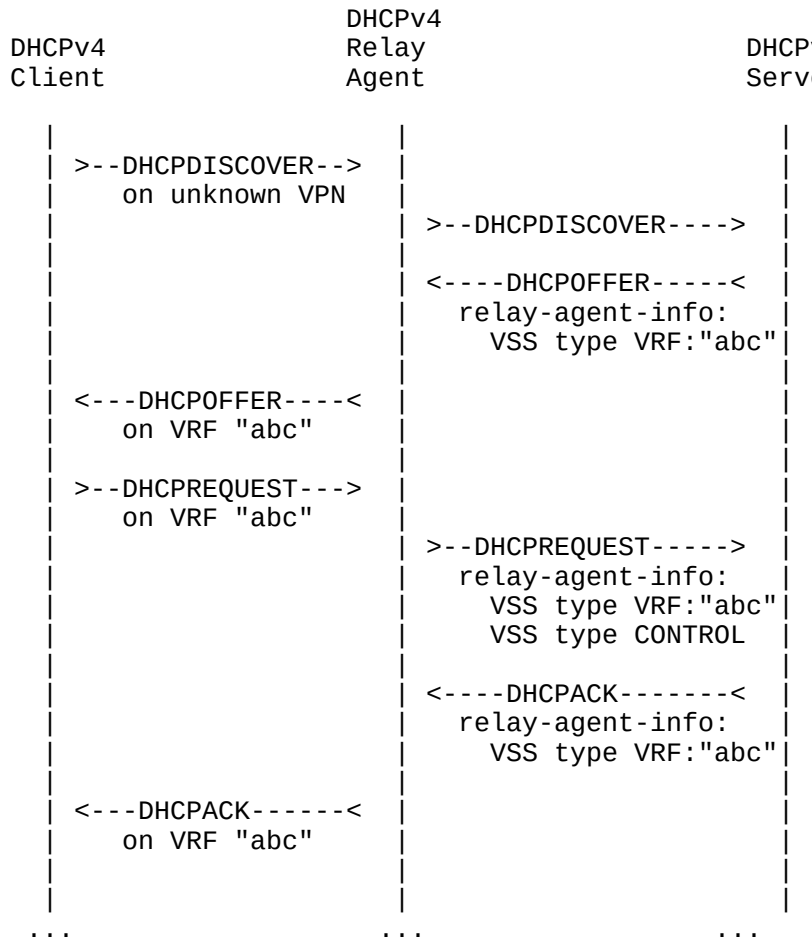


Figure 4.2-1: DHCPv4 - DHCPv4 Server knows VPN

In this approach, the DHCP client is again unaware of any VPN activity. In this case, however, the DHCP server knows the VPN for the client, and the relay agent responds to the VSS information specified by the DHCP server. Similar to the previous approach, each entity knows its role through a means external to this document and no two entities try to specify VSS information in conflict.

It is important that both the relay agent as well as the DHCP server both support the VSS option and sub-option (for DHCPv4) and the VSS option (for DHCPv6). Deploying and configuring VPN support in one element and not in the other is not a practical approach.

4.3. Required Support

DHCP relay agents and servers MUST support the approach discussed in Section 4.1. DHCP relay agents and server SHOULD support the approach discussed in Section 4.2. DHCP relay agents and servers SHOULD NOT be configured to operate with both approaches simultaneously.

4.4. Alternative VPN assignment approaches

There are many other approaches which can be created with multiple relay agents each inserting VSS information into different Relay-forward messages, relay agent VSS information conflicting with client VSS information, or DHCP server VSS information conflicting with relay agent and client VSS information. Since these approaches do not describe situations that are useful today, specifying precisely how to resolve all of these conflicts is unlikely to be valuable in the event that these approaches actually become practical in the future.

The current use of the VSS option and sub-option require that each entity knows the part that it plays in dealing with VPN data. Each entity -- client, relay agent or agents, and server -- SHOULD know through some policy or configuration beyond the scope of this document whether it is responsible for specifying VPN information using the VSS option or sub-option or responsible for responding to VSS information specified by another entity, or simply ignoring any VSS information which it might see.

Some simple conflict resolution approaches are discussed below, in the hopes that they will cover simple cases that may arise from situations beyond those envisioned today. However, for more complex situations, or simple situations where appropriate conflict resolution strategies differ from those discussed in this document, a document detailing the usage situations and appropriate conflict resolution strategies SHOULD be created and submitted for discussion and approval.

5. Relay Agent Behavior

A relay agent which receives a DHCP request from a DHCP client on a VPN SHOULD include Virtual Subnet Selection information in the DHCP packet prior to forwarding the packet on to the DHCP server unless inhibited from doing so by configuration information or policy to the contrary.

In this situation, a DHCPv4 relay agent MUST include a DHCPv4 VSS sub-option in a relay-agent-information option [RFC3046], while a

DHCPv6 relay agent MUST include a DHCPv6 VSS option in the Relay-forward message.

The value placed in the Virtual Subnet Selection sub-option or option would typically be sufficient for the relay agent to properly route any DHCP reply packet returned from the DHCP server to the DHCP client for which it is destined. In some cases, the information in the VSS sub-option or option might be an index into some internal table held in the relay agent, though this document places no requirement on a relay agent to have any such internal state.

A DHCPv4 relay agent SHOULD, in addition, include a DHCPv4 VSS sub-option with a type of CONTROL, no additional VSS information, and a length of one, in the relay-agent-information option [RFC3046]. The inclusion of two VSS sub-options in the relay-agent-information option, one with valid VSS information, and one with a type of CONTROL, will allow the DHCPv4 relay agent to determine whether the DHCPv4 server actually processed the information in the VSS sub-option containing valid VSS information.

The reason to include this additional VSS DHCPv4 sub-option is that [RFC3046] specifies (essentially) that a DHCPv4 server should copy all sub-options that it receives in a relay-agent-information option in a request into a corresponding relay-agent-information option in the response. Thus, a server that didn’t support the DHCPv4 VSS sub-option would normally just copy it to the response packet, leaving the relay agent to wonder if in fact the DHCPv4 server actually used the VSS information when processing the request.

To alleviate this potential confusion, a DHCPvr4 relay agent instead sends in two VSS sub-options, one with valid VSS information, and one with a VSS type of CONTROL. If both sub-options appear in the response from the DHCPv4 server, then the DHCPv4 relay agent MUST assume that the DHCPv4 server did not act on the valid VSS information in one of the sub-options. If only the VSS sub-option with the valid information appears in the response from the DHCPv4 server and no VSS sub-option with type CONTROL appears in the response from the DHCPv4 server, then the relay agent SHOULD assume that the DHCPv4 server acted successfully on the VSS sub-option with the valid VSS information.

Anytime a relay agent places a VSS option or sub-option in a DHCP request, it SHOULD send it only to a DHCP server which supports the VSS option or sub-option, and it MUST check the response to determine if the DHCP server actually honored the requested VSS information.

In the DHCPv6 case, the appearance of the option in the Relay-reply packet indicates that the DHCPv6 server understood and acted upon the

contents of the VSS option in the Relay-forward packet. In the DHCPv4 case, as discussed above, the appearance of the VSS sub-option containing valid VSS information without the appearance of a VSS sub-option of type CONTROL indicates that the DHCPv4 server successfully acted upon the VSS sub-option that was returned containing valid VSS information.

This document does not create a requirement that a relay agent remember the contents of a VSS DHCPv4 sub-option or VSS DHCPv6 option sent to a DHCP server. In many cases, the relay agent may simply use the value of the VSS returned by the DHCP server to forward the response to the DHCP client. If the VSS information, the IP address allocated, and the VPN capabilities of the relay agent all interoperate correctly, then the DHCP client will receive a working IP address. Alternatively, if any of these items don't interoperate with the others, the DHCP client will not receive a working address.

Note that in some environments a relay agent may choose to always place a VSS option or sub-option into packets and messages that it forwards in order to forestall any attempt by a relay agent closer to the client or the client itself to specify VSS information. In this case, a type field of 255 is used to denote the global, default VPN. When the type field of 255 is used, there MUST NOT be any additional VSS information in the VSS option or sub-option. In the DHCPv4 case, an additional VSS sub-option with type CONTROL should be used, as discussed above.

5.1. VPN assignment by the DHCP server

In some cases, a DHCP server may use the Virtual Subnet Selection sub-option or option to inform a relay agent that a particular DHCP client is associated with a particular VPN. It does this by sending the Virtual Subnet Selection sub-option or option with the appropriate information to the relay agent in the relay-agent-information option for DHCPv4 or the Relay-reply message in DHCPv6. If the relay agent cannot respond correctly to the DHCP server's requirement to place the DHCP client into that VPN (perhaps because it has not been configured with a VPN that matches the VSS information received from the DHCP server) it MUST drop the packet and not send it to the DHCP client.

In this situation, once the relay agent has placed the DHCP client into the VPN specified by the DHCP server, it will insert a VSS option or sub-option when forwarding packets from the client. The DHCP server in normal operation will echo this VSS information into the outgoing replies.

In the event that the relay agent doesn't include VSS information on

subsequent requests after the DHCP server has included VSS information in a reply to the relay agent, the DHCP server can conclude that the relay agent doesn't support VSS processing, and the DHCP server SHOULD stop processing this transaction and not respond to the request.

5.2. DHCP Leasequery

Sometimes a relay-agent needs to submit a DHCP Leasequery [RFC4388] [RFC5007] packet to the DHCP server in order to recover information about existing DHCP allocated IP addresses on other than the normal, global VPN. In the context of a DHCP Leasequery the relay agent is a direct client of the DHCP server and is not relaying a packet for another DHCP client. Thus, the instructions in Section 6 on Client Behavior should be followed to include the necessary VSS information.

6. Client Behavior

Typically, DHCPv4 and DHCPv6 clients have no interaction with VSS options or sub-options. The VSS information is handled by exchanges between a DHCPv4 or DHCPv6 relay agent and the corresponding DHCPv4 or DHCPv6 server.

However, there are times when an entity is acting as a DHCPv4 or DHCPv6 client in that it is communicating directly with a DHCPv4 or DHCPv6 server. In these instances -- where communications is occurring without employing the DHCPv4 relay-agent-information option or the DHCPv6 Relay-forward or Relay-reply messages, the entity is acting as a DHCPv4 or DHCPv6 client with regard to its communication with the DHCPv4 or DHCPv6 server, but not necessarily as a DHCP client who is requesting a DHCPv4 or DHCPv6 address for its own use.

The client, in this context, may be requesting an IP address for another entity, thus acting as a DHCP proxy. The client may be requesting information about another client-to-address binding, using the DHCPv4 [RFC4388] or DHCPv6 [RFC5007] Leasequery protocol.

In the rest of this section, the term "client" refers to an entity communicating VSS information directly to a DHCPv4 or DHCPv6 server without using the DHCPv4 relay-agent-information option or the DHCPv6 Relay-forward or Relay-reply messages, and there is no requirement that such a client is a traditional DHCPv4 or DHCPv6 client requesting an IP address binding for itself.

A DHCPv4 or DHCPv6 client will employ the VSS option to communicate VSS information to their respective servers. This information MUST be included in every message concerning any IP address on a different VPN than the global or default VPN. A DHCPv4 client will place the

DHCPv4 VSS option in its packets, and a DHCPv6 client will place the DHCPv6 VSS option in its messages.

A DHCPv6 client that needs to place a VSS option into a DHCPv6 message SHOULD place a single VSS option into the DHCPv6 message at the same level as the Client Identifier option. A DHCPv6 client MUST NOT include different VSS options in the same DHCPv6 message.

Note that, as mentioned in Section 1, throughout this document when a DHCPv6 address is indicated the same information applies to DHCPv6 Prefix Delegation [RFC3633] as well.

Since this option is placed in the packet in order to change the VPN on which an IP address is allocated for a particular DHCP client, one presumes that an allocation on that VPN is necessary for correct operation. Thus, a client which places this option in a packet and doesn't receive it or receives a different value in a returning packet SHOULD drop the packet since the IP address that was allocated will not be in the requested VPN.

Clients should be aware that some DHCP servers will return a VSS option with different values than that which was sent in. In addition, a client may receive a response from a DHCP server with a VSS option when none was sent in by the Client.

Note that when sending a DHCP Leasequery request, a relay agent is acting as a DHCP client and so it SHOULD include the respective DHCPv4 or DHCPv6 VSS option in its DHCPv4 or DHCPv6 Leasequery packet if the DHCP Leasequery request is generated for other than the default, global VPN. It SHOULD NOT include a DHCPv4 sub-option in this case.

7. Server Behavior

A DHCP server receiving the VSS option or sub-option SHOULD allocate an IP address (or use the VSS information to access an already allocated IP address) from the VPN specified by the included VSS information.

In the case where the type field of the VSS option or sub-option is 255, the VSS option denotes the global, default VPN. In this case, there is no explicit VSS information beyond the type field.

This document does not prescribe any particular address allocation policy. A DHCP server may choose to attempt to allocate an address using the VSS information and, if this is impossible, to not allocate an address. Alternatively, a DHCP server may choose to attempt address allocation based on the VSS information and, if that is not

possible, it may fall back to allocating an address on the global or default VPN. This, of course, is also the apparent behavior of any DHCP server which doesn't implement support for the VSS option and sub-option. Thus, DHCP clients and relay agents SHOULD be prepared for either of these alternatives.

In some cases, a DHCP server may use the Virtual Subnet Selection sub-option or option to inform a relay agent that a particular DHCP client is associated with a particular VPN. It does this by sending the Virtual Subnet Selection sub-option or option with the appropriate information to the relay agent in the relay-agent-information option for DHCPv4 or the Relay-reply message in DHCPv6.

In this situation, the relay agent will place the client in the proper VPN, and then it will insert a VSS option or sub-option in subsequent forwarded requests. The DHCP server will see this VSS information and since it doesn't conflict in any way with the server's notion of the VPN on which the client is supposed to reside, it will process the requests based on the VPN specified in the VSS option or sub-option, and echo the same VSS information in the outgoing replies.

The relay agent receiving a reply containing a VSS option should support the VSS option. Otherwise the relay agent will end up attempting to use the address as though it were a global address. Should this happen, the subsequent DHCPREQUEST will not contain any VSS information, in which case the DHCP server SHOULD NOT respond with a DHCPACK.

If a server uses a different VPN than what was specified in the VSS option or sub-option, it SHOULD send back the VPN information using the same type as the received type. It MAY send back a different type if it is not possible to use the same type (such as the RFC2685 VPN-ID if no ASCII VPN identifier exists).

7.1. Returning the DHCPv4 or DHCPv6 Option

DHCPv4 or DHCPv6 servers receiving a VSS option (for sub-option processing, see below) MUST return an instance of this option in the reply packet or message if the server successfully uses this option to allocate an IP address, and it MUST NOT include an instance of this option if the server is unable to support, is not configured to support, or does not implement support for VSS information in general or the requested VPN in particular.

If they echo the option (based on the criteria above), servers SHOULD return an exact copy of the option unless they desire to change the VPN on which a client was configured.

The appearance of the DHCPv4 VSS option code in the DHCPv4 Parameter Request List option [RFC2132] should not change the processing or decision to return or not return the VSS option as specified in this document. The appearance of the DHCPv6 VSS option in the OPTION_ORO [RFC3315] or the OPTION_ERO [RFC4994] should not change the processing or decision to return (or not to return) the VSS option as specified in this document.

7.2. Returning the DHCPv4 Sub-Option

The case of the DHCPv4 sub-option is a bit more complicated. Note that [RFC3046] specifies that a DHCPv4 server which supports the relay-agent-information option SHALL copy all sub-options received in a relay-agent-information option into any outgoing relay-agent-information option. Thus, the default behavior for any DHCPv4 server is to return any VSS sub-option received to the relay agent whether or not the DHCPv4 server understands the VSS sub-option.

In order to distinguish a DHCPv4 server which is simply copying relay-agent-information option sub-options from an incoming to an outgoing relay-agent-informaion option from one which successfully acted upon the information in the VSS sub-option, DHCPv4 relay agents MUST include two VSS sub-options in the relay-agent-information in the request. One of these VSS sub-options contains valid VSS information, and one of these VSS sub-options has a type of CONTROL, no additional VSS information, and a length of one.

A DHCPv4 server which does not support the VSS sub-option will copy both sub-options into the outgoing relay-agent-information option, thus signalling to the DHCPv4 relay agent that it did not understand the VSS sub-option.

A DHCPv4 server which supports the VSS sub-option and acts upon the VSS sub-option with valid VSS information in it:

- MUST copy the VSS sub-option containing the valid VSS information into the outgoing relay-agent-information option
- MUST NOT copy the VSS sub-option with the type of CONTROL into the outgoing relay-agent-information option

Moreover, if a server uses different VSS information to allocate an IP address than it receives in a particular DHCPv4 sub-option, it MUST include that alternative VSS information in the VSS sub-option that it returns to the DHCPv4 relay agent instead of the origian VSS information it was given.

If a DHCPv4 server supports this sub-option and for some reason (perhaps administrative control) does not honor this sub-option from the request then it MUST NOT echo either sub-option into the outgoing relay-agent-information option.

7.3. Making sense of conflicting VSS information

It is possible for a DHCPv4 server to receive both a VSS option and VSS sub-options in the same packet. Likewise, a DHCPv6 server can receive multiple VSS options in nested Relay-forward messages as well as in the client message itself. In either of these cases, the VSS information from the relay agent closest to the DHCP server SHOULD be used in preference to all other VSS information received. In the DHCPv4 case, this means that the VSS sub-option takes precedence over the VSS option, and in the DHCPv6 case, this means that the VSS option from the outer-most Relay-forward message in which a VSS option appears takes precedence.

The reasoning behind this approach is that the relay-agent closer to the DHCP server is almost certainly more trusted than the DHCP client or more distant relay agents, and therefore information in the relay-agent-information option or the Relay-forward message is more likely to be correct.

In general, relay agents SHOULD be aware through configuration or policy external to this document whether or not they should be including VSS information in packets that they forward and so there should not be conflicts among relay agent specified VSS information.

In these situations where multiple VSS option or sub-options appear in the incoming packet or message, when the DHCP server constructs the response to be sent to the DHCP client or relay agent, all existing VSS options or sub-options MUST be replicated in the appropriate places in the response and MUST contain only the VSS information that was used by the DHCP server to allocate the IP address (with, of course, the exception of a DHCPv4 relay-agent-information VSS sub-option with a type of CONTROL).

