

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 4, 2016

S. Bhandari
S. Gundavelli
M. Grayson
B. Volz
Cisco Systems
J. Korhonen
Broadcom Communications
February 01, 2016

Access Network Identifier Option in DHCP
draft-ietf-dhc-access-network-identifier-13

Abstract

This document specifies the format and mechanism that is to be used for encoding access network identifiers in DHCPv4 and DHCPv6 messages by defining new access network identifier options and sub-options.

Status of this Memo

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

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

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

This Internet-Draft will expire on August 4, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Motivation	3
3. Terminology	4
4. DHCPv4 Access-Network-Identifier Option	5
4.1. DHCPv4 Access-Network-Identifier Sub-options	5
4.2. DHCPv4 Access-Technology-Type Sub-option	6
4.3. DHCPv4 Network-Identifier Sub-options	7
4.3.1. DHCPv4 Network Name Sub-option	7
4.3.2. DHCPv4 Access-Point Name Sub-option	8
4.3.3. DHCPv4 Access-Point BSSID Sub-option	9
4.4. DHCPv4 Operator Identifier Sub-options	9
4.4.1. DHCPv4 Operator-Identifier Sub-option	9
4.4.2. DHCPv4 Operator-Realm Sub-option	10
5. DHCPv6 Access-Network-Identifier Options	10
5.1. DHCPv6 Access-Technology-Type Option	11
5.2. DHCPv6 Network-Identifier Options	11
5.2.1. DHCPv6 Network Name Option	11
5.2.2. DHCPv6 Access-Point Name Option	12
5.2.3. DHCPv6 Access-Point BSSID Option	12
5.3. DHCPv6 Operator Identifier Options	13
5.3.1. DHCPv6 Operator-Identifier Option	13
5.3.2. DHCPv6 Operator-Realm Option	13
6. Relay Agent Behavior	14
7. Server Behavior	14
8. IANA Considerations	15
9. Security Considerations	16
10. Acknowledgments	17
11. References	17
11.1. Normative References	17
11.2. Informative References	18
Authors' Addresses	19

1. Introduction

Access network identification (ANI) of a network device has a range of applications. For example the local mobility anchor in a Proxy Mobile IPv6 domain is able to provide access network and access operator specific handling or policing of the mobile node traffic using information about the access network to which the mobile node is attached.

This document specifies Dynamic Host Configuration Protocol for IPv4 (DHCPv4) [RFC2131] and Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315] options for access network identification that is added by Relay agent in the DHCPv4 or DHCPv6 messages towards the Server. The scope of applicability for this option is between a DHCP relay agent and a mobile access gateway where the same operator typically operates both these functions

Dynamic Host Configuration Protocol (DHCP) relay agent aware of the access network and access operator add this information in the DHCP messages. This information can be used to provide differentiated services and policing of traffic based on the access network to which a client is attached. Examples of how this information can be used in mobile networks can be found in [RFC6757].

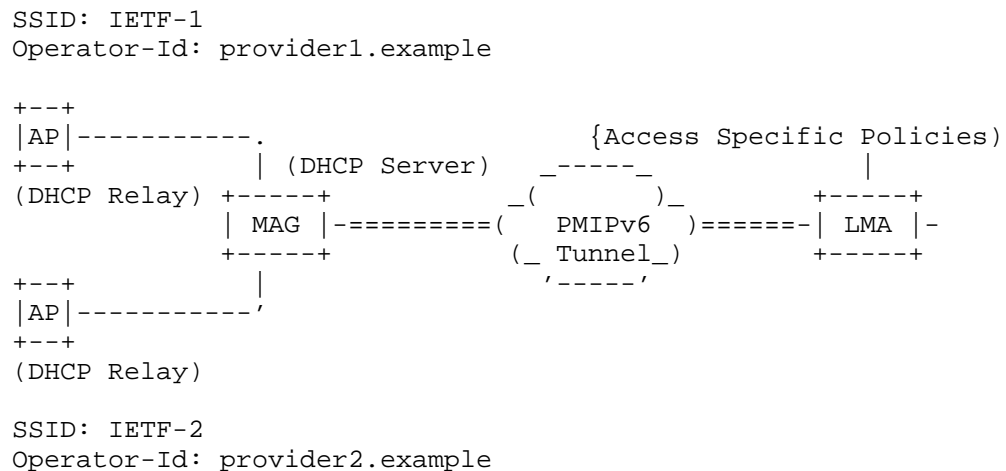
2. Motivation

Proxy mobile IPv6 [RFC5213] can be used for supporting network-based mobility management in various types of network deployments. The network architectures, such as Service provider Wi-Fi access aggregation or, WLAN integrated mobile packet core are examples where Proxy Mobile IPv6 is a component of the overall architecture. Some of these architectures require the ability of the local mobility anchor (LMA) [RFC5213] to provide differentiated services and policing of traffic to the mobile nodes based on the access network to which they are attached. Policy systems in mobility architectures such as PCC [TS23203] and ANDSF [TS23402] in 3GPP system allow configuration of policy rules with conditions based on the access network information. For example, the service treatment for the mobile node's traffic may be different when they are attached to a access network owned by the home operator than when owned by a roaming partner. The service treatment can also be different based on the configured Service Set Identifiers (SSID) in case of IEEE 802.11 based access networks. Other examples of services include the operator's ability to apply tariff based on the location.

The PMIPv6 extension as specified in [RFC6757] defines PMIPv6 options to carry access network identifiers in PMIPv6 signaling from Mobile

Access Gateway (MAG) to LMA. MAG can learn this information from DHCP options as inserted by DHCP Relay agent before MAG. If MAG relays DHCP messages to LMA as specified in [RFC5844] this information can be inserted by MAG towards LMA in the forwarded DHCP messages.

Figure 1, illustrates an example Proxy Mobile IPv6 deployment. In this example, the access network is IEEE 802.11 based access-network, the DHCP Relay Agent function is located on the access point (AP), and the DHCP Server function is located on the MAG. The MAG delivers the information elements related to the access network to the LMA over Proxy Mobile IPv6 signaling messages. The MAG obtains these information elements from the DHCP Relay Agent as per this specification. The informational elements related to the access network include the SSID of the used IEEE 802.11 network, the geo-location of the access-network to which the mobile node is attached, and the identity of the operator running the IEEE 802.11 access network infrastructure.



Access Networks attached to MAG

3. 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].

All the DHCP related terms used in this document are to be interpreted as defined in the Dynamic Host Configuration Protocol

(DHCPv4) [RFC2131] and Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315] specifications. DHCP message refers to both DHCPv4 and DHCPv6 messages throughout this document.

All the mobility related terms used in this document are to be interpreted as defined in the Proxy Mobile IPv6 specifications [RFC5213] and [RFC5844]. Additionally, this document uses the following abbreviations:

Service Set Identifier (SSID)

Service Set Identifier (SSID) identifies the name of the IEEE 802.11 network. SSID differentiates from one network to the other.

Operator-Identifier

The Operator-Identifier is the Structure of Management Information (SMI) Network Management Private Enterprise Code of the IANA-maintained "Private Enterprise Numbers" registry [SMI]. It identifies the operator running the access-network where the client is attached.

4. DHCPv4 Access-Network-Identifier Option

The Access Network Identifier carries information to identify the access network to which the client is attached. This information includes access technology type, network identifier, and access-network operator identifiers.

Relay agents that include Access Network Identifier information include one or more sub-options (see Section 4.1) in the Relay Agent Information option [RFC3046].

4.1. DHCPv4 Access-Network-Identifier Sub-options

The access network identifier information will be defined in multiple sub-options, allocated from the DHCP Relay Agent Sub-Option Codes.

ANI Sub-options: The ANI Sub-options consists of a sequence of Sub-Option Code, Length, and Value tuples for each sub-option, encoded in the following manner:

SubOpt	Len	Sub-option Data				
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
code	N	s1	s2	s3	s4	... sN

```
+-----+-----+-----+-----+-----+-----+---...-+-----+
```

Subopt code

The 1-octet code for the sub-options defined in the following sections.

Len

An unsigned 8-bit integer giving the length of the Sub-option Data field in this sub-option in octets.

Sub-option Data (s1 to sN)

The data area for the sub-option.