8. Security

Message authentication in DHCPv4 for intradomain use where the out-of-band exchange of a shared secret is feasible is defined in [RFC3118]. Potential exposures to attack are discussed in section 7 of the DHCP protocol specification in [RFC2131].

Implementations should consider using the DHCPv4 Authentication option [RFC3118] to protect DHCPv4 client access in order to provide a higher level of security if it is deemed necessary in their

environment.

Message authentication in DHCPv4 relay agents as defined in [RFC4030] should be considered for DHCPv4 relay agents employing this sub-option. Potential exposures to attack are discussed in section 7 of the DHCP protocol specification in [RFC2131].

For DHCPv6 use of the VSS option, the "Security Considerations" section of [RFC3315] details the general threats to DHCPv6, and thus to messages using the VSS option. The "Authentication of DHCP Messages" section of [RFC3315] describes securing communication between relay agents and servers, as well as clients and servers.

The VSS option could be used by a client in order to obtain an IP address from any VPN. This option would allow a client to perform a more complete address-pool exhaustion attack since the client would no longer be restricted to attacking address-pools on just its local subnet.

A DHCP server that implements these options and sub-option should be aware of this possibility and use whatever techniques that can be devised to prevent such an attack. Information such as the giaddr in DHCPv4 or link address in the Relay-forward DHCPv6 message might be used to detect and prevent this sort of attack.

One possible defense would be for the DHCP relay to insert a VSS option or sub-option to override the DHCP client’s VSS option.

Servers that implement the VSS option and sub-option MUST by default disable use of the feature; it must specifically be enabled through configuration. Moreover, a server SHOULD provide the ability to selectively enable use of the feature under restricted conditions, e.g., by enabling use of the option only from explicitly configured client-ids, enabling its use only by clients on a particular subnet, or restricting the VSSs from which addresses may be requested.

9. IANA Considerations

IANA is requested to assign DHCPv4 option number 221 for the DHCPv4 VSS option defined in Section 3.1, in accordance with [RFC3942].

IANA is requested to assign sub-option number 151 for the DHCPv4 sub-option defined in Section 3.2 from the DHCP Relay Agent Sub-options space [RFC3046], in accordance with the spirit of [RFC3942]. While [RFC3942] doesn’t explicitly mention the sub-option space for the DHCP Relay Agent Information option [RFC3046], sub-option 151 is already in use by existing implementations of this sub-option and the current draft is essentially compatible with these current

implementations.

IANA is requested to assign the value of TBD for the DHCPv6 VSS option defined in Section 3.3 from the DHCPv6 option registry.

The type byte defined in Section 3.4 defines a number space for which IANA is to create and maintain a new sub-registry entitled "VSS Type values". This sub-registry needs to be related to both the DHCPv4 and DHCPv6 VSS options and the DHCPv4 relay-agent-information option sub-option (all defined by this document), since the type byte in these two options and one sub-option MUST have identical definitions.

New values for the type byte may only be defined by IETF Consensus, as described in [RFC5226]. Basically, this means that they are defined by RFCs approved by the IESG.

10. Acknowledgments

Bernie Volz recommended consolidation of the DHCPv4 option and sub-option drafts after extensive review of the former drafts, and provided valuable assistance in structuring and reviewing this document. Alper Yegin expressed interest in the DHCPv6 VSS option, resulting in this combined draft covering all three areas. Alfred Hoenes provided assistance with editorial review as well as raising substantive protocol issues. David Hankins and Bernie Volz each raised important protocol issues which resulted in a clarified document. Josh Littlefield provided editorial assistance. Several IESG reviewers took the time to substantially review this document, resulting in much increased clarity.

11. References

11.1. Normative References

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.

[RFC2685] Fox, B., Gleeson, B., "Virtual Private Networks Identifier", RFC 2685, September 1999.

[RFC3046] Patrick, M., "DHCP Relay Agent Information Option", RFC 3046, January 2001.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

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

[RFC4994] Zeng, S., Volz, B., Kinnear, K. and J. Brzozowski, "DHCPv6 Relay Agent Echo Request Option", RFC 4994, September 2007.

11.2. Informative References

[RFC951] Croft, B. and J. Gilmore, "Bootstrap Protocol", RFC 951, September 1985.

[RFC1542] Wimer, W., "Clarifications and Extensions for the Bootstrap Protocol", RFC 1542, October 1993.

[RFC3118] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.

[RFC3942] Volz, B., "Reclassifying Dynamic Host Configuration Protocol version 4 (DHCPv4) Options", RFC 3942, November 2004.

[RFC4030] Stapp, M. and T. Lemon, "The Authentication Suboption for the Dynamic Host Configuration Protocol (DHCP) Relay Agent Option", RFC 4030, March 2005.

[RFC4388] Woundy, R. and K. Kinnear, "Dynamic Host Configuration Protocol (DHCP) Leasequery", RFC 4388, February 2006.

[RFC5007] Brzozowski, J., Kinnear, K., Volz, B., and S. Zeng, "DHCPv6 Leasequery", RFC 5007, September 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

Authors' Addresses

Kim Kinnear
Cisco Systems

1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: kkinnear@cisco.com

Richard Johnson
Cisco Systems
170 W. Tasman Dr.
San Jose, CA 95134

Phone: (408) 526-4000

EMail: raj@cisco.com

Mark Stapp
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: mjs@cisco.com

Jay Kumarasamy

Network Working Group   Sheng Jiang
Internet Draft   Sam(Zhongqi) Xia
Intended status: Standards Track   Huawei Technologies Co., Ltd
Expires: February 24, 2011   August 24, 2010

Configuring Cryptographically Generated Addresses (CGA) using DHCPv6
draft-jiang-dhc-cga-config-dhcpv6-01.txt

Status of this Memo

Copyright Notice

Abstract

A Cryptographically Generated Address is an IPv6 addresses binding with a public/private key pair. However, the current CGA specifications are lack of procedures to enable proper management of CGA generation. Administrators should be able to configure parameters used to generate CGA. The Dynamic Host Configuration Protocol for IPv6 (DHCPv6), which enables network management to dynamically configure hosts, can be used in the CGA configuration. This document defines the process using DHCPv6 to configure CGA in detail, include configuration CGA parameters and that a DHCPv6 server grants the CGA usage. Two new DHCPv6 options are defined accordingly.

Table of Contents

1. Introduction

Cryptographically Generated Addresses (CGA, [RFC3972]) provide means to verify the ownership of IPv6 addresses without requiring any security infrastructure such as a certification authority. The use of CGAs allows identity verification in different protocols, such as SEure Neighbor Discovery (SEND, [RFC3971]), Enhanced Route Optimization for MIPv6 [RFC4866] or Site Multihoming by IPv6 Intermediation (SHIM6 [RFC5533]).

However, as [I-D.ietf-csi-dhcpv6-cga-ps] analyses, in the current specifications, there is a lack of procedures to enable proper management of CGA generation, in particular, in the configuration of the parameters that define the security properties of the addresses. Administrators should be able to configure parameters used to generate CGA. The Dynamic Host Configuration Protocol for IPv6 (DHCPv6), which enables network management to dynamically configure hosts, can be used in the CGA configuration. For example, DHCPv6 server should be able to assign certain level of CGA Sec value or other relevant parameters to CGA address owner.

This draft provides detailed solutions for CGA configuration. Two existing DHCPv6 options are re-used. Two new DHCPv6 options, CGA Sec Option and Address Grant Option, are also defined in this document.

2. Terminology

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

3. CGA Configure Process Using DHCPv6

The CGA specifications [RFC3972] define the procedure to generate a CGA. However, it assumes that hosts decide by itself or have been preconfigured all CGA relevant parameters. In reality, network management MAY want to assign/enforcement some parameters to hosts.

Among the mechanisms in which configuration parameters could be pushed to the end hosts and/or CGA related information sent back to a central administration, we discuss the stateful configuration mechanism based on DCHPv6 in this document. Other mechanisms may also provide similar functions, but out of scope.

In this section, configuration CGA parameters and that a DHCPv6 server grants the CGA usage are described in details.

3.1. Configuration of the parameters required for the generation of CGA

Each CGA is associated with a CGA Parameters data structure, which is formed by all input parameters [RFC3972] except for Sec value that is embedded in the CGA. The CGA associated Parameters used to generate a CGA includes:

- a Public Key,

- a Subnet Prefix,

- a 3-bit security parameter, Sec. Additionally, it should be noted that the hash algorithm to be used in the generation of the CGA is also defined by the Sec value [RFC4982],

- any Extension Fields that could be used.

- Note: the modifier and the Collision Count value in the CGA Parameter data structure are generated during the CGA generation process. They do NOT need to be configured.

A node may initiate a request for the relevant CGA configuration information needed to the DHCPv6 server. The server responds with the configuration information for the node. The Option Request Option, defined in Section 22.7 in [RFC3315], can be used for node to indicate which options the client requests from the server. For response, the requested Option should be included. The server MAY also initiatively push these parameters by attaching these option in the response messages which are initiated for other purposes.

The Public/Private key pair is generated by hosts themselves and considered not suitable for network transmission for security reasons. The configuration of the client key pair or certificate is out of scope.

Currently, there are convenient mechanisms for allowing an administrator to configure the subnet prefix for a host, by Router Advertisement [RFC4861, RFC4862]. However, this does not suitable for the DHCP-managed network. To propagate the prefix through DHCP interactions, DHCPv6 Prefix Delegation Option [RFC3633] MAY be used. However, this option was designed to assign prefix block for routers. A new Prefix Assignment Option MAY need to be defined. Since alternative approach is existing and there are debates

whether a new Prefix Assignment Option MAY is necessary, this document does not define it.

A new DHCPv6 CGA Sec Option is defined in Section 4.

Although there is an optional Extension Fields in CGA Parameter data structure, there is NO any defined extension fields. Future specification may define more options to carry CGA-related configuration parameters.

Upon reception of the CGA relevant parameters from DHCPv6 server, the end hosts SHOULD generate addresses compliant with the received parameters. If the parameters change, the end hosts SHOULD generate new addresses compliant with the parameters propagated.

3.2. Node requests CGA Approved to the DHCPv6 server

A CGA address is generated by the associated key pair owner, normally an end host. However, in a DHCPv6-managed network, hosts should use IPv6 global addresses only from a DHCPv6 server. The process described below allows a host, also DHCPv6 client, uses self-generated CGAs in a DHCPv6-managed environment, by requesting the granting from a DHCPv6 server.

The client sends a CGA, which is generated by itself, to a DHCPv6 server, and requests the DHCP server to determine whether the generated CGA satisfies the requirements of the network configuration, wherein the network configuration comprises a CGA security level set by the DHCP; and generates a new CGA if the generated CGA does not satisfy the requirements of the network configuration.

Client initiation behavior

In details, a DHCPv6 client SHOULD send a DHCPv6 Request message to initiate the CGA granting process.

This DHCPv6 Request message MUST include an Option Request option, which requests Address Grant Option, defined in Section 5 in this document, to indicate the DHCPv6 server responses with the address granting decision. The Addr_Grant field in the embedded Address Grant Option should be set 0.

The client MUST include one or more IA Options, either IA_NA or IA_TA, in the Request message. Each IA Option MUST include one or more IA Address Options. CGAs are carried in the IA Address Options.

Server behavior

Upon reception of the Request message, the DHCPv6 server SHOULD verify whether the client's CGAs satisfy the CGA-related configuration parameters of the network. The DHCPv6 server then send an acknowledgement, a Reply message, to the client to either grant the use of the CGA or to indicate that the node must generate a new CGA with the correct CGA-related configuration parameters of the network. The Addr_Grant field sets 1 indicating that the requested address is granted; The Addr_Grant field sets 2 indicating that the requested address is declined, defined in Section 5 in this document. When the requested CGA is declined, the DHCPv6 server may attach CGA-relevant parameters in the Reply message to indicate the client generates a new CGA accordingly.

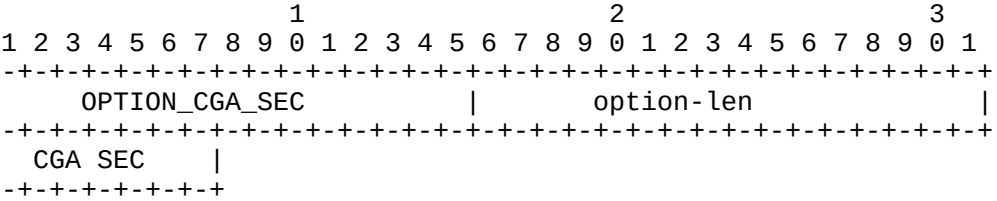
In the meantime, the DHCPv6 server MAY log the requested CGA addresses. This information MAY later be used by other network functions, such as ACL.

Client receiving behavior

Upon reception of the acknowledgement from server, the client can legally use the granted CGAs. If the server declines the requested CGA, the client MUST generate a new CGA. If the server replies with CGA-relevant parameters, the client SHOULD generate a new CGA accordingly.

4. DHCPv6 CGA Sec Option

DHCPv6 CGA Sec Option is used to carry a Sec value, the parameters associated with CGA generation on a client. After receiving the CGA Sec Option, the client SHOULD generate a CGA using a Sec value that is not lower than the option indicated.



option-code OPTION_CGA_SEC (TBA1).

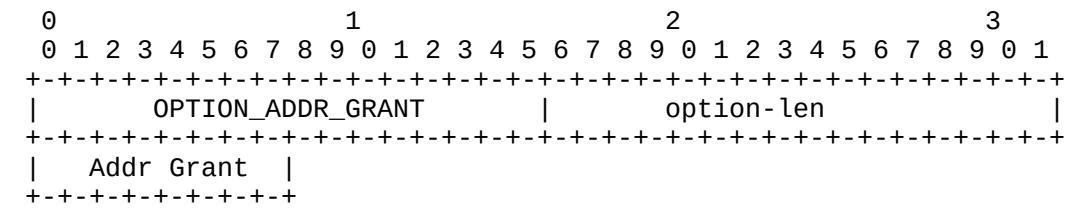
option-len 1.

CGA SEC a digit between 0 and 7, the SEC level.

Note: On receiving the CGA Sec Option, the client MAY choose to generate a CGA using a lower sec value. It MAY cause that the client is not able to use full network capabilities.

5. DHCPv6 Address Grant Option

DHCPv6 Address Grant Option is used to indicate the DHCPv6 client whether the requested address is granted or not.



option-code OPTION_ADDR_GRANT (TBA2).

option-len 1.

Addr Grant The Addr_Grant field sets 0 when a client requests granting from server. It sets 1 indicating that the requested Address is granted; it sets 2 indicating that the requested Address is declined.

6. Security Considerations

The mechanisms based on DHCPv6 are all vulnerable to attacks to the DHCP client, such as request for CGA generation with a high Sec value. Proper use of DHCPv6 autoconfiguration facilities [RFC3315], such as AUTH option or Secure DHCP [I-D.ietf-dhc-secure-dhcpv6] can prevent these threats, provided that a configuration token is known to both the client and the server.

Note that, as expected, it is not possible to provide secure configuration of CGA without a previous configuration of security information at the client (either a trust anchor, a DHCPv6 configuration token...). However, considering that the values of these elements could be shared by the nodes in the network segment, these security elements can be configured more easily in the end nodes than its addresses.

Regarding to the configuration of the Sec parameter, one risk is that a malicious node could propagate a Sec value providing less protection than intended by the network administrator, facilitating a

brute force attack against the hash, or the selection of the weakest hash algorithm available for CGA definition. However, even in the worst case, if the hash algorithm cannot be inverted, the expected number of iterations required for a brute force attack is O(2^59) in order to find a CGA Parameters data structure that matches a given CGA. Another risk is the use of a Sec, higher than intended by the administrator, which would require a large number of resources of the client to compute the modifier, requiring a long time before the device can communicate. This can be considered a kind of DOS attack. A variation of this attack is the propagation of different Sec values. This kind of attack may be prevented by protected DHCPv6 interactions.

7. IANA Considerations

This document defines two new DHCPv6 [RFC3315] options, which must be assigned Option Type values within the option numbering space for DHCPv6 messages:

The DHCPv6 CGA Sec Option (TBA1), described in Section 4.

The DHCPv6 Address Grant Option (TBA2), described in Section 5.

8. Acknowledgments

The authors would like to thank Marcelo Bagnulo Braun and Alberto Garcia-Martinez from Universidad Carlos III de Madrid for been involved in the early requirement identification. Valuable comments from Bernie Volz, Cisco and Dujuan Gu, Huawei are appreciated.

9. References

9.1. Normative References

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

[RFC3315] R. Droms, Ed., "Dynamic Host Configure Protocol for IPv6", RFC3315, July 2003.

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

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

[RFC3972] T. Aura, "Cryptographically Generated Address", RFC3972, March 2005.

[RFC4861] T. Narten, et al., "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

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

[RFC4866] J. Arkko, C. Vogt and W. Haddad, "Enhanced Route Optimization for Mobile IPv6", RFC4866, May 2007.

[RFC4982] M. Bagnulo, "Support for Multiple Hash Algorithms in Cryptographically Generated Addresses (CGAs) ", RFC4982, July 2007.

[RFC5533] E. Nordmark and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6" FRC 5533, June 2009.

9.2. Informative References

[I-D.ietf-csi-dhcpv6-cga-ps]
S. Jiang, S. Shen and T. Chown, "DHCPv6 and CGA Interaction: Problem Statement", draft-ietf-csi-dhcpv6-cga-ps (work in progress), June, 2010.

[I-D.ietf-dhc-secure-dhcpv6]
S. Jiang and S. Shen, "Secure DHCPv6 Using CGAs", draft-ietf-dhc-secure-dhcpv6 (work in progress), June 2010.

Author’s Addresses

Sheng Jiang
Huawei Technologies Co., Ltd
Huawei Building, No.3 Xinxi Rd.,
Shang-Di Information Industry Base, Hai-Dian District, Beijing 100085
P.R. China
Email: shengjiang@huawei.com

Sam(Zhongqi) Xia
Huawei Technologies Co., Ltd
Huawei Building, No.3 Xinxi Rd.,
Shang-Di Information Industry Base, Hai-Dian District, Beijing 100085
P.R. China
Email: xiazhongqi@huawei.com

dhc
Internet-Draft
Intended status: BCP
Expires: January 13, 2011

J. Brzozowski
Comcast Cable Communications
J. Tremblay
Videotron Ltd.
J. Chen
Time Warner Cable
T. Mrugalski
Gdansk University of Technology
July 12, 2010

DHCPv6 Redundancy Deployment Considerations
draft-jjmb-dhc-dhcpv6-redundancy-consider-01

Abstract

This document documents some deployment considerations for those who wishing to use DHCPv6 to support their deployment of IPv6. Specifically, providing semi-redundant DHCPv6 services is discussed in this document.

Status of this Memo

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

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

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

This Internet-Draft will expire on January 13, 2011.

Copyright Notice

Table of Contents

1. Introduction

To support the deployment of IPv6 redundancy and high availability are required for many if not all components. This document provides information specific to the proposed near term approach for deploying semi-redundant DHCPv6 services in advance of DHCPv6 server implementations that support a standards based failover or redundancy protocol.

2. Scope and Assumptions

This document specifies an interim architecture to provide a semi-redundant DHCPv6 solution before the availability of vendor or standard based solutions. The proposed architecture may be used in wide range of networks, two notable deployment models are discussed: service provider and enterprise network environments. The described architecture leverages only existing and implemented DHCPv6 standards. This document does not address a standards based solution for DHCPv6 redundancy. In the absence of a standards based DHCPv6 redundancy protocol and implementation, some analogies are loosely drawn with the DHCPv4 failover protocol for reference. Specific discussions related to DHCPv4 failover and redundancy is out of scope for this document.

Although DHCPv6 redundancy may be useful in a wide range of scenarios, they may be generalized for illustration purposes in the two aforementioned. The following assumptions were made with regards to the existing DHCPv6 infrastructure, regardless of the model used:

1. At least two DHCPv6 servers are used to service to the same clients, but the number of servers is not restricted.

2. Existing DHCPv6 servers will not directly communicate or interact with one another in the assignment of IPv6 addresses and configuration information to requesting clients.

3. DHCPv6 clients are instructed to run stateful DHCPv6 to request at least one IPv6 address. Configuration information and other options like a delegated IPv6 prefix may be also requested.

4. Clients requesting IPv6 addresses, prefixes, and or options care of DHCPv6 must recognize and honor the DHCPv6 preference option. Furthermore, the requesting clients must process DHCPv6 ADVERTISE messages per [RFC3315] when the preference option is present.

5. DHCPv6 server failure does not imply failure of any other network service or protocol, e.g. TFTP servers. Redundancy of any

additional services configured by means of DHCPv6 are outside of scope of this document. For example, a single DHCPv6 server may configure multiple TFTP servers, with preference for each TFTP server, as specified in [I-D.ietf-dhc-dhcpv6-opt-netboot].

2.1. Service provider model

The service provider model represents cases, where end-user devices may be configured directly, without any intermediate devices (like home routers used in service provider model). DHCPv6 clients include cable modems, customer gateways or home routers, and end-user devices. In some cases hosts may be configured directly using the service provider DHCPv6 infrastructure or via intermediate router, that is in turn being configured by the provider DHCPv6 infrastructure. The service provider DHCPv6 infrastructure may be semi-redundant in either case. Cable modems, customer gateways or home routers, and end-user devices are commonly referred to as CPE (Customer Premises Equipment). The following additional assumptions were made, besides the ones made in Section 2:

1. The service provider edge routers and access routers (CMTS for cable or DSLAM/BRAS for DSL for example) are IPv6 enabled when required.

2. CPE devices are instructed to perform stateful DHCPv6 to request atleast one IPv6 address, delegated prefix, and or configuration information. CPE devices may also be instructed to leverage stateless DHCPv6 [RFC3736] to acquire configuration information only. This assumes that IPv6 address and prefix information has been acquired using other means.

3. The primary application of this BCP is for native IPv6 services. Use and applicability to transition mechanisms is out of scope for this document.

4. CPE devices must implement a stateful DHCPv6 client [RFC3315], support for DHCPv6 prefix delegation [RFC3633] or stateless DHCPv6 [RFC3736] may also be implemented.

2.2. Enterprise model

The enterprise model represents cases, where end-user devices are most often configured directly, without any intermediate devices (like home routers used in service provider model). However, enterprise IPv6 environments quite often use or require that DHCPv6 relay agents are in place to support the use of DHCPv6 for the acquisition of IPv6 addresses and or configuration information. The assumptions here extend those that are defined in the beginning of

Section 2:

1. DHCPv6 clients are hosts and are considered end nodes. Examples of such clients include computers, laptops, and possibily mobile devices.

2. DHCPv6 clients generally do not require the assignment of an IPv6 prefix delegation and as such do not support DHCPv6 prefix delegation [RFC3633].

3. Protocol requirements

The following sections outline the requirements that must be satisfied by DHCPv6 clients, relays, and servers to ensure the desired behavior is provided using pre-existing DHCPv6 server implementations as is. The objective is to provide a semi-redundant DHCPv6 service to support the deployment of IPv6 where DHCPv6 is required for the assignment of IPv6 addresses, prefixes, and or configuration information.

3.1. DHCPv6 Servers

This interim architecture requires DHCPv6 servers that are RFC 3315 [RFC3315] compliant and support the necessary options required to support this solution. Essential to the the use of the interim architecture is support for stateful DHCPv6 and the DHCPv6 preference option both which are specified in [RFC3315]. For deployment scenarios where IPv6 prefix delegation is employed DHCPv6 servers must support DHCPv6 prefix delegation as defined by [RFC3633]. Further, where stateless DHCPv6 is used support for [RFC3736] is required by DHCPv6 servers.

3.2. DHCPv6 Relays

There are no specific requirements regarding relays. However, it is implied that DHCPv6 relay agents must be [RFC3315] compliant and must support the ability to relay DHCPv6 messages to more than one destination minimally.

3.3. DHCPv6 Clients

DHCPv6 clients are required to be compliant to [RFC3315] and support the necessary options required to support this solution depending on the mode of operations and desired behavior. Where prefix delegation is required DHCPv6 clients will be required to support DHCPv6 prefix delegation as defined in [RFC3633]. Clients used with this semi-redundant DHCPv6 deployment model must support the acquistion of at

least one IPv6 address and configuration information using stateful DHCPv6 as specified by [RFC3315]. The use of stateless DHCPv6 which is also specified in [RFC3315] may also be supported. DHCPv6 client must recognize and adhere to the processing of the advertised DHCPv6 preference options sent by the DHCPv6 servers.

4. Deployment models

At the time of this writing a standards-based DHCPv6 redundancy protocol and implementations are not available. As a result DHCPv6 server implementations will be used as-is to provide best effort, semi-redundant DHCPv6 services. Behavior of the DHCPv6 services will in part be governed by the configuration used by each of the servers. Additionally, various aspects of the DHCPv6 protocol [RFC3315] will be leveraged to yield the desired behavior. No inter-server or inter-process communications will be used to coordinate DHCPv6 events and or activities. DHCP services for both IPv4 and IPv6 may operate simultaneously on the same physical server(s) or may operate on different ones.

4.1. Split Prefixes

In the split prefixes model, each DHCPv6 server is configured with a unique, non-overlapping range derived from the /64 prefix deployed for use within an IPv6 network. Distribution between two servers, for example, would require that an allocated /64 be split in two /65 ranges. 2001:db8:1:0001:0000::/65 and 2001:db8:1:0001:8000::/65 would be assigned to each DHCPv6 server for allocation to clients derived from 2001:db8:1:0001::/64 prefix.

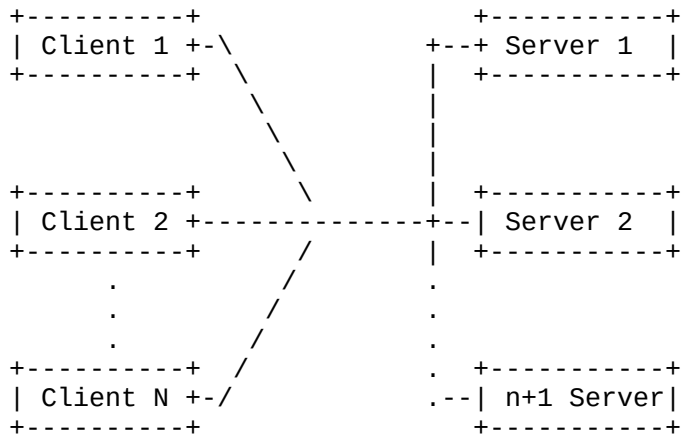
Each DHCP server allocates IPv6 addresses from the corresponding ranges per device class. Each DHCPv6 server will be simultaneously active and operational. Address allocation is governed largely through the use of the DHCPv6 preference option, so server with higher preference value is always prefered. Additional proprietary mechanisms can be leveraged to further enforce the favoring of one DHCP server over another. Example of such scenario is presented in Figure 1.

It is important to note that over time, it is possible that bindings may be disproportionally distributed amongst DHCPv6 servers and not any one server will be authoritative for all bindings. Per [RFC3315], a DHCPv6 ADVERTISE messages with a preference option of 255 is an indicator to a DHCPv6 client to immediately begin a client-initiated message exchange by transmitting a REQUEST message. Alternatively, a DHCPv6 ADVERTISE messages with a preference option of any value lesser than 255 or is absent is an indicator to the

client that it must wait for subsequent ADVERTISE messages (for a specified period of time) before proceeding. Additionally, in the event of a DHCPv6 server failure it is desirable for a server other than the server that originally responded to be able to rebind the client. It is not critical, that the DHCPv6 server be able to rebind the client in this scenario, however, this is generally desirable behavior. Given the proposed architecture, the remaining active DHCPv6 server will have a different range configured making it technically incorrect for the same to rebind the client in its current state. Ultimately, when rebinding fails the client will acquire a new binding from the configured range unique to an active server. Furthermore, shorter T1, T2, valid, and preferred lifetimes can be used to reduce the possibility that a client or some other element on the network will experience a disruption in service or access to relevant binding data. The values used for T2, preferred and valid lifetime can be adjusted or configured to minimize service disruption. Ideally T2, preferred and valid lifetimes that are equal or near equal can be used to trigger a DHCPv6 client to reacquire IPv6 address, prefix, and or configuration information almost immediately after rebinding fails. It is important to note that shorter values will most certainly create additional load and processing for the DHCPv6 server, which must be considered.

Using a split prefix configuration model dynamic updates to DNS can be coordinated to ensure that the DNS is properly updated with current binding information. Challenges arise with regards to the update of PTR for IPv6 addresses since the DNS may need to be overwritten in a failure condition. The use of a split prefixes enables the differentiation of bindings and binding timing to determine which represents the current state. This becomes particularly important when DHCPv6 Leasequery [RFC5007] and/or DHCPv6 Bulk Leasequery [RFC5460] are leveraged to determine lease or binding state. An additional benefit is that the use of separate ranges per DHCPv6 server makes failure conditions more obvious and detectable.

(@todo - add more useful illustration)



Server 1
========
Prefix=2001:db8:abcd:0000::/64
Range=2001:db8:abcd:5678:0000:/65
Preference=255

Server 2
========
Prefix=2001:db8:abcd:0000::/64
Range=2001:db8:abcd:5678:8000:/65
Preference=0

Server n+1
==========
Prefix, range, and preference would
vary based on range definition

Split prefixes approach.

Figure 1

4.2. Multiple Unique Prefixes

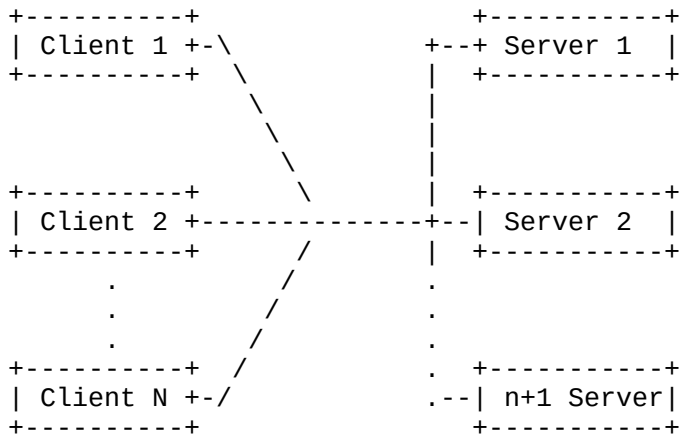
In multiple prefix model, each DHCPv6 server is configured with a unique, non-overlapping range derived from multiple unique prefixes deployed for use within an IPv6 network. Distribution between two servers, for example, would require that a /64 range be configured from an allocated from unique /64 prefixes. For example, the range 2001:db8:1:0001:0000::/64 would be assigned to a single DHCPv6 server for allocation to clients derived from 2001:db8:1:0001::/64 prefix, subsequently the 2001:db8:1:0001:1000::/64 from the prefix 2001:db8:

1:0001:1000::/64 could be used by a second DHCP server. This would be repeated for each active DHCP server. Example of this scenario is presented in Figure 2.

This approach uses a unique prefix and ultimately range per DHCPv6 server with corresponding prefixes configured for use in the network. The corresponding network infrastructure must in turn be configured to use multiple prefixes on the inteface(s) facing the DHCPv6 client. The configuration is similar on all the servers, but a different prefix and a different preference is used per DHCPv6 server.

This approach would drastically increase the rate of consumption of IPv6 prefixes and would also yield operational and management challenges related to the underlying network since a significantly higher number of prefixes would need to be configured and routed. This approach also does not provide a clean migration path to the desired solution leveraging a standards-based DHCPv6 redundancy or failover protocol, which of course has yet to be specified.

The use of multiple unique prefixes provides benefits similar to those referred to in Section 4.1 related to dynamic updates to DNS. The use of multiple unique prefixes enables the differentiation of bindings and binding timing to determine which represents the current state. This becomes particularly important when DHCPv6 Leasequery [RFC5007] and/or DHCPv6 Bulk Leasequery [RFC5460] are leveraged to determine lease or binding state. The use of separate prefixes and ranges per DHCPv6 server makes failure conditions more obvious and detectable.



Server 1
========
Prefix=2001:db8:abcd:0000::/64
Range=2001:db8:abcd:0000::/64
Preference=255

Server 2
========
Prefix=2001:db8:abcd:1000::/64
Range=2001:db8:abcd:1000::/64
Preference=0

Server 3
========
Prefix=2001:db8:abcd:2000::/64
Range=2001:db8:abcd:2000::/64
Preference=(>0 and <255)

Multiple unique prefix approach.

Figure 2

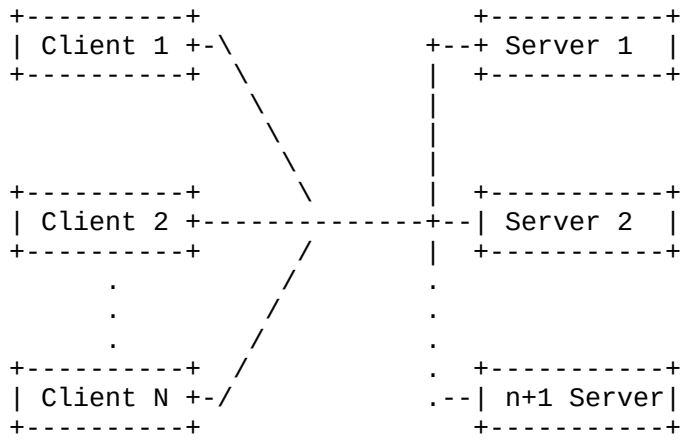
4.3. Identical Prefixes

In the identical prefix model, each DHCPv6 server is configured with the same overlapping prefix and range deployed for use within an IPv6 network. Distribution between two or more servers, for example, would require that the same /64 prefix and range be configured on all DHCP servers. For example, the range 2001:db8:1:0001:0000::/64 would be assigned to all DHCPv6 server for allocation to clients derived from 2001:db8:1:0001::/64 prefix. This would be repeated for each active DHCP server. Example of such scenario is presented in

Figure 3.

This approach uses the same prefix, length, and range definition across multiple DHCPv6 servers. All other configuration remaining the same the only other attribute of configuration option configured differently per DHCPv6 server would be DHCPv6 preference. This approach conceivably eases the migration of DHCPv6 services to fully support a standards based redundancy or failover protocol. Similar to the split prefix architecture described above this approach does not place any additional addressing requirements on network infrastructure.

The use of identical prefixes provides no benefit or advantage related to dynamic DNS updates, support of DHCPv6 Leasequery [RFC5007] or DHCPv6 Bulk Leasequery [RFC5460]. In this case all DHCP servers will use the same prefix and range configurations making it less obvious that a failure condition or event has occurred.




```
Server 1
========
Prefix=2001:db8:abcd:0000::/64
Range=2001:db8:abcd:0000::/64
Preference=255

Server 2
========
Prefix=2001:db8:abcd:0000::/64
Range=2001:db8:abcd:0000::/64
Preference=0

Server 3
========
Prefix=2001:db8:abcd:0000::/64
Range=2001:db8:abcd:0000::/64
Preference=(>0 and <255)
```

Identical prefix approach.

Figure 3

5. Challenges and Issues

The lack of interaction between DHCPv6 servers introduces a number of challenges related to the operations of the same in a production environment. The following areas of are particular concern.

- Interactions with DNS server(s) to support the dynamic update of the same adress and prefix when one or more DHCPv6 servers have become unavailable. This specifically becomes a challenge when or

if nodes that were initially granted a lease:

1. Attempt to renew or rebind the lease originally granted, or

2. Attempt to obtain a new lease

In either of the cases cited above, safeguards leveraged to prevent the deliberate or inadvertent overwriting of DNS data will likely prevent the responding DHCPv6 server from properly updating DNS with the client's new information and or may result in stale data in DNS. Possible solutions include the following:

* The ability to configure the override and or disabling of the safeguards that prevent the over-writing of DNS data care of RFC2136, specifically, related to [RFC4701] and [RFC4703]. This behavior must specifically be supported by the DHCPv6 server. This will allow for the overwriting of existing RRs in DNS that represent the former binding for the client. As a result clients will not have multiple RRs in DNS for a client's FQDN-to-IPv6 address mapping. Conversely, RR's for a client's IPv6 address-to-FQDN mapping will not be actively overwritten or deleted. Stale reverse zone data will be purged using well known DNS constructs, including but not limited to leveraging TTLs. Access control on the DNS server must be leveraged to restrict which DHCP servers may update DNS.

o Interactions with DHCPv6 servers to facilitate the acquisition of IPv6 lease data care of the DHCPv6 Leasequery [RFC5007] or DHCPv6 Bulk Leasequery [RFC5460] protocols when one or more DHCPv6 servers have become unavailable and have granted leases to DHCPv6 clients. If IPv6 lease data is required and the granting server is unavailable it will not be possible to obtain any information about leases granted until one of the following has taken place. It is important to note that with DHCPv6 until such time that a redundancy or failover protocol is available binding updates and synchronization will not occur between DHCPv6 servers.