The initial assignment of DHCP access network identifier sub-options is as follows:

SUB-OPTION CODE	SUB-OPTION DESCRIPTION
<IANA-1>	Access Technology Type Sub-option
<IANA-2>	Access Network Name Sub-option
<IANA-3>	Access Point Name Sub-option
<IANA-4>	Access Point BSSID Sub-option
<IANA-5>	Operator-Identifier Sub-option
<IANA-6>	Operator-Realm Sub-option

4.2. DHCPv4 Access-Technology-Type Sub-option

This sub-option is used for exchanging the type of the access technology of the network to which the client is attached. Its format is as follows:

0	1	2	3
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1
Subopt Code	Length	Reserved	ATT

Subopt Code
<IANA-1>.

Length
2.

Reserved

An 8-bit field that is unused for now. The value MUST be initialized to 0 by the sender and MUST be ignored by the receiver.

Access-Technology-Type (ATT)

An 8-bit field that specifies the access technology through which the client is connected to the access link from the IANA name space Access Technology Type Option type value registry defined in [RFC5213].

4.3. DHCPv4 Network-Identifier Sub-options

These sub-options are used for carrying the name of the access network (e.g., a SSID in case of IEEE 802.11 Access Network, or PLMN Identifier [TS23003] in case of 3GPP access) and Access Point name to which the client is attached. The format of these sub-options is defined the following sections. The Network-Identifier sub-options are only for the currently known access technology types.

4.3.1. DHCPv4 Network Name Sub-option

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Subopt Code										Length																													
Network Name (e.g., SSID or PLMNID)																																							

Subopt Code
<IANA-2>.

Length
The length of the Network Name field.

Network Name

The name of the access network to which the mobile node is attached. The encoding MUST be UTF-8 as described in [RFC3629].

The type of the Network Name is dependent on the access technology to which the mobile node is attached. For IEEE 802.11 based networks, the network name will be the SSID of the network. For 3GPP access based it is the PLMN Identifier of the access network and for 3GPP2 access, the Network Name is the Access Network Identifier[ANI].

When encoding the PLMN Identifier, both the Mobile Network Code (MNC) [TS23003] and Mobile Country Code (MCC) [TS23003] MUST be 3 digits. If the MNC in use only has 2 digits, then it MUST be preceded with a '0'.

4.3.2. DHCPv4 Access-Point Name Sub-option

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-----+-----+-----+-----+-----+-----+-----+-----+
      | Subopt Code |      Length      |                               |
      +-----+-----+-----+-----+-----+-----+-----+-----+
      .
      .                               .
      .                               .
      +-----+-----+-----+-----+-----+-----+-----+-----+

```

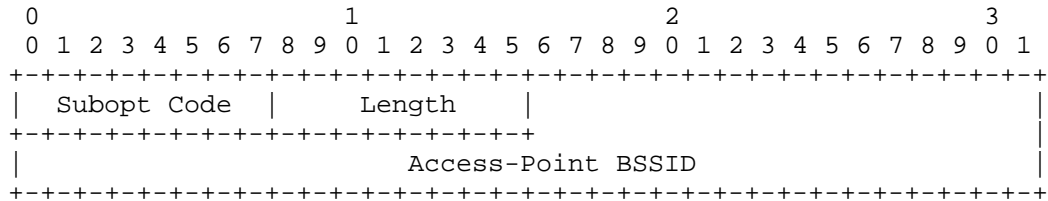
Subopt Code
<IANA-3>.

Length
The length of the Access-Point Name field.

Access-Point Name

The name of the access point (physical device name) to which the mobile node is attached. This is the identifier that uniquely identifies the access point. While Network Name (e.g., SSID) identifies the operator's access network, Access-Point Name identifies a specific network device in the network to which the mobile node is attached. In some deployments, the Access-Point Name can be set to the string representation of the Media Access Control (MAC) address as specified in [RFC6991] mac-address string type of the device or some unique identifier that can be used by the policy systems in the operator network to unambiguously identify the device. The encoding MUST be UTF-8 as described in [RFC3629].

4.3.3. DHCPv4 Access-Point BSSID Sub-option



Subopt Code
<IANA-4>.

Length
6.

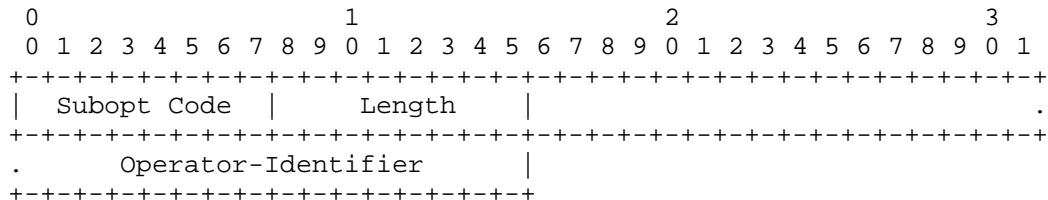
Access-Point BSSID

The 48-bit Basic Service Set Identification (BSSID) of the access point to which the mobile node is attached.

4.4. DHCPv4 Operator Identifier Sub-options

The Operator identifier sub-options can be used for carrying the operator identifiers of the access network to which the client is attached. The format of these sub-options is defined below.

4.4.1. DHCPv4 Operator-Identifier Sub-option

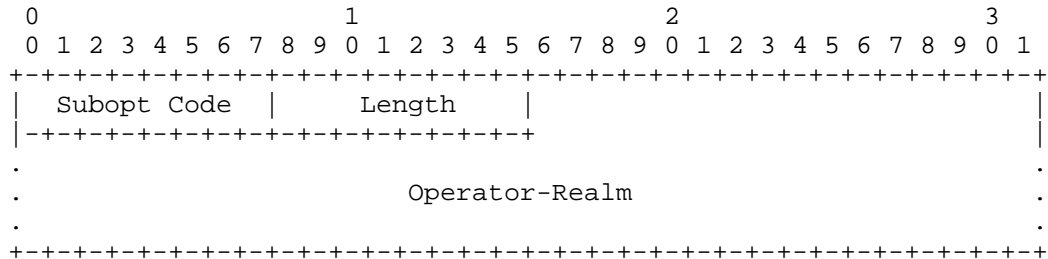


Subopt Code <IANA-5>.

Length
4.

Operator-Identifier Operator-Identifier as a variable-length Private Enterprise Number (PEN) [SMI] encoded in a network-byte order. Please refer to (section 3.1.3 of [RFC6757]) for additional details.

4.4.2. DHCPv4 Operator-Realm Sub-option



Subopt Code
<IANA-6>.

Length
The length of the Operator Realm field.

Operator-Realm
Realm of the operator (e.g., EXAMPLE.COM). Please refer to (section 3.1.3 of [RFC6757]) for additional details.

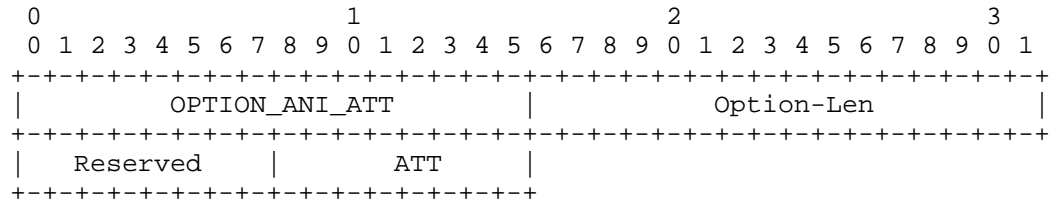
5. DHCPv6 Access-Network-Identifier Options

The Access Network Identifier options defined here may be added by the DHCPv6 Relay agent in Relay-forward messages.

OPTION CODE	OPTION DESCRIPTION
<IANA-7>	OPTION_ANI_ATT
<IANA-8>	OPTION_ANI_NETWORK_NAME
<IANA-9>	OPTION_ANI_AP_NAME
<IANA-10>	OPTION_ANI_AP_BSSID
<IANA-11>	OPTION_ANI_OPERATOR_ID
<IANA-12>	OPTION_ANI_OPERATOR_REALM

5.1. DHCPv6 Access-Technology-Type Option

This option is used for exchanging the type of the access technology the client is attached to the network. Its format is as follows:



Option-Code

OPTION_ANI_ATT (<IANA-7>).

Option-Len

2.

Reserved

An 8-bit field that is unused for now. The value MUST be initialized to 0 by the sender and MUST be ignored by the receiver.

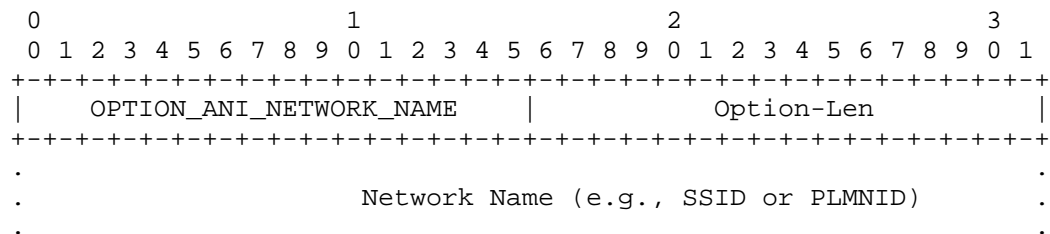
Access Technology Type (ATT):

The contents of this field is the same as the ATT field described in Section 4.2.

5.2. DHCPv6 Network-Identifier Options

These options can be used for carrying the name of the access network (e.g., a SSID in case of IEEE 802.11 Access Network, or PLMN Identifier [TS23003] in case of 3GPP access) and Access Point name to which the client is attached. The format of these options is defined below.

5.2.1. DHCPv6 Network Name Option



```

+-----+
Option-Code
  OPTION_ANI_NETWORK_NAME (<IANA-8>).