1. The granting DHCPv6 server becomes available with all lease information restored

2. The client has renewed or rebound its lease against a different DHCPv6 server

6. IANA Considerations

IANA is not requested to assign any numbers at this time.

7. Security Considerations

Security considerations specific to the operation of the DHCPv6 protocol are created through the use of this interim architecture for DHCPv6 redundancy beyond what has been cited for Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315]. There are considerations related to DNS, specifically the dynamic updating of DNS, when such models are employed. Potential opportunities are created to overwrite valid DNS resource records when provisions have been made accommodate some of the models cited in this document. In some cases this is desirable to ensure that DNS remains up to date when using one or more of these models, however, abuse of the same could result in undesirable behavior.

8. Acknowledgements

Many thanks to Bernie Volz, Kim Kinnear, and Ralph Droms for their input and review.

9. Normative References

[I-D.ietf-dhc-dhcpv6-opt-netboot]
Huth, T., Freimann, J., Zimmer, V., and D. Thaler, "DHCPv6 option for network boot", draft-ietf-dhc-dhcpv6-opt-netboot-09 (work in progress), June 2010.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

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

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

[RFC4701] Stapp, M., Lemon, T., and A. Gustafsson, "A DNS Resource Record (RR) for Encoding Dynamic Host Configuration Protocol (DHCP) Information (DHCID RR)", RFC 4701, October 2006.

[RFC4703] Stapp, M. and B. Volz, "Resolution of Fully Qualified Domain Name (FQDN) Conflicts among Dynamic Host Configuration Protocol (DHCP) Clients", RFC 4703,

October 2006.

[RFC5007] Brzozowski, J., Kinnear, K., Volz, B., and S. Zeng, "DHCPv6 Leasequery", RFC 5007, September 2007.

[RFC5460] Stapp, M., "DHCPv6 Bulk Leasequery", RFC 5460, February 2009.

Authors' Addresses

John Jason Brzozowski
Comcast Cable Communications
1306 Goshen Parkway
West Chester, PA 19380
USA

Phone: +1-609-377-6594
Email: john_brzozowski@cable.comcast.com

Jean-Francois Tremblay
Videotron Ltd.
612 Saint-Jacques
Montreal, Quebec H3C 4M8i
Canada

Email: Jean-Francois.TremblayING@videotron.com

Jack Chen
Time Warner Cable
13820 Sunrise Valley Drive
Herndon, VA 20171
USA

Email: jack.chen@twcable.com

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

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

dhc Working Group
Internet Draft
Intended Status: Standards Track
Expires: April 22, 2011

Kim Kinnear
Bernie Volz
Mark Stapp
Cisco Systems

Neil Russell
Nokia

October 22, 2010

Active DHCPv4 Lease Query
<draft-kinnear-dhc-dhcpv4-active-leasequery-01.txt>

Status of this Memo

Copyright Notice

Abstract

The Dynamic Host Configuration Protocol for IPv4 (DHCPv4) has been extended with a Leasequery capability that allows a client to request information about DHCPv4 bindings. That mechanism is limited to queries for individual bindings. In some situations individual binding queries may not be efficient, or even possible. In addition, continuous update of an external client with Leasequery data is sometimes desired. This document expands on the DHCPv4 Leasequery protocol, and allows for active transfer of real-time DHCPv4 address binding information data via TCP.

Table of Contents

1. Introduction

The DHCPv4 Leasequery capability [RFC4388] extends the basic DHCPv4 capability [RFC2131] [RFC2132] to allow an external entity to query a DHCPv4 server to recover lease state information about a particular IP address or client in near real-time.

Requirements exist for external entities to keep up to date on the correspondence between DHCPv4 clients and the IPv4 addresses for which they have leases. These requirements often stem from regulatory requirements placed on service providers by governmental agencies.

These entities need to keep up with the current IPv4 address binding activity of the DHCPv4 server. Keeping up with address binding activity is termed "active" leasequery.

The DHCPv4 Bulk Leasequery [DHCPv4Bulk] capability can be used to recover useful information from a DHCPv4 server when some external entity starts up. This entity could be one which is directly involved in the DHCPv4 client - server transactions (e.g., a relay agent), or it could be an external process which needs information present in the DHCPv4 server’s lease state database.

The Active Leasequery capability documented here is designed to allow an entity not directly involved in DHCPv4 client - server transactions to nevertheless keep current with the state of the DHCPv4 lease state information in real-time.

2. Terminology

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

This document uses the following terms:

- "address binding"

  The information that a DHCPv4 server keeps regarding the relationship between a DHCPv4 client and an IPv4 IP address. This includes the identity of the DHCPv4 client and the expiration time, if any, of any lease that client has on a particular IPv4 address.

- "Active Leasequery"

  Keeping up to date in real-time (or near real-time) with DHCPv4 address binding activity.

- "Bulk Leasequery"

  Requesting and receiving the existing DHCPv4 address binding information in an efficient manner.

- "catch-up information, catch-up phase"

  If a DHCPv4 Active Leasequery requestor sends in a query-start-time option in a DHCPACTIVELEASEQUERY message, the DHCPv4 server will attempt to send the requestor the information that changed since the time specified in the query-start-time option. The address binding information sent to satisfy this request is the catch-up information, and the period while it is being sent is the catch-up phase.

- "clock skew"

  The difference between the absolute time on a DHCPv4 server and the absolute time on the system where a requestor of an Active or Bulk Leasequery is executing is termed the "clock skew" for that Active or Bulk Leasequery connection. It is not absolutely constant but is likely to vary only slowly. While it is easy to think that this can be calculated precisely after one packet is received by a requestor from a DHCPv4 server, a more accurate value is derived from continuously examining the instantaneous value developed from each packet received from a DHCPv4 server and using it to make small adjustments to the existing value held in the requestor.

- "DHCPv4 client"

  A DHCPv4 client is an Internet host using DHCP to obtain configuration parameters such as a network address.

- "DHCPv4 relay agent"

  A DHCPv4 relay agent is a third-party agent that transfers BOOTP and DHCPv4 messages between clients and servers residing on different subnets, per [RFC951] and [RFC1542].

- "DHCPv4 server"

  A DHCPv4 server is an Internet host that returns configuration parameters to DHCPv4 clients.

- "IP address binding"

  The information that a DHCPv4 server keeps regarding the relationship between a DHCPv4 client and an IPv4 IP address. This includes the identity of the DHCPv4 client and the expiration time, if any, of any lease that client has on a particular IPv4 address.

- "MAC address"

  In the context of a DHCP message, a MAC address consists of the fields: hardware type "htype", hardware length "hlen", and client hardware address "chaddr".

3. Protocol Overview

The Active Leasequery mechanism is modeled on the existing individual Leasequery protocol in [RFC4388] as well as related work on DHCPv4 Bulk Leasequery [DHCPv4Bulk]; most differences arise from the long term nature of the TCP connection required for Active Leasequery. In addition, a DHCPv4 server which supports Active Leasequery MUST support Bulk Leasequery [DHCPv4Bulk] as well.

An Active Leasequery client opens a TCP connection to a DHCPv4 Server, using the DHCPv4 port 67. Note that this implies that the Leasequery client has server IP address(es) available via configuration or some other means, and that it has unicast IP reachability to the DHCPv4 server. No relaying for Active Leasequery is specified.

After establishing a connection, the client sends an DHCPACTIVELEASEQUERY message over the connection. In response, the server sends updates to the requestor using DHCPLEASEACTIVE and DHCPLEASEUNASSIGNED messages which are extensions of these messages as defined in [RFC4388] and [DHCPv4Bulk].

Active Leasequery is designed to provide continuous updates of DHCPv4 IPv4 address binding activity to an external entity.

Active Leasequery has features which allow this external entity to lose its connection and then reconnect and receive the latest information concerning any IP addresses changed while it was not connected.

These capabilities are designed to allow the Active Leasequery requestor to efficiently become current with respect to the lease state database after it has been restarted or the machine on which it is running has been reinitialized. It is easy to define a protocol which works when the requestor is always connected to the DHCPv4 server. Since that isn’t sufficiently robust, much of the mechanism in this document is designed to deal efficently with situations that occur when the Active Leasequery requestor becomes disconnected from the DHCPv4 server from which it is receiving updates and then becomes reconnected to that server.

Central to this approach, if the Active Leasequery requestor loses service, it is allowed to specify the time of its most recent update in a subsequent Active Leasequery request and the DHCPv4 server will determine whether or not data was missed while the Active Leasequery requestor was not connected.

The DHCP server processing the Active Leasequery request may limit the amount of data saved, and methods exist for the DHCPv4 server to inform the Active Leasequery requestor that more data was missed than could be saved. In this situation, the Active Leasequery requestor would issue a Bulk Leasequery [DHCPv4Bulk] to recover information not available through an Active Leasequery.

DHCPv4 servers are not required to keep any data corresponding to data missed on a Active Leasequery connection, but will typically choose to keep data corresponding to some recent activity available for subsequent queries by a DHCPv4 Active Leasequery client whose connection was temporarily interrupted.

An Active Leasequery requestor would typically use Bulk Leasequery to initialize its database with all current data when that database contains no address binding information. In addition, it would use Bulk Leasequery to recover missed information in the event that its connection with the DHCPv4 server was lost for a longer time than the DHCPv4 server would keep track of the specific changes to the IP address binding information.

The messages sent by the server in response to an Active Leasequery request SHOULD be identical to the messages sent by the server to a Bulk Leasequery request regarding the way the data is encoded into the Active Leasequery responses. In addition, the actions taken by the Active Leasequery requestor to interpret the responses to an Active Leasequery request SHOULD be identical to the way that the requestor interprets the responses to a Bulk Leasequery request. Thus, the handling of time, clock skew, data source, and other items discussed in the Bulk Leasequery specification [DHCPv4Bulk] are to be followed when implementing Active Leasequery.

4. Interaction Between Active Leasequery and Bulk Leasequery

Active Leasequery can be seen as an extension of the Bulk Leasequery protocol [DHCPv4Bulk]. The contents of packets returned to an Active Leasequery requestor are identical to that defined for the Bulk Leasequery protocol [DHCPv4Bulk].

Applications which employ Active Leasequery to keep a database up to date with respect to the DHCPv4 server’s lease state database will

usually use an initial Bulk Leasequery to bring their database into equivalence with that of the DHCPv4 server, and then use Active Leasequery to keep that database current with respect to the DHCPv4 server's lease state database.

There are several differences between the Active and Bulk Leasequery protocols. Active Leasequery defines only one qualifier (the query-start-time) and no query types, while Bulk Leasequery defines several of query types and qualifiers. An Active Leasequery connection sends all available updates to the requestor.

An Active Leasequery connection does not ever "complete", though the DHCPv4 server may drop the connection for a variety of reasons associated with some sort of exception condition.

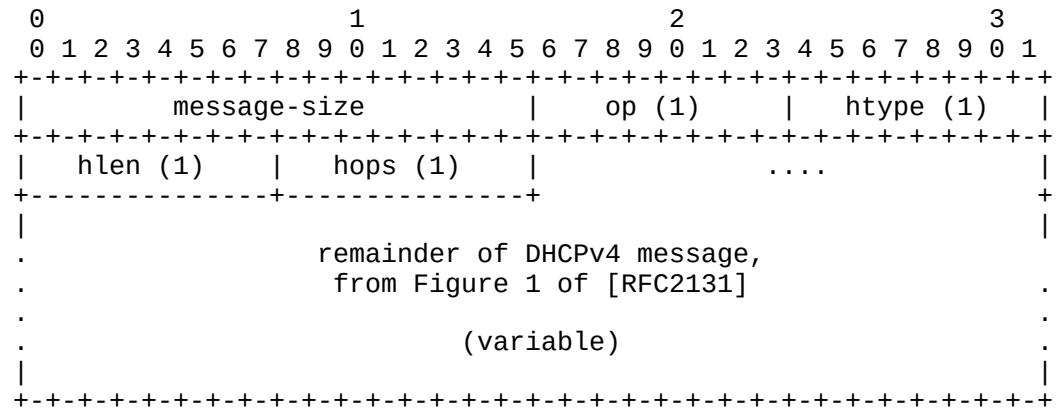
5. Message and Option Definitions

5.1. Message Framing for TCP

The use of TCP for the Active Leasequery protocol permits one or more DHCPv4 messages to be sent at a time. The receiver needs to be able to determine how large each message is. The same framing techique used for Bulk Leasequery [DHCPv4Bulk] is used for Active Leasequery.

Two octets containing the message size in network byte-order are prepended to each DHCPv4 message sent on an Active Leasequery TCP connection. The two message-size octets 'frame' each DHCPv4 message.

DHCPv4 message framed for TCP:



message-size the number of octets in the message that follows, as a 16-bit integer in network byte-order.

All other fields are as specified in DHCPv4 [RFC2131].

Figure 1: Format of a DHCPv4 message in TCP

The intent in using this format is that code which currently knows how to deal with a message returned from DHCPv4 Leasequery [RFC4388] will be able to deal with the message held inside of the TCP framing.

5.2. New or Changed Options

The existing messages DHCPLEASEUNASSIGNED and DHCPLEASEACTIVE are used as the value of the dhcp-message-type option to indicate an IP address which is currently not leased or currently leased to a DHCPv4 client, respectively.

All of the message types and options defined for Bulk Leasequery [DHCPv4Bulk] are also used by Active Leasequery. In addition, new message types and option types are defined for Active Leasequery, as described below.

5.2.1. dhcp-message-type

The message type option (option 53) from [RFC2132] requires additional values. The values of these message types are shown below

in an extension of the table from Section 9.6 of [RFC2132]:

Value	Message Type
16	DHCPACTIVELEASEQUERY
17	DHCPLEASEQUERYSTATUS

5.2.2. dhcp-status-code

The dhcp-status-code option defined in [DHCPv4Bulk] allows greater detail to be returned regarding the status of a DHCP request. While specified in the Bulk Leasequery document, this DHCPv4 option is also used in Active Leasequery.

This option has two possible scopes when used with Active Leasequery, depending on the context in which it appears. It refers to the information in a single leasequery reply if the value of the dhcp-message-type is DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED. It refers to the message stream related to an entire request if the value of the dhcp-message-type is DHCPLEASEQUERYSTATUS.

Additional status codes defined for support of Active Leasequery are:

Name	status-code	Description
DataMissing	005	Indicates that IP address binding information requested is not available.
ConnectionActive	006	Indicates that this connection remains active.
CatchUpComplete	007	Indicates that this Active Leasequery connection has completed sending all of the saved data requested.

A dhcp-status-code option MAY appear in the options field of a DHCP message. If the dhcp-status-code option does not appear, it is assumed that the operation was successful. The dhcp-status-code option SHOULD NOT appear in a message which is successful unless it is needed to convey some text message along with the Success status code.

5.3. Connection and Transmission Parameters

DHCPv4 servers that support Active Leasequery SHOULD listen for incoming TCP connections on the DHCPv4 server port 67. Implementations MAY offer to make the incoming port configurable, but port 67 MUST be the default. Requestors SHOULD make TCP connections to port 67, and MAY offer to make the destination server port configurable.

This section presents a table of values used to control Active Leasequery behavior, including recommended defaults. Implementations MAY make these values configurable. However, configuring too-small timeout values may lead to harmful behavior both to this application as well as to other traffic in the network. As a result, timeout values smaller than the default values are NOT RECOMMENDED.

Parameter	Default	Description
BULK_LQ_DATA_TIMEOUT	300 secs	Bulk Leasequery data timeout
BULK_LQ_MAX_CONNS	10	Max Bulk Leasequery TCP connections
ACTIVE_LQ_RCV_TIMEOUT	120 secs	Active Leasequery receive timeout
ACTIVE_LQ_SEND_TIMEOUT	120 secs	Active Leasequery send timeout
ACTIVE_LQ_IDLE_TIMEOUT	60 secs	Active Leasequery idle timeout

6. Information Communicated by Active Leasequery

While the information communicated by a Bulk Leasequery [DHCPv4Bulk] is taken directly from the DHCPv4 server's lease state database, the information communicated by an Active Leasequery is real-time information. As such, it is the information which is currently associated with a particular IP address in the DHCPv4 server's lease state database.

This is of significance, because if the Active Leasequery requestor runs slowly or the requestor disconnects from the DHCPv4 server and then reconnects with a query-start-time (signalling a catch-up operation), the information communicated to the Active Leasequery requestor is only the most current information from the DHCPv4 server's lease state database.

The requestor of an Active Leasequery MUST NOT assume that every lease state change is communicated across an Active Leasequery connection. Even if the Active Leasequery requestor remains connected, the DHCPv4 server is only required to transmit information

about an IP address that is current when the packet is created and handed off to the TCP stack to send to the requestor.

If the TCP connection blocks and the DHCPv4 server is waiting to send information down the connection, when the connection becomes available to be written the DHCPv4 server MAY create the packet to send at this time. The current state of the IP address will be sent, and any transition in state or other information that occurred while the TCP connection was blocked will be lost.

Thus, the Active Leasequery protocol does not allow the requestor to build a complete history of every activity on every lease. An effective history of the important state changes for a lease can be created if the parameters of the DHCPv4 server are tuned to take into account the requirements of an Active Leasequery requestor. For instance, the period after the expiration or release of an IP address could be configured long enough (say several minutes, well more than the receive timeout), so that an Active Leasequery requestor would never miss any changes in the client to IP address binding.

7. Requestor Behavior

7.1. Connecting and General Processing

A Requestor attempts to establish a TCP connection to a DHCPv4 Server in order to initiate a Leasequery exchange. If the attempt fails, the Requestor MAY retry.