Option-Len
  The length of the Network Name field.

Network Name
  The contents of this field is the same as the Network Name field
  described in Section 4.3.1.

```

5.2.2. DHCPv6 Access-Point Name Option

```

      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+
|          OPTION_ANI_AP_NAME          |          Option-Len          |
+-----+-----+-----+-----+
.                                     .
.                                     .
.                                     .
+-----+-----+-----+-----+

```

```

Option-Code
  OPTION_ANI_AP_NAME (<IANA-9>).

Option-Len
  The length of the Access-Point Name field.

```

```

Access-Point Name
  The contents of this field is the same as the Access-Point Name
  field described in Section 4.3.2.

```

5.2.3. DHCPv6 Access-Point BSSID Option

```

      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+
|          OPTION_ANI_AP_BSSID          |          Option-Len          |
+-----+-----+-----+-----+
|                                     Access-Point BSSID                                     |
+-----+-----+-----+-----+
|                                     +-----+-----+-----+-----+                                     |
|                                     |                                     |                                     |
+-----+-----+-----+-----+

```

Option-Code

OPTION_ANI_AP_BSSID (<IANA-10>).

Option-Len

6.

Access-Point BSSID

The contents of this field is the same as the Access-Point BSSID field described in Section 4.3.3.

5.3. DHCPv6 Operator Identifier Options

The Operator Identifier options can be used for carrying the operator identifier of the access network to which the client is attached. The format of these options is defined below.

5.3.1. DHCPv6 Operator-Identifier Option

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   OPTION_ANI_OPERATOR_ID   |   Option-Len   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Operator-Identifier                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Option-Code

OPTION_ANI_OPERATOR_ID (<IANA-11>).

Option-Len

4.

Operator-Identifier

The contents of this field is the same as the DHCPv4 Operator-Identifier Sub-option field described in Section 4.4.1.

5.3.2. DHCPv6 Operator-Realm Option

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   OPTION_ANI_OPERATOR_REALM   |   Option-Len   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
.                               .
.                               .
.                               .

```

```

+-----+

```

Option-Code

OPTION_ANI_OPERATOR_REALM (<IANA-12>).

Option-Len

The length of the Operator Realm field.

Operator-Realm

The contents of this field is the same as the Operator-Realm field described in Section 4.4.2.

6. Relay Agent Behavior

DHCPv4 Relay Agents MAY include sub-options defined in section 4.2 through 4.4 in the Relay Agent Information option as defined in [RFC3046] for providing information about the access network over which DHCP messages from the client is received.

The DHCPv4 Relay Agent when including any of these sub-options in the DHCP message, DHCPv4 Network Name Sub-option (Section 4.3.1), DHCPv4 Access-Point Name Sub-option (Section 4.3.2), DHCPv4 Access-Point BSSID Sub-option (Section 4.3.3), MUST include the DHCPv4 Access-Technology-Type Sub-option (Section 4.2)

DHCPv6 Relay Agents MAY include options defined in Section 5 in Relay-forward message when forwarding any DHCPv6 message type from clients to the servers to provide information about the access network over which DHCPv6 messages from the client is received.

The DHCPv6 Relay Agent when including any of these options in the DHCP message, DHCPv6 Network Name Option (Section 5.2.1), DHCPv6 Access-Point Name Option (Section 5.2.2), DHCPv6 Access-Point BSSID Option (Section 5.2.3), MUST include the DHCPv6 Access-Technology-Type Option (Section 5.1)

7. Server Behavior

DHCPv4 base specification [RFC2131] requires that the DHCPv4 server ignore the DHCPv4 Access Network Identifier option if it does not understand the option.

If the DHCPv4 server does not understand the received sub-option defined in sections 4.1 through 4.4 in the DHCPv4 Relay Agent Information option (82) it MUST ignore those sub-options only. If DHCPv4 Server is able to process the DHCPv4 Access Network Identifier

sub-options defined in sections 4.1 through 4.4 received in DHCPv4 Relay Agent Information option, it MAY use this information obtained from the sub-option for address pool selection, or for policy decisions as per its configured policy. This information obtained from the sub-option SHOULD NOT be stored unless it is absolutely needed. However, if it is stored, the information MUST be deleted as quickly as possible to eliminate any possibility of the information getting exposed to an intruder.

If the received DHCPv4 message does not include DHCPv4 Access-Technology-Type Sub-option (Section 4.2), but if it includes any one of these other options, DHCPv4 Network Name Sub-option (Section 4.3.1), DHCPv4 Access-Point Name Sub-option (Section 4.3.2), or DHCPv4 Access-Point BSSID Sub-option (Section 4.3.3), then the DHCPv4 server MUST ignore the received DHCPv4 Access-Network-Identifier option and process the rest of the message as per the base DHCPv4 specifications

DHCPv6 base specification [RFC3315] requires that the DHCPv6 server ignore the DHCPv6 Access-Network-Identifier option if it does not understand the option.

If the DHCPv6 server receives the options defined in Section 5 and is configured to use the options defined in Section 5, it SHOULD look for the DHCPv6 Access Network identifier options in the Relay-forward message of the DHCPv6 relay agent(s) based on its configured policy. The server MAY use received ANI options for its address pool selection policy decisions as per its configured policy. This information obtained from the options SHOULD NOT be stored unless it is absolutely needed. However, if it is stored, the information MUST be deleted as quickly as possible to eliminate any possibility of the information getting exposed to an intruder.

If the received DHCPv6 message does not include DHCPv6 Access-Technology-Type Option (Section 5.1), but it includes any one of these other options, DHCPv6 Network Name Option (Section 5.2.1), DHCPv6 Access-Point Name Option (Section 5.2.2), or DHCPv6 Access-Point BSSID Option (Section 5.2.3), then the DHCPv6 server MUST ignore the received DHCPv6 Access-Network-Identifier option and process the rest of the message as per the base DHCPv6 specifications.

8. IANA Considerations

IANA is requested to assign Sub-option codes for the following DHCPv4 Sub-options from the "DHCP Relay Agent Sub-Option Codes" registry, <<http://www.iana.org/assignments/bootp-dhcp-parameters>>:

SUB-OPTION CODE	SUB-OPTION DESCRIPTION
<IANA-1>	Access Technology Type Sub-option
<IANA-2>	Access Network Name Sub-option
<IANA-3>	Access Point Name Sub-option
<IANA-4>	Access Point BSSID Sub-option
<IANA-5>	Operator Identifier Sub-option
<IANA-6>	Operator Realm Sub-option

IANA is requested to assign option codes for the following DHCPv6 options from the "Option Codes registry for DHCPv6" registry <<http://www.iana.org/assignments/dhcpv6-parameters>>, as specified in [RFC3315]:

OPTION CODE	OPTION DESCRIPTION
<IANA-7>	OPTION_ANI_ATT
<IANA-8>	OPTION_ANI_NETWORK_NAME
<IANA-9>	OPTION_ANI_AP_NAME
<IANA-10>	OPTION_ANI_AP_BSSID
<IANA-11>	OPTION_ANI_OPERATOR_ID
<IANA-12>	OPTION_ANI_OPERATOR_REALM

9. Security Considerations

Since there is no privacy protection for DHCP messages, an eavesdropper who can monitor the link between the DHCP server and relay agent can discover access network information.

[RFC3118] and [RFC3315] describe many of the threats in using DHCP. [RFC3118] and [RFC3315] each provide a solution, the Authentication Option for DHCPv4 and DHCPv6 (respectively). However, neither of

these options are in active use and therefore are not a viable mitigation option. DHCP itself is inherently insecure and thus link-layer confidentiality and integrity protection SHOULD be employed to reduce the risk of disclosure and tampering.

It is possible for a rogue DHCP relay agent to insert or overwrite with incorrect access network identifier options for malicious purposes. A DHCP client can also pose as a rogue DHCP relay agent by sending incorrect access network identifier options. While the introduction of fraudulent DHCP relay agent information options can be prevented by a perimeter defense that blocks these options unless the DHCP relay agent is trusted, a deeper defense using the authentication sub-option for DHCPv4 relay agent information option [RFC4030] SHOULD be deployed as well. Administrators SHOULD configure DHCP servers that use this option to communicate with their relay agents using IPsec, as described in Section 21.1 of [RFC3315].

The information elements that this draft is exposing is the client's access-network information. These pertain to the access network to which the client is attached, such as Access Technology Type (Ex: WLAN, Ethernet...etc), Access Point Identity (Name, BSSID), Operator Id/Realm. In deployments where this information cannot be secured using IPsec [RFC4301] or other security protocols, administrators SHOULD disable the capability specified in this document on the DHCP entities.

10. Acknowledgments

The authors would like to thank Kim Kinnear, Ted Lemon, Gaurav Halwasia, Hidetoshi Yokota, Sheng Jiang and Francis Dupont for their valuable inputs. And, to Tomek Mrugalski for a thorough review of the document.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <<http://www.rfc-editor.org/info/rfc2131>>.

- [RFC3046] Patrick, M., "DHCP Relay Agent Information Option", RFC 3046, DOI 10.17487/RFC3046, January 2001, <<http://www.rfc-editor.org/info/rfc3046>>.
- [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <<http://www.rfc-editor.org/info/rfc3315>>.

11.2. Informative References

- [ANI] "Interoperability Specification (IOS) for High Rate Packet Data (HRPD) Radio Access Network Interfaces with Session Control in the Access Network, A.S0008-A v3.0", October 2008.
- [RFC3118] Droms, R. and W. Arbaugh., Ed., "Authentication for DHCP Messages", RFC 3118, DOI 10.17487/RFC3118, June 2001, <<http://www.rfc-editor.org/info/rfc3118>>.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <<http://www.rfc-editor.org/info/rfc3629>>.
- [RFC4030] Stapp, M. and T. Lemon, "The Authentication Suboption for the Dynamic Host Configuration Protocol (DHCP) Relay Agent Option", RFC 4030, DOI 10.17487/RFC4030, March 2005, <<http://www.rfc-editor.org/info/rfc4030>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<http://www.rfc-editor.org/info/rfc4301>>.
- [RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <<http://www.rfc-editor.org/info/rfc5213>>.
- [RFC5844] Wakikawa, R. and S. Gundavelli, "IPv4 Support for Proxy Mobile IPv6", RFC 5844, DOI 10.17487/RFC5844, May 2010, <<http://www.rfc-editor.org/info/rfc5844>>.
- [RFC6757] Gundavelli, S., Ed., Korhonen, J., Ed., Grayson, M., Leung, K., and R. Pazhyannur, "Access Network Identifier (ANI) Option for Proxy Mobile IPv6", RFC 6757, DOI 10.17487/RFC6757, October 2012, <<http://www.rfc-editor.org/info/rfc6757>>.

- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<http://www.rfc-editor.org/info/rfc6991>>.
- [SMI] "PRIVATE ENTERPRISE NUMBERS, SMI Network Management Private Enterprise Codes", February 2011.
- [TS23003] "Numbering, addressing and identification", 2011.
- [TS23203] "Policy and Charging Control Architecture", 2012.
- [TS23402] "Architecture enhancements for non-3GPP accesses", 2012.

Authors' Addresses

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

Phone: +91 80 4426 0474
Email: shwethab@cisco.com

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

Email: sgundave@cisco.com

Mark Grayson
Cisco Systems
11 New Square Park
Bedfont Lakes, FELTHAM TW14 8HA
England

Email: mgrayson@cisco.com

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

Email: volz@cisco.com

Jouni Korhonen
Broadcom Communications
Porkkalankatu 24
FIN-00180 Helsinki,
Finland

Phone:
Email: jouni.nospam@gmail.com

DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 15, 2015

S. Jiang
Huawei Technologies Co., Ltd
G. Chen
China Mobile
S. Krishnan
Ericsson
R. Asati
Cisco Systems, Inc.
September 11, 2014

Registering Self-generated IPv6 Addresses in DNS using DHCPv6
draft-ietf-dhc-addr-registration-07

Abstract

In networks that are centrally managed, self-generated addresses cause some traceability issues due to their decentralized nature. One of the most important issues in this regard is the inability to register such addresses in DNS. This document defines a mechanism to register self-generated and statically configured addresses in DNS through a DHCPv6 server.

Status of This Memo

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

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

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

This Internet-Draft will expire on March 15, 2015.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	2
2. Terminology	3
3. Solution Overview	3
4. DHCPv6 ADDR-REGISTRATION-REQUEST Message	4
5. DHCPv6 Address Registration Procedure	5
5.1. DHCPv6 Address Registration Request	6
5.2. Registration Expiry and Refresh	6
5.3. Acknowledging Registration and Retransmission	6
6. Security Considerations	7
7. IANA Considerations	8
8. Acknowledgements	8
9. References	8
9.1. Normative References	8
9.2. Informative References	9
Authors' Addresses	9

1. Introduction

In several common network scenarios, IPv6 addresses are self-generated by the end-hosts by appending a self-generated interface identifier to a network-specified prefix. Examples of self-generated addresses include those created using IPv6 Stateless Address Configuration [RFC4862] , temporary addresses [RFC4941] and Cryptographically Generated Addresses (CGA) [RFC3972] etc. In several tightly controlled networks, hosts with self-generated addresses may face some limitations. One such limitation is related to the inability of nodes with self-generated addresses to register their IPv6-address-to-FQDN bindings in DNS. This is related to the fact that, in such networks, only certain nodes (e.g. The DHCPv6 server) are allowed to update these bindings in order to prevent end-hosts from registering arbitrary addresses for their FQDNs or associating their addresses with arbitrary domain names. The administrators may not want to distribute the address of authoritative name-server. Also, there is no way to propagate the address of authoritative name server by any protocols. It is preferred that the address registration server, which is under the same management with the authoritative name-server, to know the address of the authoritative name-server and make registration requests on behalf of clients. It is preferred by administrators to

establish and manage one trust relationship between a single DHCPv6 (address registration) server and the DNS authoritative name-server, rather than to distribute and manage trust relationships between many clients and the DNS authoritative name-server.

For nodes that obtain their addresses through DHCPv6, a solution has been specified in [RFC4704]. The solution works by including a Client FQDN option in the SOLICIT, REQUEST, RENEW or REBIND messages during the process of acquiring an address through DHCPv6. This document provides an analogous mechanism to register self-generated addresses in DNS.

A new ADDR-REGISTRATION-REQUEST DHCPv6 message type is defined to initiate the address registration request, and two new Status codes are defined to indicate registration errors on the server side.

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].

Certificate In this document, the term "Certificate" is all referred to public key certificate.

3. Solution Overview

After successfully assigning a self-generated IPv6 address on one of its interfaces, an end-host implementing this specification SHOULD send an ADDR-REGISTRATION-REQUEST message to a DHCPv6 address registration server. After receiving the address registration request, the DHCPv6 server registers the IPv6 address to FQDN binding towards a configured DNS server. An acknowledgement MUST be sent back to the end host to indicate whether or not the registration operation succeeded.