If an Active Leasequery is terminated prematurely by a DHCPLEASEQUERYDONE with a dhcp-message status-code of QueryTerminated or by the failure of the connection over which it was being submitted, the requestor MAY retry the request after the creation of a new connection.

Messages from the DHCPv4 server come as multiple responses to a single DHCPACTIVELEASEQUERY message. Thus, each DHCPACTIVELEASEQUERY or DHCPBULKLEASEQUERY request MUST have an xid (transaction-id) unique on the connection on which it is sent, and all of the messages which come as a response to it all contain the same xid as the request. It is the xid which allows the data-streams of two or more different DHCPACTIVELEASEQUERY or DHCPBULKLEASEQUERY requests to be demultiplexed by the requestor.

A requestor MAY send a DHCPACTIVELEASEQUERY request to a DHCPv4 server and immediately close the transmission side of its TCP connection, and then read the resulting response messages from the DHCPv4 server. This is not required, and the usual approach is to

leave both sides of the TCP connection up until at least the conclusion of the Active Leasequery.

7.2. Forming an Active Leasequery

The Active Leasequery is designed to create a long lived connection between the requestor and the DHCPv4 server processing the active query. The DHCPv4 server will send IPv4 address binding information back across this connection with minimal delay after it learns of the binding information. It will learn about IPv4 address bindings either because it makes the bindings itself or because it has received information about a binding from another server.

To form the Active query, a DHCPv4 request is constructed with a dhcp-message-type of DHCPACTIVELEASEQUERY. This DHCPv4 request MUST NOT have a ciaddr, a chaddr, or a dhcp-client-identifier. The DHCPv4 request MUST contain a transaction-id, and that transaction-id MUST BE locally unique to the TCP connection to the DHCPv4 server. The DHCPv4 request SHOULD have a dhcp-parameter-request-list to inform the DHCPv4 server which DHCPv4 options are of interest to the requestor sending the DHCPACTIVELEASEQUERY message.

An important capability of the Active Leasequery is the ability of the requestor to specify that some recent data be sent immediately to the requestor in parallel with the transmission of the ongoing IPv4 address binding information in more or less real time. This capability is used in order to allow an Active Leasequery requestor to recover missed information in the event that it temporarily loses connectivity with the DHCPv4 server processing a previous Active Leasequery.

Note that until all of the recent data (catch-up data) has been received, the requestor MUST NOT keep track of the base time received in Leasequery reply messages to use later in a subsequent Bulk Leasequery or Active Leasequery request.

This capability is enabled by the transmission of a 4 octet base-time option with each Leasequery reply sent as the result of a previous Active Leasequery. The requestor will typically keep track of the highest base-time received from a particular DHCPv4 server over an Active Leasequery connection, and in the event that the requestor finds it necessary (for whatever reason) to reestablish an Active Leasequery connection to that DHCPv4 server, the requestor will place this highest base-time value into a query-start-time option in the new DHCPACTIVELEASEQUERY request.

If the requestor doesn’t wish to request an update of information missed when it was not connected to the DHCPv4 server, then it does

not include the query-start-time option in the DHCPACTIVELEASEQUERY request.

If the TCP connection becomes blocked or stops being writable while the requestor is sending its query, the requestor SHOULD be prepared to terminate the connection after BULK_LQ_DATA_TIMEOUT. We make this recommendation to allow requestors to control the period of time they are willing to wait before abandoning a connection, independent of notifications from the TCP implementations they may be using.

7.3. Processing Active Replies

The Requestor attempts to read a DHCPv4 leasequery reply message from the TCP connection. If the stream of replies becomes blocked, the Requestor SHOULD be prepared to terminate the connection after ACTIVE_LQ_RCV_TIMEOUT, and MAY begin retry processing if configured to do so.

Note that a DHCPACTIVELEASEQUERY request specifically requests the DHCPv4 server to create a long-lived connection which may not have data transferring continuously during its lifetime. Therefore the DHCPv4 server will send a DHCPLEASEQUERYSTATUS message with a dhcp-status-code of ConnectionActive every ACTIVE_LQ_IDLE_TIMEOUT seconds (default 60) in order for the requestor to know that the connection remains alive. Note that the default for ACTIVE_LQ_RCV_TIMEOUT is 120 seconds, twice the value of the ACTIVE_LQ_IDLE_TIMEOUT's default of 60 seconds which drives the DHCPv4 server to send messages. Thus ACTIVE_LQ_RCV_TIMEOUT controls how sensitive the requestor is to be to delays by the DHCPv4 server in sending updates or DHCPLEASEQUERYSTATUS messages.

A successful query that is returning binding data MUST include a non-zero ciaddr. It may also include a non-zero chaddr, htype, and hlen as well as additional options. If there are additional bindings to be returned, they will be carried in additional Active Leasequery messages.

Any requestor of an Active Leasequery operation MUST be prepared to receive multiple copies of the IPv4 address binding information for a particular IPv4 address. See the Bulk Leasequery draft [DHCPv4Bulk] for information on how to deal with this situation.

A single Active Leasequery can and usually will result in a large number of replies. The Requestor MUST be prepared to receive more than one reply with transaction-ids matching a single DHCPACTIVELEASEQUERY message from a single DHCPv4 server.

A DHCPACTIVELEASEQUERY has two regimes -- during the catch-up phase,

if any, and after any catch-up phase. During the catch-up phase (if one exists), the data returned in the base-time option in a DHCPLEASEACTIVE or DHCPLEASEUNASSIGNED message may appear to be ordered, but the most recent change in the lease state data being returned is not related to the base-time option value in the messages. Another way to say this is that the ordering of the updates sent by the DHCPv4 server during the catch-up phase is independent of the ordering in the changes in the lease state data. The base-time option from messages during this phase MUST NOT be saved and used in a subsequent DHCPACTIVELEASEQUERY message's query-start-time option as it does not represent the extent of progress of the catch-up activity.

After the catch-up phase, or during the entire series of messages received as the response to a DHCPACTIVELEASEQUERY request with no query-start-time (and therefore no catch-up phase), the base-time option of the most recent message SHOULD be saved as a record of the most recent time that data was received. This base-time (in the context of the DHCPv4 server) can be used in a subsequent DHCPACTIVELEASEQUERY message's query-start-time, or in a DHCPBULKLEASEQUERY message's query-start-time if one is required, after a loss of the Active Leasequery connection.

The DHCPLEASEQUERYSTATUS message MAY unilaterally terminate a successful DHCPACTIVELEASEQUERY request which is currently in progress in the event that the DHCPv4 server determines that it cannot continue processing a DHCPACTIVELEASEQUERY request. For example, when a server is requested to shut down it SHOULD send a DHCPLEASEQUERYSTATUS message with a dhcp-status-code of QueryTerminated and include in the message a base time. This SHOULD be the last message on that connection, and once the message has been transmistted, the server should close the connection.

After receiving DHCPLEASEQUERYSTATUS with a QueryTerminated status from a server, the Requestor MAY close the TCP connection to that server.

The DHCPv4 Leasequery protocol uses the associated-ip option as an indicator that multiple bindings were present in response to a single client based query. For Active Leasequery, client-based queries are not supported and so the associated-ip option is not used, and MUST NOT be present in replies.

7.3.1. Processing Replies from a Request Containing a query-start-time

If the DHCPACTIVELEASEQUERY was requested with a query-start-time, the DHCPv4 server will attempt to send information about all IP

address bindings that changed since the time specified in the query-start-time. This is the catch-up phase of the DHCPACTIVELEASEQUERY processing. The DHCPv4 server MAY also begin immediate updates over the same connection of real-time IP address binding information changes. Thus, the catch-up phase may run in parallel with the normal updates generated by the DHCPACTIVELEASEQUERY request.

A DHCPv4 server MAY keep only a limited amount of time ordered information available to respond to a DHCPACTIVELEASEQUERY request containing a query-start-time. Thus, it is possible that the time specified in the query-start-time represents a time not covered by the time ordered information kept by the DHCPv4 server. If this should occur, and there is not enough data saved in the DHCPv4 server to satisfy the request specified by the query-start-time option, the DHCPv4 server will reply immediately with a DHCPLEASEQUERYSTATUS message with a dhcp-status-code of DataMissing with a base-time option equal to the server's current time. This will signal the end of the catch-up phase, and the only updates that will subsequently be received on this connection are the real-time updates from the DHCPACTIVELEASEQUERY request.

If there is enough data saved to satisfy the request, then DHCPLEASEACTIVE and DHCPLEASEUNASSIGNED messages will begin arrive from the DHCPv4 server. Some of these messages will be related to the query-start-time request and be part of the catch-up phase. Some of these messages will be real-time updates of IP address binding changes taking place in the DHCPv4 server. In general, there is no way to determine the source each message.

Until the catch-up phase is complete, the latest base-time value received from a DHCPv4 server processing an Active Leasequery request cannot be reset from the incoming messages because to do so would compromise the ability to recover lost information if the DHCPACTIVELEASEQUERY were to terminate prior to the completion of the catch-up phase.

The requestor will know that the catch-up phase is complete because the DHCPv4 server will transmit a DHCPLEASEQUERYSTATUS message with the dhcp-status-code of CatchUpComplete. Once this message is transmitted, all additional DHCPLEASEACTIVE and DHCPLEASEUNASSIGNED messages will relate to real-time ("new") IP address binding changes in the DHCPv4 server.

As discussed in Section 6.3, the requestor SHOULD keep track of the latest base-time option value received over a particular connection, to be used in a subsequent DHCPACTIVELEASEQUERY request -- but only if the catch-up phase is complete. Prior to the completion of the catch-up phase, if the connection should go away or if the requestor

receives a DHCPLEASEQUERYDONE message, then when it reconnects it MUST use the base-time value from the previous connection and not any base-time value received from the recently closed connection.

In the event that there was enough data available to the DHCPv4 server to begin to satisfy the request implied by the query-start-time option, but during the processing of that data the server found that it was unable to continue (perhaps there was barely enough, the connection is very slow, and the aging algorithm causes the saved data to become unavailable) the DHCPv4 server will terminate the catch-up phase of processing immediately by sending a DHCPLEASEQUERYSTATUS message with a dhcp-status-code of DataMissing and with a base-time option of the current time.

The requestor MUST NOT assume that every individual state change of every IP address during the period from the time specified in the query-start-time and the present is replicated in an Active Leasequery reply message. The requestor MAY assume that at least one Active Leasequery reply message will exist for every IP address which had one or more changes of state during the period specified by the query-start-time and the current time. The last message for each IP address will contain the state at the current time, and there may be one or more messages concerning a single IP address during the catch-up phase of processing.

If an IP address changed state multiple times during the time that the requestor was not connected (that is, during the time from the query-start-time and the present), then only the current IP address binding information will be sent during the catch-up phase. However, the requestor MUST NOT assume that every intermediate state change that occurred during the period from the query-start-time to the present will be represented by an individual Leasequery message.

If the DHCPLEASEQUERYSTATUS message containing a dhcp-status-code of DataMissing is received and the requestor is interested in keeping its database up to date with respect to the current state of IP address bindings in the DHCPv4 server, then the requestor SHOULD issue a DHCPBULKLEASEQUERY request to recover the information missing from its database. This DHCPBULKLEASEQUERY should include a query-start-time, set to be the same as its query-start-time previously included in the DHCPACTIVELEASEQUERY responses from the DHCPv4 server, and a query-end-time equal to the base-time returned by the DHCPv4 server in the DHCPLEASEQUERYSTATUS message with the dhcp-status-code of DataMissing.

In the event that the requestor receives a DHCPLEASEQUERYSTATUS message with a dhcp-status-code of DataMissing, it is a reasonable assumption that it is interested in keeping its database up to date

with respect to the DHCPv4 server's internal IP address binding database or it would not have included the query-start-time in the DHCPACTIVELEASEQUERY message.

Typically, the requestor would have one connection open to a DHCPv4 server for a DHCPACTIVELEASEQUERY request and possibly one additional connection open for a DHCPBULKLEASEQUERY request to the same DHCPv4 server to fill in the data that might have been missed prior to the initiation of the DHCPACTIVELEASEQUERY. The Bulk Leasequery connection would typically run to completion and be closed, leaving one Active Leasequery connection open to a single DHCPv4 server. Alternatively, both requests could be issued over a single connection.

7.4. Closing Connections

The Requestor or DHCPv4 leasequery server MAY close its end of the TCP connection at any time. The Requestor MAY choose to retain the connection if it intends to issue additional queries. Note that this client behavior does not guarantee that the connection will be available for additional queries: the server might decide to close the connection based on its own configuration.

8. Server Behavior

A DHCPv4 server which supports Active Leasequery MUST support Bulk Leasequery [DHCPv4Bulk] as well.

8.1. Accepting Connections

Servers that implement DHCPv4 Active Leasequery listen for incoming TCP connections. Port numbers are discussed in Section 5.3. Servers MUST be able to limit the number of currently accepted and active connections. The value BULK_LQ_MAX_CONNS MUST be the default; implementations MAY permit the value to be configurable. Connections SHOULD be accepted and, if the number of connections is over BULK_LQ_MAX_CONNS, they SHOULD be closed immediately.

Servers MAY restrict Active Leasequery connections and DHCPACTIVELEASEQUERY messages to certain clients. Connections not from permitted clients SHOULD be closed immediately, to avoid server connection resource exhaustion.

If the TCP connection becomes blocked while the server is accepting a connection or reading a query, it SHOULD be prepared to terminate the connection after an BULK_LQ_DATA_TIMEOUT. We make this recommendation to allow servers to control the period of time they are willing to wait before abandoning an inactive connection,

independent of the TCP implementations they may be using.

8.2. Replying to an Active Leasequery

If the connection becomes blocked while the server is attempting to send reply messages, the server SHOULD be prepared to terminate the TCP connection after ACTIVE_LQ_SEND_TIMEOUT. This timeout governs how much congestion the DHCPv4 server is prepared to tolerate over any Active Leasequery connection. The default is two minutes, which means that if more than two minutes goes by without the requestor reading enough information to unblock the TCP connection, the DHCPv4 server will drop the TCP connection.

If the DHCPv4 server encounters an error during processing of the DHCPACTIVELEASEQUERY message, either during initial processing or later during the message processing, it SHOULD send a DHCPLEASEQUERYSTATUS containing an error code of some kind in a dhcp-status-code option. It SHOULD close the connection after this error is signalled.

Every reply to a DHCPACTIVELEASEQUERY request MUST contain the information specified in replies to a DHCPBULKLEASEQUERY request [DHCPv4Bulk].

If a DHCPACTIVELEASEQUERY request contains a query-start-time option, it indicates that the requestor would like the DHCPv4 server to send it not only messages that correspond to DHCPv4 address binding activity that occurs subsequent to the receipt of the DHCPLEASEACTIVE request, but also messages that correspond to DHCPv4 address binding activity that occurred prior to the DHCPACTIVELEASEQUERY request.

If a query-end-time option appears in a DHCPACTIVELEASEQUERY the DHCPv4 server should send a DHCPLEASEQUERYSTATUS message with a dhcp-status-code of MalformedQuery and terminate the connection.

In order to implement a meaningful response to this query, the DHCPv4 server MAY keep track of the address binding activity and associate changes with particular base-time values from the messages. Then, when requested to do so by a DHCPACTIVELEASEQUERY request containing a query-start-time option, the DHCPv4 server can respond with replies for all address binding activity occurring on that query-start-time or later times.

These replies based on the query-start-time MAY be interleaved with the messages generated due to current IP address binding activity.

Once the transmission of the DHCPv4 Leasequery messages associated

with the query-start-time option are complete, a DHCPLEASEQUERYSTATUS message MUST be sent with a dhcp-status-code value of CatchUpComplete.

The DHCPv4 server SHOULD but is not required to keep track of a limited amount of previous address binding activity and associate it with base-time values. The DHCPv4 server MAY choose to only do this in the event that it has received at least one DHCPACTIVELEASEQUERY request in the past, as to do so will almost certainly entail some utilization of resources which would be wasted if there are no DHCPACTIVELEASEQUERY clients for this DHCPv4 server. The DHCPv4 server SHOULD make the amount of previous address binding activity it retains configurable. There is no requirement on the DHCPv4 server to retain this information over a server restart (or even to retain such information at all).

Unless there is an error or some requirement to cease processing a DHCPACTIVELEASEQUERY request yielding a DHCPLEASEQUERYSTATUS message, such as a server shutdown, there will be no DHCPLEASEQUERYSTATUS message at the conclusion of the DHCPACTIVELEASEQUERY processing because that processing will not conclude but will continue until either the client or the server drops the connection.

8.3. Multiple or Parallel Queries

Requestors may want to use an existing connection if they need to make multiple queries. Servers MAY support reading and processing multiple queries from a single connection. A server MUST NOT read more query messages from a connection than it is prepared to process simultaneously.

Typically, a requestor of a Active Leasequery would not need to send a second Active Leasequery while the first is still active. However, sending an Active Leasequery and a Bulk Leasequery over the same connection would be possible and reasonable.

This MAY be a feature that is administratively controlled. Servers that are able to process queries in parallel SHOULD offer configuration that limits the number of simultaneous queries permitted from any one requestor, in order to control resource use if there are multiple requestors seeking service.

8.4. Closing Connections

The server MAY close its end of the TCP connection after sending its last message, a DHCPLEASEQUERYSTATUS message in response to a query. Alternatively, the server MAY retain the connection and wait for additional queries from the client. The server SHOULD be prepared to

limit the number of connections it maintains, and SHOULD be prepared to close idle connections to enforce the limit.

The server MUST close its end of the TCP connection if it encounters an error sending data on the connection. The server MUST close its end of the TCP connection if it finds that it has to abort an in-process request. A server aborting an in-process request SHOULD attempt to signal that to its clients by using the QueryTerminated status code in the dhcp-status-code option in a DHCPLEASEQUERYSTATUS message. If the server detects that the client end has been closed, the server MUST close its end of the connection after it has finished processing any outstanding requests.

9. Security Considerations

The "Security Considerations" section of [RFC2131] details the general threats to DHCPv4. The DHCPv4 Leasequery specification [RFC4388] describes recommendations for the Leasequery protocol, especially with regard to relayed LEASEQUERY messages, mitigation of packet-flooding DOS attacks, restriction to trusted clients, and use of IPsec [RFC4301].