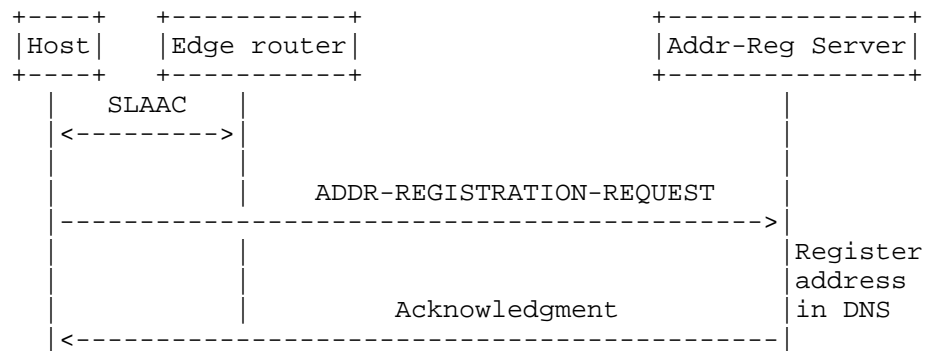


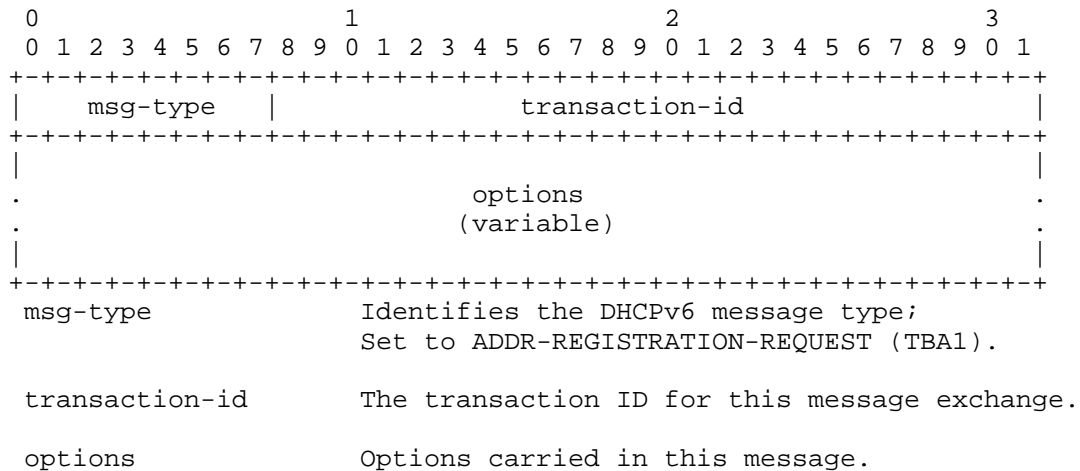
Figure 1: Address Registration Procedure

Furthermore, the registration server MAY apply certain filter/accept criteria for the address registration requests, particularly for the client chosen domain names.

It is RECOMMENDED to only set up one addressregistration server within an administration domain, although there may be multiple DHCPv6 servers. While using multiple address registration servers does potentially increase the load on DNS, because of how [RFC4703] and [RFC4704] work, this should NOT be an issue - the servers should work correctly in updating DNS (either adding or removing the entries). The broken part with multiple servers is the 'extension' of the registration. If there are two address registration servers and both receive the initial registration and (correctly) update DNS, the problem comes when the client extends this but one of the servers does not receive this extension. Then, the server that missed the extension removes the entry prematurely (i.e., when it expired originally).

4. DHCPv6 ADDR-REGISTRATION-REQUEST Message

The DHCPv6 client sends an ADDR-REGISTRATION-REQUEST message to a server to request an address to be registered in the DNS. The format of the ADDR-REGISTRATION-REQUEST message is described as follows:



DHCPv6 ADDR-REGISTRATION-REQUEST message

The ADDR-REGISTRATION-REQUEST message MUST NOT contain server-identifier option and MUST contain the IA Address option and the DHCPv6 FQDN option [RFC4704]. The ADDR-REGISTRATION-REQUEST message is dedicated for clients to initiate an address registration request toward an address registration server. Consequently, clients MUST NOT put any Option Request Option(s) in the ADDR-REGISTRATION-REQUEST message.

Clients MUST discard any received ADDR-REGISTRATION-REQUEST messages.

Servers MUST discard any ADDR-REGISTRATION-REQUEST messages that meet any of the following conditions:

- o the message does not include a Client Identifier option;
- o the message includes a Server Identifier option;
- o the message does not include at least one IA Address option;
- o the message does not include FQDN option (or include multiple FQDN options);
- o the message includes an Option Request Option.

5. DHCPv6 Address Registration Procedure

The DHCPv6 protocol is used as the address registration protocol when a DHCPv6 server performs the role of an address registration server. The DHCPv6 IA Address option [RFC3315] and the DHCPv6 FQDN option

[RFC4704] are adopted in order to fulfill the address registration interactions.

5.1. DHCPv6 Address Registration Request

The end-host sends a DHCPv6 ADDR-REGISTRATION-REQUEST message to the address registration server to the All_DHCP_Relay_Agents_and_Servers multicast address (ff02::1:2).

The end-host MUST include a Client Identifier option in the ADDR-REGISTRATION-REQUEST message to identify itself to the server. The DHCPv6 ADDR-REGISTRATION-REQUEST message MUST contain at least one IA Address option and exactly one FQDN option. The valid-lifetime field of the IA Address option MUST be set to the period for which the client would like to register the binding in DNS.

After receiving this ADDR-REGISTRATION-REQUEST message, the address registration server MUST register the binding between the provided FQDN and address(es) in DNS. If the DHCPv6 server does not support address registration function, it MUST silently drop the message.

5.2. Registration Expiry and Refresh

For every successful binding registration, the address registration server MUST record the IPv6-address-to-FQDN bindings and associated valid-lifetimes in its storage.

The address registration client MUST refresh the registration before it expires (i.e. before the valid-lifetime of the IA address elapses) by sending a new ADDR-REGISTRATION-REQUEST to the address registration server. If the address registration server does not receive such a refresh after the valid-lifetime has passed, it SHOULD remove the IPv6-address-to-FQDN bindings in DNS, also the local record.

It is RECOMMENDED that clients initiate a refresh at about 85% of the valid-lifetime. Because RAs may periodically 'reset' the valid-lifetime, the refresh timer MUST be independently maintained from the address valid-lifetime. Clients SHOULD set a refresh timer to 85% of the valid-lifetime when they complete a registration operation and only update this timer if 85% of any updated valid-lifetime would be sooner than the timer.

5.3. Acknowledging Registration and Retransmission

After an address registration server accepts an address registration request, it MUST send a Reply message as the response to the client. The acceptance reply only means that the server has taken

responsibility to registry for the client. It may not have actually completed the update yet. The server is responsible to register all the addresses in DNS. The server generates a Reply message and includes a Status Code option with value Success, a Server Identifier option with the server's DUID, and a Client Identifier option with the client's DUID.

If there is no reply received within some interval, the client SHOULD retransmits the message according to section 14 of [RFC3315], using the following parameters:

- o IRT ADDR_REG_TIMEOUT
- o MRT ADDR_REG_MAX_RT
- o MRC ADDR_REG_MAX_RC
- o MRD 0

The below presents a table of values used to describe the message transmission behavior of clients and servers:

Parameter	Default	Description
ADDR_REG_TIMEOUT	1 secs	Initial Addr Registration Request timeout
ADDR_REG_MAX_RT	60 secs	Max Addr Registration Request timeout value
ADDR_REG_MAX_RC	5	Max Request retry attempts

For each IA Address option in the ADDR-REGISTRATION-REQUEST message for which the server does not accept its associated registration request, the server adds an IA Address option with the associated IPv6 address, and includes a Status Code option with the value RegistrationDenied (TBA2) in the IA Address option. No other options are included in the IA Address option.

Upon receiving a RegistrationDenied error status code, the client MAY also resend the message following normal retransmission routines defined in [RFC3315] with above parameters. The client MUST wait out the retransmission time before retrying.

6. Security Considerations

An attacker may attempt to register large number of addresses in quick succession in order to overwhelm the address registration server. These attacks may be prevented generic DHCPv6 protection by using the AUTH option [RFC3315] or Secure DHCPv6 [I-D.ietf-dhc-sedhcpv6].

7. IANA Considerations

This document defines a new DHCPv6 message, the ADDR-REGISTRATION-REQUEST message (TBA1) described in Section 4, that requires an allocation out of the registry of Message Types defined at <http://www.iana.org/assignments/dhcpv6-parameters/>

Value	Description	Reference
TBA1	ADDR-REGISTRATION-REQUEST	this document

This document defines a new DHCPv6 Status code, the RegistrationDenied (TBA2) described in Section 5, that requires an allocation out of the registry of Status Codes defined at <http://www.iana.org/assignments/dhcpv6-parameters/>

Code	Name	Reference
TBA2	RegistrationDenied	this document

8. Acknowledgements

The authors would like to thank Ralph Droms, Ted Lemon, Bernie Volz, Sten Carlsen, Erik Kline, Lorenzo Colitti, Joel Jaeggli, Sten Carlsen, Mark Smith, Marcin Siodelski, Darpan Malhotra, Tomek Mrugalski, Jinmei Tatuya and other members of dhc and v6ops working groups for their valuable comments.

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.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 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.
- [RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005.

- [RFC4703] Stapp, M. and B. Volz, "Resolution of Fully Qualified Domain Name (FQDN) Conflicts among Dynamic Host Configuration Protocol (DHCP) Clients", RFC 4703, October 2006.
- [RFC4704] Volz, B., "The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option", RFC 4704, October 2006.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

9.2. Informative References

- [I-D.ietf-dhc-sedhcpv6]
Jiang, S., Shen, S., Zhang, D., and T. Jinmei, "Secure DHCPv6 with Public Key", draft-ietf-dhc-sedhcpv6-03 (work in progress), June 2014.

Authors' Addresses

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

Email: jiangsheng@huawei.com

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

Phone: 86-13910710674
Email: phdgang@gmail.com

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

Phone: +1 514 345 7900 x42871
Email: suresh.krishnan@ericsson.com

Rajiv Asati
Cisco Systems, Inc.
7025 Kit Creek road
Research Triangle Park, NC 27709-4987
USA

Email: rajiva@cisco.com

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

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

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

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on December 13, 2014.

Copyright Notice

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

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

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

Table of Contents

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

1. Introduction

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

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

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

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

2. Requirements Language

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

3. Terminology

This document makes use of the following terms:

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

4. Applicability

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

5. Architecture Overview

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

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

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

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

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

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

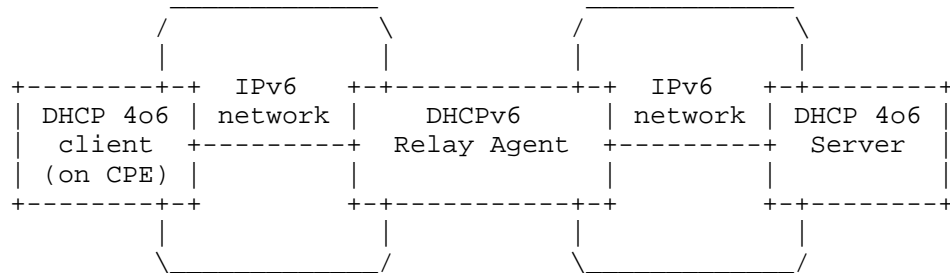


Figure 1: Architecture Overview

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

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

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

6. New DHCPv6 Messages

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

6.1. Message Types

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

6.2. Message Formats

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

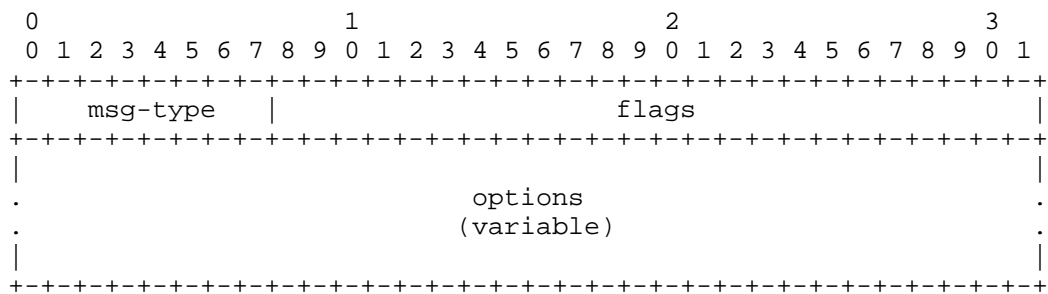


Figure 2: The format of DHCPv4-query and DHCPv4-response messages

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

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

6.3. DHCPv4-query Message Flags

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

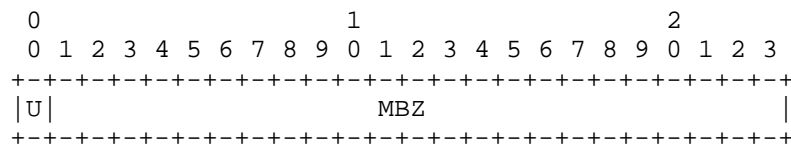


Figure 3: DHCPv4-query flags format

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

6.4. DHCPv4-response Message Flags

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

7. New DHCPv6 Options

7.1. DHCPv4 Message Option Format

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

The format of the DHCPv4 Message option is:

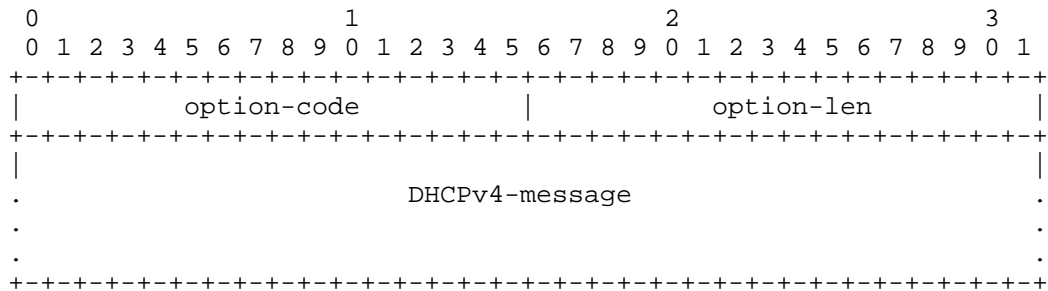


Figure 4: DHCPv4 Message option Format

option-code OPTION_DHCPV4_MSG (TBD).

option-len Length of the DHCPv4 message.

DHCPv4-message The DHCPv4 message sent by the client or the server.
 In a DHCPv4-query message it contains a DHCPv4
 message sent by a client. In a DHCPv4-response
 message it contains a DHCPv4 message sent by a server
 in response to a client.

7.2. 4o6 Server Address Option Format

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

This option may also carry no IPv6 addresses, which instructs the client to use the All_DHCP_Relay_Agents_and_Servers multicast address as the destination address.

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

The format of the 4o6 Server Address option is:

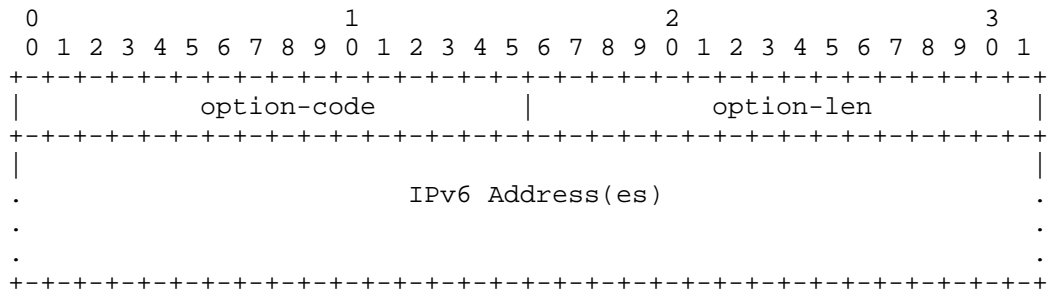


Figure 5: 4o6 Servers Address Option Format

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

8. Use of the DHCPv4-query Unicast Flag

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

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

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

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

9. DHCP 4o6 Client Behavior

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

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

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

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

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

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

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

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

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

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

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

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

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

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

10. Relay Agent Behavior

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

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

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

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

11. DHCP 4o6 Server Behavior

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

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

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

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

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

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

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

12. Security Considerations

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

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

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

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

13. IANA Considerations

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

14. Contributors List

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

15. References

15.1. Normative References

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

15.2. Informative References

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

Authors' Addresses

Qi Sun
Tsinghua University
Beijing 100084
P.R.China

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

Yong Cui
Tsinghua University
Beijing 100084
P.R.China

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

Marcin Siodelski
950 Charter Street
Redwood City, CA 94063
USA

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

Suresh Krishnan
Ericsson

Email: suresh.krishnan@ericsson.com

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

Email: ian.farrer@telekom.de

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

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

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

Abstract

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

Status of This Memo

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

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

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

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

Copyright Notice

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

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

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

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

Table of Contents

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

1. Introduction

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

2. Requirements Language

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

3. Problem Statement

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

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

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

4. Relay Agent Behavior Update

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

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

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

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

Section 20.1 of [RFC3315] states that:

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

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

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

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

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

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

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

4.2. Relaying a Message toward Server

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

4.3. Relaying a Message toward Client

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

5. Client and Server Behavior Update

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

6. Security Considerations

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

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

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

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

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

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

7. IANA Considerations

This document does not include an IANA request.