The use of TCP introduces some additional concerns. Attacks that attempt to exhaust the DHCPv4 server's available TCP connection resources, such as SYN flooding attacks, can compromise the ability of legitimate clients to receive service. Malicious clients who succeed in establishing connections, but who then send invalid queries, partial queries, or no queries at all also can exhaust a server's pool of available connections. We recommend that servers offer configuration to limit the sources of incoming connections, that they limit the number of accepted connections and the number of in-process queries from any one connection, and that they limit the period of time during which an idle connection will be left open.

10. IANA Considerations

IANA is requested to assign the following new values for this document. See Section 5.2 for details.

1. A dhcp-message-type of 16 for DHCPACTIVELEASEQUERY.

2. A dhcp-message-type of 17 for DHCPLEASEQUERYSTATUS.

3. Values for dhcp-status-code:

Name	status-code
DataMissing	005
ConnectionActive	006
CatchUpComplete	007

11. Acknowledgements

The ideas in this document came in part from work in DHCPv6 and DHCPv4 Bulk Leasequery as well as from in depth discussions between the authors.

12. References

12.1. Normative References

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC4301] Kent, S., Seo, K., "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4388] Woundy, R., Kinnear, K., "Dynamic Host Configuration Protocol (DHCP) Leasequery", RFC 4388, February 2006.

12.2. Informative References

[RFC951] Croft, B., Gilmore, J., "Bootstrap Protocol (BOOTP)", RFC 951, September 1985.

[RFC1542] Wimer, W., "Clarifications and Extensions for the Bootstrap Protocol", RFC 1542, October 1993.

[RFC2132] Alexander, S., Droms, R., "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.

[DHCPv4Bulk] Kinnear, K., Volz, B., Russell, N., Stapp, M., Rao, D.,Joshi B., Kurapati, P., "DHCPv4 Bulk Leasequery", draft-ietf-

dhc-dhcpv4-bulk-leasequery-03.txt, October 2010.

Authors' Addresses

Kim Kinnear
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: kkinnear@cisco.com

Bernie Volz
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: volz@cisco.com

Neil Russell
Nokia
5 Wayside Drive
Burlington, MA 01803

EMail: neil.russell@nokia.com

Mark Stapp
Cisco Systems
1414 Massachusetts Ave.
Boxborough, Massachusetts 01719

Phone: (978) 936-0000

EMail: mjs@cisco.com

dhc
Internet-Draft
Intended status: Informational
Expires: April 16, 2011

Ma
CNNIC

October 13, 2010

Extended DHCPv6 for Piggybacking Security Association Configuration
draft-madi-dhc-dhcpv6-psac-00

Abstract

IPSec [RFC2401]is pervasive in many scenarios to build the channel of security mechanism to protect the communication between the host and the local servers, such as DNS recursive name server [RFC1304]. In the public wireless access environment, an extra trust relationship configuration between the roaming host and the local server, manually or by IKE [RFC2409], is indispensible.

DHCP is typically the first protocol executed by a mobile host when it enters a new network, so this document present an extension to DHCPv6 to piggyback the parameters needed for IPSec, avoiding the delay invited by manual configuration of security association or IKE interaction for IPSec.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 16, 2011.

Copyright Notice

Table of Contents

1. Introduction

There are an increasing number of mobile hosts who, roaming across different access domains, want to resort to the shared key based method to gain secrecy and integrity of the data exchange with the local server of the access network, the communication between the DNS stub resolver and the local DNS recursive name sever as an example. IPSec, for instance, is pervasive in many scenarios to build the channel of security mechanism. And IPSec architecture has a security policy database that specifies which traffic is protected and how, which could be established with a manual configuration of security associations or with IKE [RFC5996]. As for the mobile host, its security association with the local server is usually established by IKE, which leads to an extra interaction delay. Even the delay MAY not be taken into consideration, IKE is not necessarily not be supported everywhere.

DHCP allows a computer to be configured automatically, eliminating the need for intervention by a network administrator. For IPv4, DHCPv4 is typically the first protocol executed by a mobile host when it enters a new network. Even in the era of IPv6, according to the point of view of Ralph, DHCP service is typically provided by a centralized service composed of fewer managed components [21], so DHCP server misconfiguration is less likely than delivery of misconfigured Route Advertisements. Since DHCP also takes the responsibility in configuring the IP address of the local server, the security association information between the host and the specific local server, such as SIP server and DNS recursive name server, could be piggybacked via DHCP messages, avoiding the delay invited by manual configuration of security association or IKE interaction for IPSec.

This document describes the extension to DHCP to integrate security association construction course into the DHCP interaction to build the trust relationship between the roaming host and the specific local server of the access network.

2. Terminology

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

DHCPv6 terminology is defined in [RFC3315]. Terminology specific to the extension of DHCPv6 can be found below:

Throughout this document, "DHCP" refers to DHCP for IPv6. DHCP terminology is defined in [RFC3315] and IPSec terminology is defined

in [RFC2401]. Terminology specific to the extension of DHCP can be found below:

SA - SA is short for Security Association and defined in IPSec specification [RFC2401].

SPI - SPI is short for Security Parameter Index and defined in IPSec specification [RFC2401].

target server - A local server of the access network, DNS recursive name server for instance, which is to be configured the security association with a specific host, by DHCP. The target server works as a DHCP client listening for DHCP messages on UDP port 546.

requestor - A host that wants to establish security association with a specific local server. The requestor works as a DHCP client requesting the configuration parameters for security association.

SA binding - SA binding is employed by DHCP server to manage the SA information which is indexed by the tuple [requestor's DUID, target server's DUID, SPI].

3. Requirements and Considerations

Although DHCP is being challenged by the SLAAC [RFC4862], yet this document is intended to take the position that DHCP is indispensible and necessary to the network administrative need.

DHCP server and the target server are usually deployed within one same organization and public key schemes are not necessary, trust relationship based on preshared secret could be established between them by administrator's manual configuration to gain secrecy and integrity. Accordingly, the extension in this document takes the position that DHCP service is indispensible and there is secured channel between DHCP server and the target server.

To realize the very function of the extended DHCP, the host in question MUST have the ability to generate asymmetric key pair, the public key of which is used to encrypt the symmetric key to be shared with the target server.

The extension of DHCPv6 SHOULD go with stateful service DHCPv6 provides.

4. Design Overview

The focus of this document is to extend DHCP to piggyback SA information for the entities that want to employ IPSec, providing a

quick configuration for security parameters. It is especially appropriate for processes and devices that already interpret DHCP messages.

With the extended DHCP, the host especially the roaming host is able to build trust relationship with the target server, the service from which is desired to be provided in a security channel. Accordingly, the host called requestor in this document is on the initiative in the course of the SA establishment in SOLICIT message. The DHCP server configured to take the responsibility responds. The requestor SHOULD also present its public key to DHCP server for encrypting the symmetric key as an element of SA, and the DHCP server MUST provide the symmetric key in cipher text together with other parameters of the very SA.

If selected by REQUEST message, the DHCP server will determine which target server will be configured with SA parameters according to the SA establishment request indicated by the requestor in a new defined option. Then the DHCP server sends message to convey the SA to the selected target server and waiting REPLY message to make sure whether the SA parameters have been configured successfully or not. To guarantee the confidentiality of the symmetric key, the access key SHOULD be encrypted by pre-shared key.

Once the DHCP server gets the confirmation of SA configuration from the intended target server, it responds to requestor in REPLY message that includes SA parameter shared between the requestor and the target server whose service is wanted to be secured by the requestor.

If needed, the DHCP serve MAY choose to update a current SA by sending RECONFIG-INIT message to the requestor.

5. Extended DHCPv6 Operation

5.1. Message and Option Definitions

5.1.1. Messages

Extended DHCP for Security Association Configuration specified in this document uses the Client/Server message formats described in [RFC3315], Section 6. Two new message codes are defined:

SACONFIGURATION (TDB by IANA)) - A DHCP server sends a SACONFIGURATION message to a target server to configure the SA parameters that are indicated in an option called OPTION_SA.

SACONFIGURATION-REPLY (TDB by IANA) - A target server sends a REGISTRATION-REPLY message to a DHCP server the SA from which has

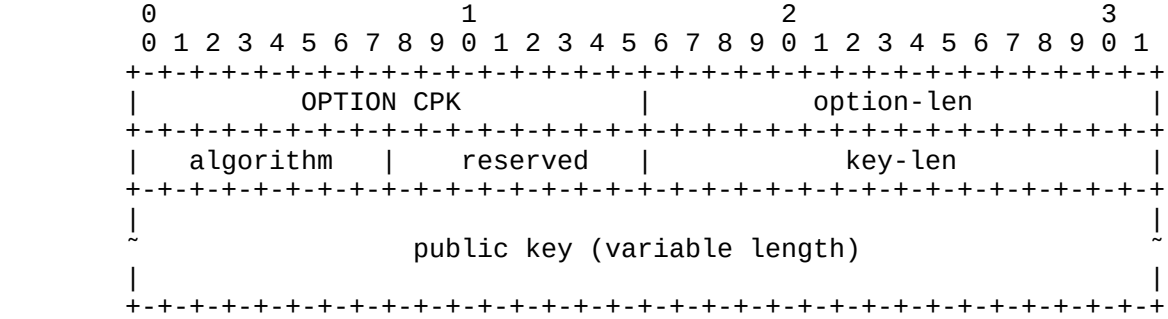
been configured successfully.

5.1.2. Options

5.1.2.1. Client Public Key Option

The Client Public Key Option is used to specify the public key associated with the client that sends the option. The Client Public Key Option SHOULD be bound to the DUID of the client.

The format of the Client Public Key Option is:



option-code OPTION_CPK (TDB by IANA).

option-len 4 + length of public key field.

algorithm the algorithm used to perform the encryption of the shared key.The algorithm are:

RSA(1), defined in [RFC3447].

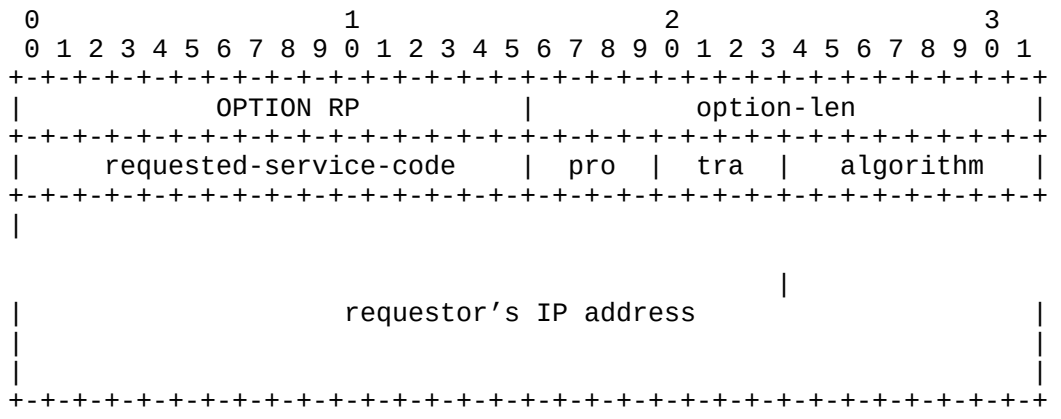
key-len length of the public key.

public key This is a variable-length field containing the public key of the DHCP client.

5.1.2.2. Requestor’s Parameters Option

The Requestor’s Parameters Option is used to specify the security parameter provided by the requestor, with which the SA will be established. The Requestor’s Parameters Option must be encapsulated in the Options field of an SA Request Option.

The format of the Requestor’s Parameters is:



option-code OPTION_RP (TDB by IANA).

option-len 20.

requested-service-code To indicate which service is wanted to be secured according to the requestor. The requested-service-code is consistent with the option-code which means if the host wants to use IPSec with a DNS recursive name server, it set the requested-service-code as the option-code for the addresses of DNS recursive name servers.

pro To specify the security protocol IPSec employ.

AH(1).

ESP(2).

tra To specify which traffic, INBOUND or OUTBOUND will be processed from the point of view of the requestor.

INBOUND(1), traffic from the target server to the requestor.

OUTBOUND(2), traffic from the requestor to the target server.

Two-Way(3), all traffic between the requestor and the target server.

algorithm The algorithm employed by IPSec:

HMAC-MD5-96(1), defined in [RFC2403]

HMAC-SHA1-96(2), defined in [RFC2404]

DES(3), defined in [RFC1829]

3DES(4), defined in [RFC1851]

DES-CBC(5), defined in [RFC2405]

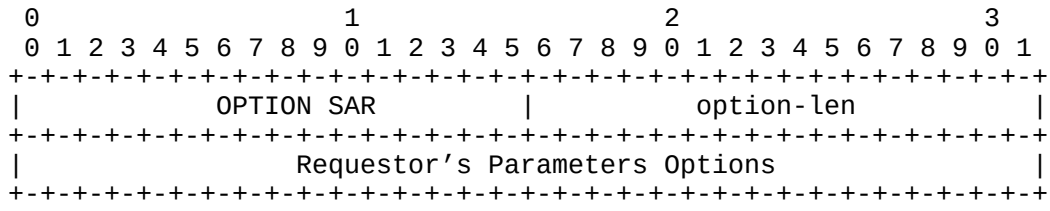
AES(6), defined in [RFC3686]

requestor's IP address To indicate which address of the requestor will be involved in SA. If the address is of all-zero and the requestor's has one or more assigned address from the DHCP server, the DHCP server will select an involved address for the requestor.

5.1.2.3. SA Request Option

The SA Request option is used to encapsulate SA Requestor's Parameters Option(s) to indicate which local service is required to be secured by the requestor and the related parameters.

The format of the SA Request Option is:



option-code OPTION_SAR (TDB by IANA).

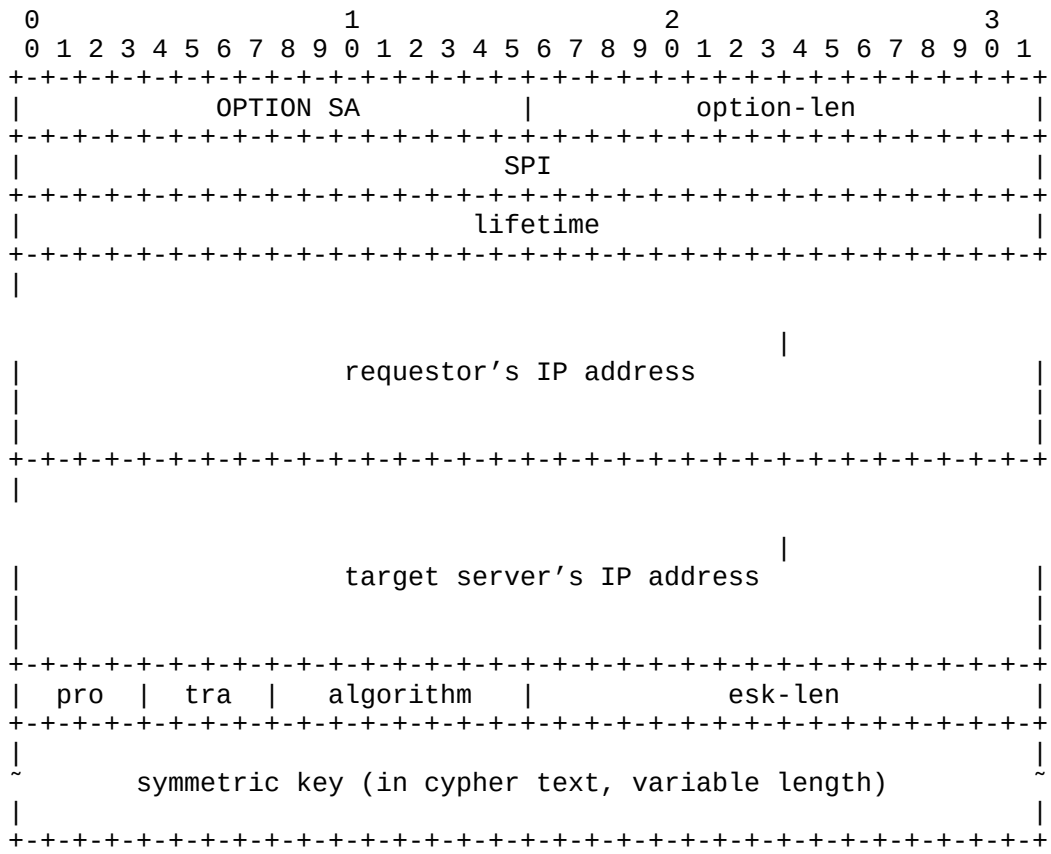
option-len 20 * number of Requestor's Parameters Options.

Requestor's Parameters Options Different Requestor's Parameters Options intended for target servers of different local services.

5.1.2.4. SA Option

The SA Option is used to specify the SA parameters shared between the requestor and a specific target server.

The format of the SA Option is:



option-code OPTION_SA (TDB by IANA).

option-len 44 + length of symmetric key field.

SPI Security Parameter Index created by the DHCP serverand defined in IPSec specification [RFC2401].

lifetime The valid lifetime for the SA, expressed in units of seconds.

requestor's IP address A copy of the requestor's IP address field in the Requestor's Parameters Option included in SOLICIT message or REQUEST message.

target server's IP address To indicate which address of the target server will be involved in SA. The addresses of a specific target server SHOULD be maintained by the DHCP server.

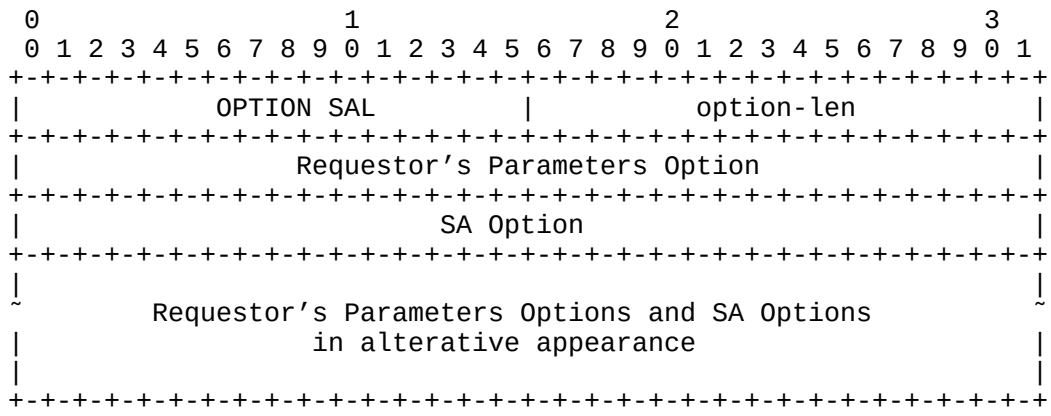
pro A copy of the pro field of Requestor's Parameters Option included in SOLICIT message or REQUEST message.

tra	A copy of the tra field of Requestor's Parameters Option included in SOLICIT message or REQUEST message.
algorithm	A copy of the algorithm field of Requestor's Parameters Option included in SOLICIT message or REQUEST message.
esk-len	The length of the encrypted symmetric key.
symmetric key	The encryption of the key shared by the requestor and the target server to use IPSec.

5.1.2.5. SA List Option

The SA List Option is used to contain one or more SA Options in response to the SA request indicated in the SA Request Option included in SOLICIT message from the requestor. In the SA List Option, the Requestor' parameters Option and the SA Option give the alterative appearances to indicate their correlation.

The format of the SA List Option is:



option-code OPTION_SAR (TDB by IANA).

option-len 20 * number of Requestor's Parameters Options + the total length of all SA Options.

Requestor's Parameters Option A copy of Requestor's Parameters Option of the SA Request Option included in SOLICIT message or REQUEST message.

SA Option A SA Option in response to a specific Requestor's Parameters Option included in SOLICIT message or REQUEST message.

5.1.3. Status Codes

Some status codes defined in [RFC5007] are redefined in the messages newly defined in this document, together with the new status codes, they are defined:

NotConfigured (9) - The status code is originally defined in [RFC5007] and redefined with the massage defined in this document. The intended SACONFIGURATION-receiver is not configured to provide secured service based on IPSec.

NotAllowed (10) - The status code is originally defined in [RFC5007] and redefined with the massage defined in this document. The sender of SACONFIGURATION is not the authorized DHCP server to configure SA for IPSec. The authentication of the DHCP server MAY be based on Authentication of DHCP Messages specified in Section 2.3.1 of [RFC3315].

InappropriateAddress (TBD) - The requestor's IP address in the

Requestor's Parameters Option is not a authorised one for the requestor or the target server's IP addressis in the SA Option is not one of the valid IP address of the target server.

UnsupportedAlgorithm (TBD) - The algorithm specified in the Requestor's Parameters Option is not supported by the target server whose service is wanted to be secured.

MalformedSACONFIGURATION (TBD) - The SACONFIGURATION is not valid; for example, the required symmetric key field is missing from the SA Option. This status code is used only in SACONFIGURATION-REPLY message.

FailedSACONFIGURATION (TBD) - The SACONFIGURATION interaction between the DHCP server and the target server is failed for some reasons. This status code is used only in REPLY message from DHCP server to the requestor.

5.1.4. Transmission and Retransmission Parameters

This section presents a table of values used to describe the message transmission behavior for IP address registration.

Parameter	Default	Description
SAC_TIMEOUT	1 sec	Initial SACONFIGURATION timeout
SAC_MAX_RT	10 secs	Max SACONFIGURATION timeout value
SAC_MAX_RC	5	Max SACONFIGURATION retry attempts

5.2. Message Validation

The Message Validation specified in this document follow the basic message validation principles in [RFC3315], Section 15. Requestors, target servers and DHCP servers SHOULD discard any messages that contain options that are not allowed to appear in the received message.

5.2.1. SOLICIT

Extended DHCP for Security Association Configuration with the Local Server in the Access Network mandates the confidentiality of the shared symmetric key. The requestor MUST include Client Public key Option for encrypting the shared symmetric key. DHCP Servers MUST discard any received SOLICIT messages that meet any of the following conditions:

o the message does not include an OPTION_SERVERID option.

5.2.2. SACONFIGURATION

The target server MUST discard any received SACONFIGURATION messages that meet any of the following conditions:

o the message does not include an OPTION_SERVERID option.

o the message includes an OPTION_CLIENTID option but the contents of the OPTION CLIENTID option does not match the target server's identifier.

o the message does not include a SA Option.

DHCP servers, relay agents and the non-requestor DHCP clients MUST discard any received SACONFIGURATION messages.

5.2.3. SACONFIGURATION-REPLY

The target server MUST discard any received SACONFIGURATION-REPLY messages that meet any of the following conditions:

o the message does not include an OPTION_CLIENTID option.

o the message includes an OPTION_SERVERID option but the contents of the OPTION_SERVERID option does not match the server's identifier.

o the message does not include a SA Option.

o the "transaction-id" field in the message does not match the value used in the original message.

Target servers (on the DHCP server port, 546 [RFC3315]), relay agents and DHCP clients MUST discard any received SACONFIGURATION-REPLY messages.

5.3. DHCP Server Solicitation

This section describes how the SA configuration with the target server in the access network will affect DHCP Server Solicitation specified in [RFC3315], Section 17.

Once a host especially a mobile one intends to establish a SA with a local server in the access network to secure one certain service such as DNS recursive resolution service, it will include a SA Request Option together with the Client Public Key Option in its SOLICIT message.

Once informed by SA Request Option included in the SOLICIT message,

the DHCP server will decide whether to provide the service for SA establishment according to its policies configured by the administrator of the access network. If a DHCP server takes the responsibility in managing SA establishment for the requestor and the target server, it responds, via the ADVERTISE message, to the requestor it is able to find a proper IP address for IPSec as the requestor's agent.

On receiving the ADVERTISE messages from several DHCP servers, the requestor selects one according to its local policies and then multicast the REQUEST message including the SA Request Option. Once the selected DHCP server gets the REQUEST message, it decides which IP address will be involved in the intended SA establishment according to the requested-service-code of the Requestor's Parameters Option in the SA Request Option.

Then DHCP server initiates SACONFIGURATION exchange with the target server. Once the SACONFIGURATION has been confirmed from the intended target server, DHCP server sends Reply message to the requestor.

The DHCP server use the SA binding to manage the SA parameters shared between the requestor and the target server. The SA binding is indexed by the tuple [requestor's DUID, target server's DUID, SPI].

The SACONFIGURATION mechanism is not compatible with Rapid-commit mode specified in [RFC3315], Section 17.1.1.

5.4. SACONFIGURATION Behavior of DHCP Server

Once receiving a REQUEST message with SA Request Option, the DHCP server then initiates the SACONFIGURATION exchange with the target server.

This section describes how DHCP Server initiates exchange with a specific target server in SACONFIGURATION.

5.4.1. Creation of SACONFIGURATION

The DHCP server sets the "msg-type" field to SACONFIGURATION. The DHCP server generates a transaction ID and inserts this value in the "transaction-id" field. The DHCP server MUST include an OPTION_ SERVERID option to identify itself to the target server which works as a DHCP client. The DHCP server MUST include a SA Option specified in Section 5.1.2.4.

5.4.2. Transmission of SACONFIGURATION

According to the requested-service-code of the Requestor's Parameters Option in the SA Request Option included in the REQUEST message and the address list of one certain local service, the DHCP server is able to choose an IP address of a proper target server for the requestor to establish SA.

The DHCP server transmits SACONFIGURATION messages according to Section 14 of [RFC3315], using the following parameters:

IRT SAC_TIMEOUT

MRT SAC_MAX_RT

MRC SAC_MAX_RC

MRD 0

If the message exchange fails, the DHCP server takes an action based on the local policy of the access network. Examples of actions the DHCP server might take include:

o Inform the client of the failure with denying offering service.

o Inform the client of the failure while assigning IP address as usual.

5.4.3. Receipt of SACONFIGURATION-REPLY

A successful SACONFIGURATION-REPLY is one without an OPTION_STATUS_CODE option (or an OPTION_STATUS_CODE option with a success code). Then the DHCP server responds to the requestor in the REPLY message with SA List Option to indicate the SA to be established between the requestor and one certain target server.

An unsuccessful SACONFIGURATION-REPLY is one that has an OPTION_STATUS_CODE with an error code. Depending on the status code, the DHCP server may try a different target server (such as for NotAllowed, and NotConfigured), try a different or corrected SACONFIGURATION request (such as for MalformedSACONFIGURATION and FailedRegistraion), or terminate the SACONFIGURATION request.

5.5. Target Server Behavior

A Target Server sends SACONFIGURATION-REPLY messages in response to valid SACONFIGURATION messages it receives to inform the DHCP server that the information conveyed by the SA Option has been configured.

5.5.1. Receipt of SACONFIGURATION Messages

Upon receipt of a valid SACONFIGURATION message, the target server updates the SA database it maintains and returns a SACONFIGURATION-REPLY.

No matter whether the SA establishment has been successfully built, the target server configures itself the SA. Once the lifetime of the SA index by SPI expires, the target server removes this SA.

With SPI field of the SA Option, SAVP decides how to deal with the binding request described in SACONFIGURATION message:

o If the SPI is new to the target server, the target server records the SA parameters included in the SACONFIGURATION message.

o If the SPI has been maintained by the target server, the target server overwrites the SA parameters index by the SPI as an update.

The target server constructs a SACONFIGURATION-REPLY message by setting the "msgtype" field to SACONFIGURATION-REPLY, and copying the transaction ID from the SACONFIGURATION message into the transaction-id field. If the SA intended to be configured is not a proper one, the target server adds the error status code according to Section 5.1.3 and sends the SACONFIGURATION-REPLY to the DHCP server.

If the target server fails to configure SA for some unknown reasons, the target server MAY discard the SACONFIGURATION message or MAY add an OPTION_STATUS_CODE option with the FailedSACONFIGURATION status code and send the SACONFIGURATION-REPLY to the DHCP server.

5.5.2. Transmission of SACONFIGURATION-REPLY Messages

The target server sends the SACONFIGURATION-REPLY message as described in the "Transmission of Reply Messages" section of [RFC3315].

5.6. The extension brings to the DHCP exchange

As in Section 5.1.2, some new defined options MUST be supported for the DHCP server solicitation. And on the receipt of REQUEST message containing the SA Request Option from the requestor, the DHCP server initiates the SA establishment exchange with the intended target server before it assigns IP address and other network parameters for the requestor.

The IP address registration exchange happens every time the DHCP server updates the lease on IP address assigned, which means once the

DHCP server is ready to reply to the RENEW or REBIND message, the IP address registration exchange takes places before sending the these very messages for DHCP client.

6. Security Considerations

The original intention of the extension to DHCP for piggybacked SA configuration makes security consideration a necessity. As mentioned before, the symmetric key to be used for IPSec should be in a confidential form. By the virtue of the Client Public Key Option, the DHCP server is able to use client's public key to encrypt the symmetric key and included it in SA Option. As for the very symmetric key transmission between the DHCP server and the target server, administrator MAY choose to make a key pre-shared between the DHCP server and the target server, or MAY employ the public key scheme for the encryption of the symmetric key shared between the DHCP server and the target server.

7. IANA Considerations

IANA is requested to assign the following new DHCPv6 Message types in the registry maintained in http://www.iana.org/assignments/dhcpv6-parameters:

SACONFIGURATION

SACONFIGURATION-REPLY

IANA is requested to assign the following new DHCPv6 Option Codes in the registry maintained in http://www.iana.org/assignments/dhcpv6-parameters:

OPTION_OPTION CPK

OPTION_OPTION RP

OPTION_OPTION SAR

OPTION_OPTION SA

OPTION_OPTION SAL

IANA is requested to assign the following new DHCPv6 Status Codes in the registry maintained in http://www.iana.org/assignments/dhcpv6-parameters:

InappropriateAddress

UnsupportedAlgorithm

MalformedSACONFIGURATION

FailedSACONFIGURATION

8. References

8.1. Normative References

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

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

[RFC2401] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[RFC2403] Madson, C. and R. Glenn, "The Use of HMAC-MD5-96 within ESP and AH", RFC 2403, November 1998.

[RFC2404] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", RFC 2404, November 1998.

[RFC1829] Karn, P., Metzger, P., and W. Simpson, "The ESP DES-CBC Transform", RFC 1829, August 1995.

[RFC1851] Karn, P., Metzger, P., and W. Simpson, "The ESP Triple DES Transform", RFC 1851, September 1995.

[RFC2405] Madson, C. and N. Doraswamy, "The ESP DES-CBC Cipher Algorithm With Explicit IV", RFC 2405, November 1998.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, January 2004.

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.

8.2. Informative References

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

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

Author's Address

Di Ma
CNNIC
4, South 4th Street,Zhongguancun,
Beijing
China(100085)

Phone: +86 010 58813216
EMail: madi@cnnic.cn

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: April 28, 2011

T. Mrugalski
Gdansk University of Technology

Oct 25, 2010

Remote DHCPv6 Autoconfiguration
draft-mrugalski-remote-dhcpv6-01

Abstract

This document describes remote autoconfiguration mechanism, an extension to DHCPv6 protocol. Every time a node attaches to a new link, it must renew or obtain new address and parameters, using DHCPv6 protocol. For mobile nodes it is beneficial to obtain address and other configuration parameters remotely, before actually attaching to destination link. This document defines mechanism using remote configuration and new options required to remotely discover destination DHCPv6 servers. Remote unicast communication with DHCPv6 servers is also defined.

Requirements Language

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

Status of this Memo

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

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

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

This Internet-Draft will expire on April 28, 2011.

Copyright Notice

Table of Contents

1. Introduction

1.1. Classic handover and DHCPv6 configuration

During normal operation, a node attaches to a link (e.g. after completing, waking up from standby mode, booting up etc.) and, if certain condtions are met (see [RFC4862]), it initiates DHCPv6 operation. It is assumed that initial communication with servers (directly or via relays) is performed locally, i.e. client and server or relay are attached to the same link.

In certain mobility scenarios, it may be beneficial to initiate configuration and obtain parameters before physically attaching to local link. When client (a mobile IPv6 node, likely to be conformant to [RFC3775]) changes its physical location, radio signal and other metrics deteriorate and eventually handover decision is made. During normal handover procedure, data link layer (e.g. IEEE 802.11 or IEEE 802.16) performs handover procedure. This phase is sometimes referred to as link layer handover. After such procedure is completed, IPv6 reconfiguration is started. After client attaches to its new location, it tries to confirm old or obtain new configuration parameters, using DHCPv6 CONFIRM or SOLICIT messages. After discovering available DHCPv6 servers, MN requests new address and possibly other configuration options. Once DHCPv6 configuration is complete, it may start other handover related activities, like updating Correspondent Nodes (CN) or Home Agent (HA) using Binding Update procedure [RFC3775]. If aforementioned actions are preformed sequencially, delays introduced by each layer are adding up, resulting in a large overall delay. This introduces gaps in communication capabilites that should be avoided, if possible.

1.2. Remote DHCPv6 Rationale

Instead of performing DHCPv6 configuration after physically attaching to a link, client communicates with server (or relay) located at destination link remotely, while still being attached to the old link. Assuming client is able to communicate with destination server, it may obtain all requested parameters. Although this mechanism is generic and not tied to any specific mobility mechanism, there are number of benefits client could gain from learning its parameters before handover:

- Mobile Client may perform certain actions earlier, e.g. Mobile Client may send Binding Update notifications to its correspondent nodes before performing handover. This may potentially halve the Round Trip Time delay.

Mobile Client may perform many actions at once, e.g. update CNs about its new address while initiating handover preparation. Some network access technologies like IEEE 802.16 (WiMAX) allow node to initiate handover procedure and remain attached to old base station. Node maintains full communication capability at that time. The exact detachment event is left up to the client.

Mobile Client may consider several target locations as target destinations. The knowledge about availability (or lack of thereof) of requested services may be leveraged during destination selection.

The exact way of exploiting gained knowledge via remote autoconfiguration mechanism is outside of scope of this document.

2. Remote DHCPv6 Operation

Client, while still connected to its old location, gains information about DHCPv6 servers located at possible handover destinations. One of the possible ways to determine such addresses is specified in Section 2.1.

After gaining knowledge about potential destinations, client chooses one or more destinations and obtains configuration parameters from each chosen location, as defined in Section 2.2. The target selection algorithm is outside of scope of this document.

After changing point of attachment, client behaves as described in [RFC3315]: attempts to verify its address using CONFIRM message.

Discussion: The concept may be briefly described as "extending unicast communication." Maybe reference to Server Unicast Option should be explicitly mentioned?

2.1. Discovering Neighboring Networks

Client, while still connected to its old location, sends regular SOLICIT, REQUEST, RENEW or REBIND message to its locally available DHCPv6 server. Client includes OPTION_NEIGHBORS code in the transmitted Option Request Option (ORO). Server responds with the list of addresses of DHCPv6 servers located at possible handover targets. Such list is conveyed in OPTION_NEIGHBORS option.

2.2. Remote Autoconfiguration

Client communicates with remote one or more DHCPv6 servers, located at potential destination. To do so, client transmits SOLICIT message

to known server unicast address and includes OPTION_REMOTE_AUTOCONF (defined in Section 3. Server, supporting remote autoconfiguration responds as if the message was received locally, e.g. responds with ADVERTISE to client's SOLICIT message. Client continues remote configuration using REQUEST or CONFIRM message. Server concludes remote autoconfiguration by responding using REPLY message.

3. DHCPv6 Options Format

Following sections define two DHCPv6 options, used in remote autoconfiguration process.

3.1. Neighbors Option Format

Neighbors Option is used to announce neighboring DHCPv6 servers, located in nearby networks. Client requests this option to learn about DHCPv6 servers located at nearby networks (potential candidates for handovers).