8. Contributors List

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

9. Normative References

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

Authors' Addresses

Yong Cui
Tsinghua University
Beijing 100084
P.R.China

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

Qi Sun
Tsinghua University
Beijing 100084
P.R.China

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

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

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

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

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

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

Abstract

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

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 2, 2015.

Copyright Notice

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

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

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

Table of Contents

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

1. Introduction

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

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

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

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

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

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

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

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

1.1. Overview of IPv4 Parameter Configuration Approaches

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

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

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

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

1.2. DHCPv4o6 Based Provisioning - Functional Overview

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

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

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

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

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

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

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

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

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

DHCPv4/UDP/IPv6

1.3. DHCPv6 Based Provisioning - Functional Overview

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

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

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

DHCPv6/UDP/IPv6

1.4. DHCPv6 + Stateless DHCPv4oSW Based Provisioning - Functional Overview

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

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

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

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

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

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

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

DHCPv6/UDP/IPv6

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

DHCPv4/UDP/IPv4

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

And for provisioning IPv4/IPv6 translation mechanisms:

DHCPv4/UDP/IPv6

1.5. DHCPv4oDHCPv6 Based Provisioning - Functional Overview

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

In this approach, dynamic IPv4 addressing, and/or any additional IPv4 configuration, is provided using DHCPv4 messages carried (without IPv4/UDP headers) within a new OPTION_DHCPV4_MSG DHCPv6 option.

OPTION_DHCPV4_MSG enables the client and server to send BOOTP/DHCPv4 messages verbatim across the IPv6 network. When a DHCPv4oDHCPv6 server receives a DHCPv6 request containing OPTION_DHCPV4_MSG within a DHCPV4-QUERY message, it passes it to the DHCPv4 server engine.

Likewise, the DHCPv4 server place its DHCPv4 response in the payload of OPTION_DHCPV4_MSG and puts this into a DHCPV4-RESPONSE message.

DHCPv4 messages can be carried within DHCPv6 multicast messages, using the All_DHCP_Relay_Agents_and_Servers multicast address. These can be relayed in exactly the same way as any other DHCPv6 multicasted messages.

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

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

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

DHCPv4/DHCPv6/UDP/IPv6

2. Requirements for the Solution Evaluation

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

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

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

3. Comparison of the Four Approaches

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

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

Table 1: Approach Comparison

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

3.1. DHCPv4o6 Based Provisioning

3.1.1. Pros

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

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

3.1.2. Cons

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

3.2. DHCPv6 Based Provisioning

3.2.1. Pros

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

3.2.2. Cons

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

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

3.3. DHCPv6 + Stateless DHCPv4oSW Based Provisioning

3.3.1. Pros

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

3.3.2. Cons

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

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

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

3.4. DHCPv4oDHCPv6 Based Provisioning

3.4.1. Pros

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

3.4.2. Cons

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

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

4. Conclusion

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

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

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

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

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

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

5. Transporting Unmodified DHCPv4 Messages over an IPv6 Link Layer

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

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

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

5.1. Combined Hub and DHCPv4 Relay Required Functionality

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

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

6. IANA Considerations

This document does not make any request from IANA.

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

7. Security Considerations

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

7.1. DHCPv4oIPv6

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

7.2. DHCPv6

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

7.3. DHCPv6+DHCPv4oSW

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

7.4. DHCPv4oDHCPv6

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

8. Acknowledgements

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

9. Informative References

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

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

Authors' Addresses

Branimir Rajtar
Hrvatski Telekom
Zagreb
Croatia

Email: branimir.rajtar@t.ht.hr

Ian Farrer
Deutsche Telekom AG
Bonn
Germany

Email: ian.farrer@telekom.de

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

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

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

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 19, 2014.

Copyright Notice

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

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

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

Table of Contents

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

1. Introduction

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

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

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

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

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

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

2. Requirements Language and Terminology

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

3. Security Overview of DHCPv6

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

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

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

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

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

4. Secure DHCPv6 Overview

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

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

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

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

4.1. New Components

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

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

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

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

4.2. Support for algorithm agility

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

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

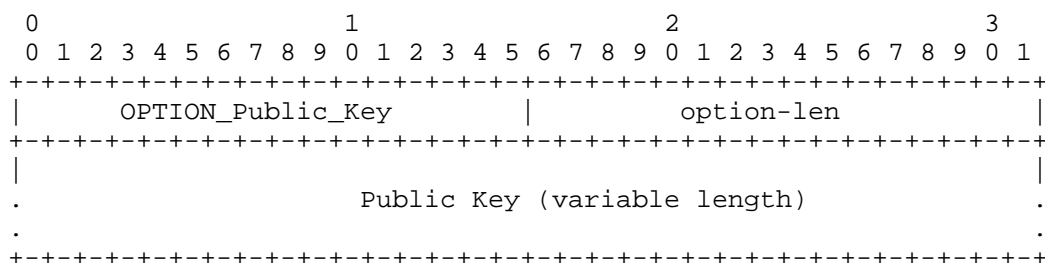
The support for algorithm agility in this document is mainly a unilateral notification mechanism from sender to recipient. If the recipient does not support the algorithm used by the sender, it cannot authenticate the message. Senders in a same administrative domain are not required to upgrade to a new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

5. Extensions for Secure DHCPv6

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

5.1. Public Key Option

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



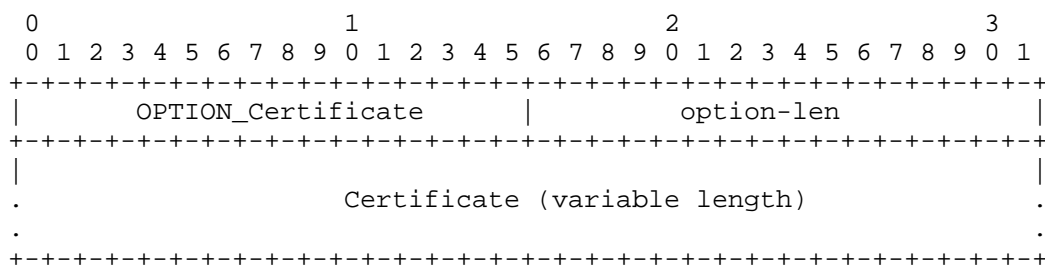
option-code OPTION_PK_PARAMETER (TBA1).

option-len Length of public key in octets.

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

5.2. Certificate Option

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



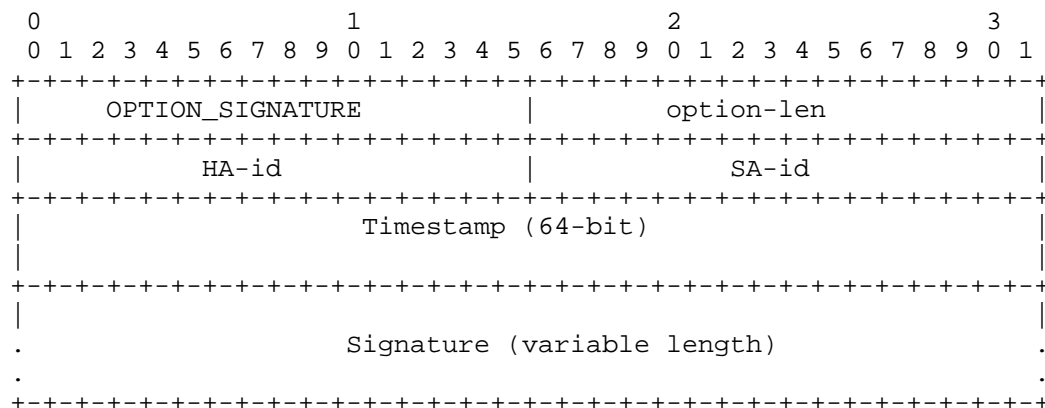
option-code OPTION_CERT_PARAMETER (TBA2).

option-len Length of certificate in octets.

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

5.3. Signature Option

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



option-code OPTION_SIGNATURE (TBA3).

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

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

SA-id Signature Algorithm id. The signature algorithm is used for computing the signature result. This design is adopted in order to provide signature algorithm agility. The value is from the Signature Algorithm for Secure DHCPv6 registry in IANA. The initial values are assigned for RSASSA-PKCS1-v1_5 is 0x0001.