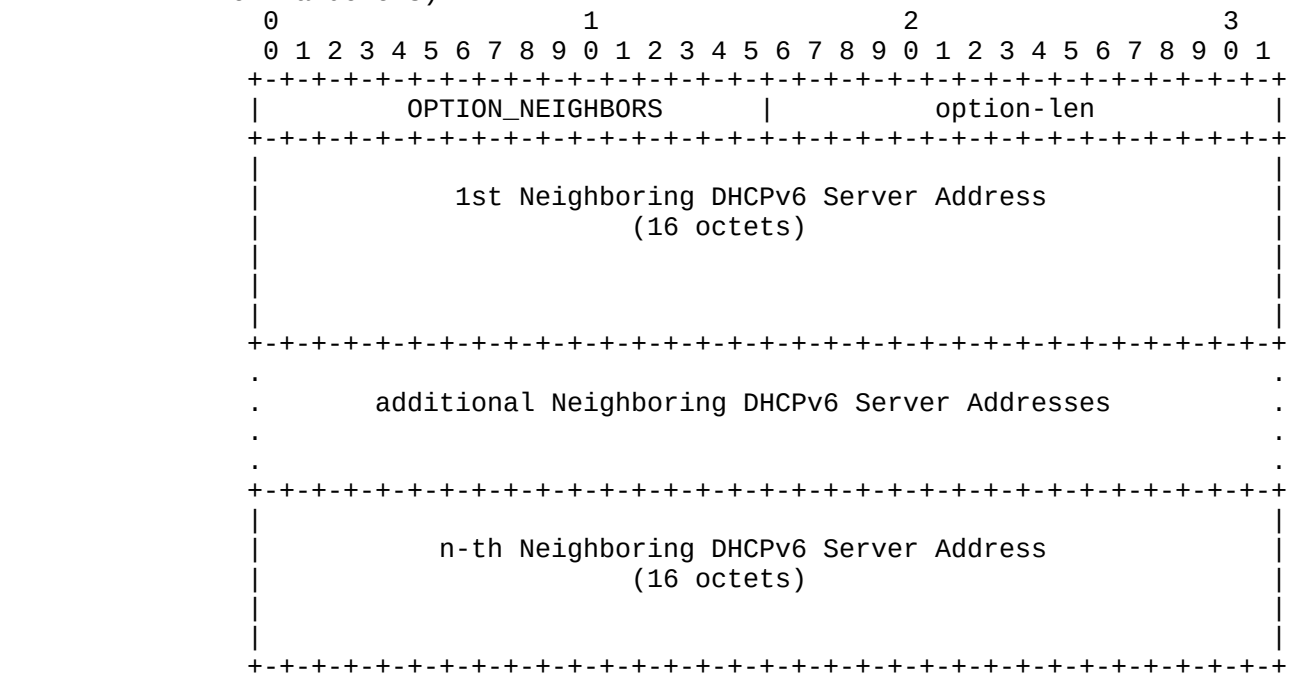


Figure 1: IPv6 Neighbors Option Format

option-code: OPTION_NEIGHBORS (TBD).

option-len: Number of specified neghbors, multiplied by 16.

Neighboring DHCPv6 server Address One or more IPv6 address of the neighboring DHCPv6 servers.

Discussion: While some network access technologies allow discovery of neighboring networks (e.g. Neighbor Advertisments or Scanning mechanism in 802.16 networks), but they provide information on link layer level (e.g. BS ID in case of 802.16 networks). Some mapping between IP information (i.e. neighboring DHCPv6 servers) and link layer level (e.g. BSID) may be considered here. If such approach is deemed feasible, Neighbors Option format should be extended with additional link layer information.

3.2. Remote Autoconfiguration Option

Remote Autoconfiguration Option is used by the client to inform server that it is performing remote configuration, rather than local.

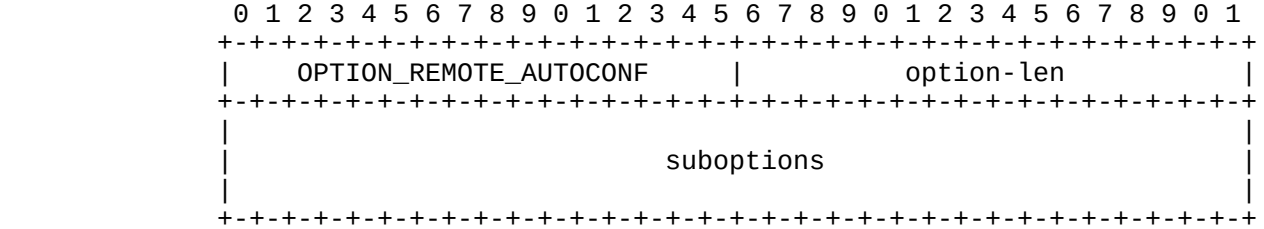


Figure 2: IPv6 Neighbors Option Format

option-code: OPTION_REMOTE_AUTOCONF (TBD).

option-len: Length of the suboptions field. 0, if there are no suboptions.

suboptions Zero or more suboptions.

The Remote Autoconfiguration Option may include suboptions. Currently there are no such suboptions defined.

Discussion: The Remote Autoconfiguration Option is used to inform server that client is currently located off-link, but would like to get on-link configuration. Also, it informs server that unicast communication from non-local prefix. However, if server is configured to allow such communication, it does not need to know about client’s current location (just treat it as regular, local

client). In that case, this option is not needed.

4. DHCPv6 Server Behavior

Servers conformant to this specification MUST support unicast communication.

Server MUST NOT send OPTION_NEIGHBOR to client, unless client explicitly requested such option.

Server MUST respond to client message transmitted off-link if OPTION_REMOTE_AUTOCONF option is included in client's message.

Server SHOULD ignore off-link messages that does not contain OPTION_REMOTE_AUTOCONF option.

5. DHCPv6 Client Behavior

Client supporting remote autoconfiguration, SHOULD request OPTION_NEIGHBORS every time it sends SOLICIT, REQUEST, RENEW, REBIND or CONFIRM messages.

Client MUST include OPTION_REMOTE_AUTOCONF in its messages, if communicating with server remotely (i.e. client and server are currently not on the same link).

Client MAY initiate remote autoconfiguration at its own discretion. Depending on client policy, that may be as soon as local configuration is completed, when radio signal quality degrades and handover is imminent or when other implementation specific conditions are met.

6. IANA Considerations

IANA is requested to allocate two DHCPv6 option codes referencing this document: OPTION_NEIGHBORS and OPTION_REMOTE_AUTOCONF.

7. Security Considerations

The overall security considerations discussed in [RFC3315] apply also to this document. OPTION_REMOTE_AUTOCONF may be used to attack known DHCPv6 server, but that does not differ from attacks directed at servers supporting unicast address option. Compromised DHCPv6 server may be used to misinform clients about available nearby networks.

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

8. Normative References

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

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3775] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.

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

Author's Address

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

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

DHC
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2011

L. Yeh, Ed.
T. Tsou
Huawei Technologies
J. Hu
Q. Sun
China Telecom
October 25, 2010

Prefix Pool Option for DHCPv6 Relay Agent
draft-yeh-dhc-dhcpv6-prefix-pool-opt-01

Abstract

The Prefix Pool option provides an automatic mechanism for the messages exchange between DHCPv6 server and DHCPv6 Relay Agent. The information about Prefix Pools maintained on DHCPv6 server can be transferred from server to relay agent through this DHCPv6 option to support the necessary route aggregation on the provide edge router, which has a huge number of routes pointing to the customer networks before.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 28, 2011.

Copyright Notice

Table of Contents

1. Introduction

DHCPv6 Relay Agents [RFC3315] are deployed to relay messages between clients and servers when they are not on the same link, and are often implemented along with a routing function on the provider edge (PE) routers [BBF WT-177]. Meanwhile, the PE router always employ the DHCPv6 Prefix Delegation [RFC3633] as the mechanism for the automated delegation of IPv6 prefix to the customer network.

In order to make the customer network to be reachable in the IPv6 network, the PE routers always need to add or remove the route entry directing to each customer network in its routing table per the messages between DHCPv6 Server (Delegation Router) and Customer router (CPE, DHCPv6 Client, DHCPv6 Requesting Router) when the PE router acts as DHCPv6 Relay Agent [BBF WT-177].

When the routing protocol is enabled on the network-facing interface of the PE router, all the routes directing to the customer networks are supposed to advertise in the ISP core network. This will make the number of entries in the routing table on the ISP core router to be unacceptable huge, so that it is necessary to aggregate the routes directing to the customer networks on the PE router.

Because the prefixes of the customer networks can not guarantee always to be valid and continuous, the routing protocol on the PE router can not make one aggregation route automatically to cover all the prefixes delegated to the customer networks, which are associated to the same client-facing link of the PE. On the other hand, the information of the prefix pools associated to each client-facing interface of PEs is always maintained on the DHCPv6 server.

When the PE router acts as the DHCPv6 Server, the aggregation routes (eg. black-hole routes) can be generated by the information of the prefix pools directly, but when the PE router acts as the DHCPv6 Relay Agent, a new mechanism to transfer the information of the prefix pools from the server to the relay agent for each client-facing interface of the PE is requested.

After the PE got the information of the prefix pools associated to its client-facing interfaces, the aggregation route entries pointing to each of the prefix pools can be added or withdrawed in the routing table of PE. When the routing protocol is enabled on PE’s network-facing interface, the above aggregation route pointing to all of the customer networks attached on the same link of the PE’s client-facing interface will be advertised to the whole ISP network.

2. Terminology and Language

This document describes new DHCPv6 options of prefix pool and the associated mechanism for the configuration on the Relay Agent. This document should be read in conjunction with the DHCPv6 specification, RFC 3315 and RFC 3633, for a complete mechanism. Definitions for terms and acronyms not specifically defined in this document are defined in RFC 3315, RFC 3633 and RFC 3769 [RFC3769].

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 BCP 14, RFC 2119 [RFC2119].

3. Scenario and Network architecture

The following figure illustrates a typical ISP-Customer network architecture.

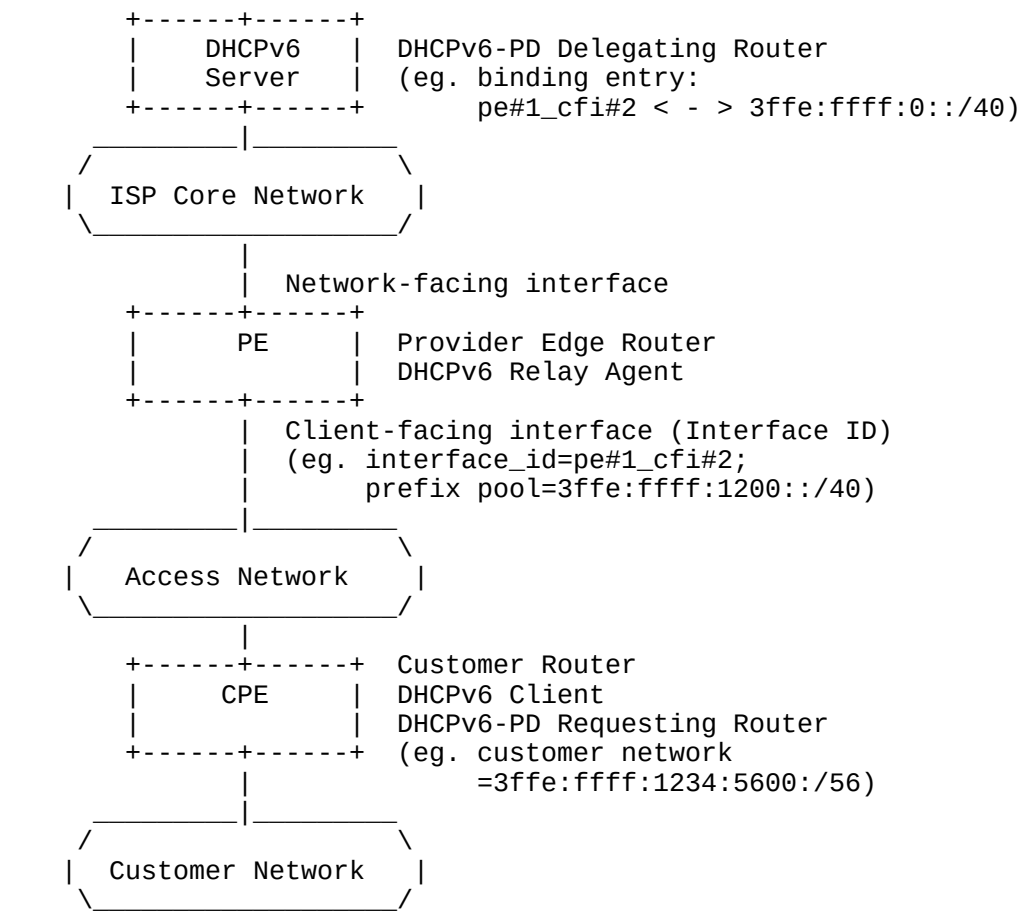
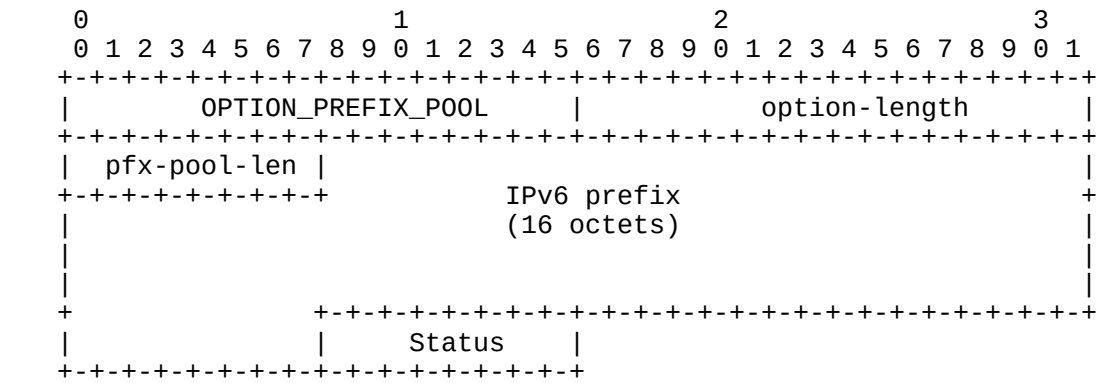


Figure 1: An example of ISP-Customer network architecture

4. Prefix Pool option

The format of the Prefix Pool option is:



option-code: OPTION_PREFIX_POOL (TBD)
option-length: 18
pfx-pool-len: Length for the prefix pool in bits
IPv6 prefix: IPv6 prefix of the prefix pool
Status: Status of the prefix pool

The Status field in the Prefix Pool option indicates the availability of the prefix pool maintained on the Server. The code of the Status is defined in the following table.

Name	Code
Valid	0
Released	1
Reserved	2˜255

5. Relay Agent Behavior

The Relay Agent who needs the information of prefix pools from the server, shall includes Option Request Option (OPTION_ORO, 6) to request Prefix Pool option from the server, who maintains the status of the prefix pools associated to the particular client-facing interface of the Relay Agent where receiving the message from clients. The Relay Agent may include the ORO for Prefix Pool Option in the relay-forward (12) message of SOLICIT (1), REQUEST (3), RENEW (5), REBIND (6) and RELEASE (8).

The Relay Agent should includes Interface ID option (OPTION_INTERFACE_ID, 18) for the server to identify the associated interface on which the prefix pool is configured, if the Server would not like to use link-address specified in the DHCPv6 message encapsulation of relay-forward message to identify the interface of the link on which clients are located.

After received the Prefix Pool option for that particular client-facing interface in the relay-reply message (13) message of REPLY (7) from the server, the Relay Agent shall add or remove the aggregation route entry per the status of the prefix pool. If the status of the prefix pool got from the server is 'Valid', the Relay Agent shall add an aggregation route entry in its routing table if the same entry has not been added in. If the status of the prefix pool got from he server is 'Released', the Relay Agent shall withdraw the associated aggregation route entry in its routing table.

The Relay Agent advertises its routing table including the entries of the aggregation routes based on the information of prefix pools when the routing protocol is enabled on its network-facing interface.

6. Server Behavior

Per RFC3633, if the prefix of the customer network associated to the IA_PD option in relay-forward message of SOLICIT , REQUEST, RENEW, REBIND is indicated to be valid on the Server, the Server (delegating router) will delegate the prefix of the customer network with the relevant parameters to the client (requesting router, customer router) in the relay-reply message of REPLY.

The Server shall use the Interface ID included in the relay-forward message by the relay agent to identify the client-facing interface of the relay agent on which the associated prefix pool will be configured. Per RFC3315, the Server may include the same Interface ID option in the relay-reply message.

After receives the ORO in the relay-forward message, the Server must include Prefix Pool option with the status indicated for the associated client-facing interface of the relay agent in the relay-reply message of REPLY.

The status of 'Valid' in the Prefix Pool option can be used to set up the prefix pool and the associated aggregation route on the relay agent; while the status of 'Released' in the Prefix Pool option can be used to withdraw the configuration of the prefix pool and the associated aggregation route on the relay agent.

On the other hand, if the prefix of the customer network associated to the IA_PD option in the relay-forward message of RELEASE is the last releasing prefix within the associated prefix pool, the Server (delegating router) shall turn the status of the associated prefix pool to be 'Released'. After receives the ORO in the relay-forward message, the Server must include Prefix Pool option with the status of 'Released' for the associated client-facing interface of the relay

agent in the relay-reply message of REPLY.

Note that multiple prefix pools may associate with the same client-facing interface of the PE router implementing Relay Agent in the binding table on the Server, and the status of the prefix pools associated to each of client-facing interface of the PE router implementing Relay Agent in the binding table can be reset by the administrator of the Server.

When the status of prefix pool is reset by manual configuration, the Server shall initiate the relay-reply message of RECONFIGURE (10), if there is at least one prefix indicated to be valid within the associated prefix pool on the Server.

7. Security Considerations

Security issues related DHCPv6 are described in section 23 of RFC 3315.

8. IANA Considerations

IANA is requested to assign an option code to Option_Prefix_Pool from the "DHCPv6 and DHCPv6 options" registry (http://www.iana.org/assignments/dhcpv6-parameters/dhcpv6-parameters.xml).

9. References

9.1. Normative References

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

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

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

[RFC3769] Miyakawa, S. and R. Droms, "Requirements for IPv6 Prefix Delegation", RFC 3769, June 2004.

9.2. Informative References

[BBF WT-177]
Broadband Forum, "IPv6 in the context of TR-101, Rev.16, Straw Ballot", September 2010.

Authors' Addresses

Leaf Y. Yeh (editor)
Huawei Technologies
Area F, Huawei Park, Bantian
Longgang District, Shenzhen 518129
P.R.China

Phone: +86-755-28971871
Email: leaf.y.yeh@huawei.com

Tina Tsou
Huawei Technologies

Email: tena@huawei.com

Jie Hu
China Telecom
No.118, Xi Zhi Men-Nei Da Jie
Xicheng District, Beijing 100035
P.R.China

Phone: +86-10-58552808
Email: huj@ctbri.com.cn

Qiong Sun
China Telecom

Email: sunqiong@ctbri.com.cn