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

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

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

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

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

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

6. Processing Rules and Behaviors

6.1. Processing Rules of Sender

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

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

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

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain either a the Public Key or Certificate option, which MUST constructed as explained in Section 5.1 or Section 5.2.

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain the Signature option, which MUST be constructed as explained in Section 5.3. It protects the message header and the message payload and all DHCPv6 options except for the Signature option itself and the Authentication Option. Within the Signature option the Timestamp field SHOULD be set to the current time, according to sender's real time clock.

A Relay-forward and relay-reply message MUST NOT contain any Public Key or Certificate option or Signature Option.

6.2. Processing Rules of Recipient

When receiving a DHCPv6 message, except for Relay-Forward and Relay-Reply messages, a Secure DHCPv6 enabled recipient SHOULD discard the DHCPv6 message if the Signature option is absent, or both the Public Key and Certificate option is absent, or both the Public Key and Certificate option are presented. If all three options are absent, the recipient MAY fall back the unsecure DHCPv6 model.

The recipient SHOULD first check the authority of this sender. If the sender uses a public key, the recipient SHOULD validate it by finding a match public key from the local trust public key list, which is pre-configured or recorded from previous communications. A local trust public key list is a data table maintained by the recipient. It restores public keys from all trustworthy senders. A fast search index may be created for this data table. If the sender uses certificate, the recipient SHOULD validate the sender's certificate following the rules defined in [RFC5280]. An implementation may then create a local trust certificate record.

The recipient may choose to further process the message from a sender for which no authorization information exists. By recording the key that was used by the sender, when the first time it is seen, the recipient can make a leap of faith that the sender is trustworthy. If no evidence to the contrary surfaces, the recipient can then validate the sender as trustworthy when it subsequently sees the same key used to sign messages from the same server.

At this point, the recipient has either recognized the authorization of the sender, or decided to attempt a leap of faith. The recipient MUST now authenticate the sender by verifying the Signature and checking timestamp. The order of two procedures is left as an implementation decision. It is RECOMMENDED to check timestamp first,

because signature verification is much more computationally expensive.

The signature field verification MUST show that the signature has been calculated as specified in Section 5.3. Only the messages that get through both the signature verifications and timestamp check are accepted as secured DHCPv6 messages and continue to be handled for their contained DHCPv6 options as defined in [RFC3315]. Messages that do not pass the above tests MUST be discarded or treated as unsecure messages.

The recipient MAY record the verified public key or certificate for future authentications.

Furthermore, the node that supports the verification of the Secure DHCPv6 messages MAY record the following information:

Minbits The minimum acceptable key length for public keys. An upper limit MAY also be set for the amount of computation needed when verifying packets that use these security associations. The appropriate lengths SHOULD be set according to the signature algorithm and also following prudent cryptographic practice. For example, minimum length 1024 and upper limit 2048 may be used for RSA [RSA].

A Relay-forward or Relay-reply message with any Public Key, Certificate or the Signature option is invilad. The message SHOULD be discarded silently.

6.3. Processing Rules of Relay Agent

To support Secure DHCPv6, relay agents just need to follow the same processing rules defined in [RFC3315]. There is nothing more the relay agents have to do, either verify the messages from client or server, or add any secure DHCPv6 options. Actually, be definition in this document, relay agents MUST NOT add any secure DHCPv6 options.

6.4. Timestamp Check

Recipients SHOULD be configured with an allowed timestamp Delta value, a "fuzz factor" for comparisons, and an allowed clock drift parameter. The recommended default value for the allowed Delta is 300 seconds (5 minutes); for fuzz factor 1 second; and for clock drift, 0.01 second.

Note: the Timestamp mechanism is based on the assumption that communication peers have rough synchronized clocks, with certain allowed clock drift. So, accurate clock is not necessary. If one

has a clock too far from the current time, the timestamp mechanism would not work.

To facilitate timestamp checking, each recipient SHOULD store the following information for each sender, from which at least one accepted secure DHCPv6 message is successfully verified (for both timestamp check and signature verification):

- o The receive time of the last received and accepted DHCPv6 message. This is called RDlast.
- o The time stamp in the last received and accepted DHCPv6 message. This is called TSlast.

An verified (for both timestamp check and signature verification) secure DHCPv6 message initiates the update of the above variables in the recipient's record.

Recipients MUST check the Timestamp field as follows:

- o When a message is received from a new peer (i.e., one that is not stored in the cache), the received timestamp, TSnew, is checked, and the message is accepted if the timestamp is recent enough to the reception time of the packet, RDnew:

$$-\text{Delta} < (\text{RDnew} - \text{TSnew}) < +\text{Delta}$$

After the signature verification also successes, the RDnew and TSnew values SHOULD be stored in the cache as RDlast and TSlast.

- o When a message is received from a known peer (i.e., one that already has an entry in the cache), the timestamp is checked against the previously received Secure DHCPv6 message:

$$\text{TSnew} + \text{fuzz} > \text{TSlast} + (\text{RDnew} - \text{RDlast}) \times (1 - \text{drift}) - \text{fuzz}$$

If this inequality does not hold, the recipient SHOULD silently discard the message. If, on the other hand, the inequality holds, the recipient SHOULD process the message.

Moreover, if the above inequality holds and TSnew > TSlast, the recipient SHOULD update RDlast and TSlast after the signature verification also successes. Otherwise, the recipient MUST NOT update RDlast or TSlast.

An implementation MAY use some mechanism such as a timestamp cache to strengthen resistance to replay attacks. When there is a very large number of nodes on the same link, or when a cache filling attack is

in progress, it is possible that the cache holding the most recent timestamp per sender will become full. In this case, the node MUST remove some entries from the cache or refuse some new requested entries. The specific policy as to which entries are preferred over others is left as an implementation decision.

7. Security Considerations

This document provides new security features to the DHCPv6 protocol.

Using public key based security mechanism and its verification mechanism in DHCPv6 message exchanging provides the authentication and data integrity protection. Timestamp mechanism provides anti-replay function.

The Secure DHCPv6 mechanism is based on the pre-condition that the recipient knows the public key of senders or the sender's certificate can be verified through a trust CA. It prevents DHCPv6 server spoofing. The clients may decline the DHCPv6 messages from unknown/unverified servers, which may be fake servers; or may prefer DHCPv6 messages from known/verified servers over unsigned messages or messages from unknown/unverified servers. The pre-configuration operation also needs to be protected, which is out of scope. The deployment of PKI is also out of scope.

However, when a DHCPv6 client first encounters a new public key or new unverified certificate, it can make a leap of faith. If the DHCPv6 server that used that public key or certificate is in fact legitimate, then all future communication with that DHCPv6 server can be protected by caching the public key. This does not provide complete security, but it limits the opportunity to mount an attack on a specific DHCPv6 client to the first time it communicates with a new DHCPv6 server.

Downgrade attacks cannot be avoided if nodes are configured to accept both secured and unsecured messages. A future specification may provide a mechanism on how to treat unsecured DHCPv6 messages.

[RFC6273] has analyzed possible threats to the hash algorithms used in SEND. Since the Secure DHCPv6 defined in this document uses the same hash algorithms in similar way to SEND, analysis results could be applied as well: current attacks on hash functions do not constitute any practical threat to the digital signatures used in the signature algorithm in the Secure DHCPv6.

A window of vulnerability for replay attacks exists until the timestamp expires. Secure DHCPv6 nodes are protected against replay attacks as long as they cache the state created by the message

containing the timestamp. The cached state allows the node to protect itself against replayed messages. However, once the node flushes the state for whatever reason, an attacker can re-create the state by replaying an old message while the timestamp is still valid.

Attacks against time synchronization protocols such as NTP [RFC5905] may cause Secure DHCPv6 nodes to have an incorrect timestamp value. This can be used to launch replay attacks, even outside the normal window of vulnerability. To protect against these attacks, it is recommended that Secure DHCPv6 nodes keep independently maintained clocks or apply suitable security measures for the time synchronization protocols.

8. IANA Considerations

This document defines three new DHCPv6 [RFC3315] options. The IANA is requested to assign values for these three options from the DHCP Option Codes table of the DHCPv6 Parameters registry. The three options are:

The Public Key Option (TBA1), described in Section 5.1.

The Certificate Option (TBA2), described in Section 5.2.

The Signature Option (TBA3), described in Section 5.3.

The IANA is also requested to add two new registry tables to the DHCPv6 Parameters registry. The two tables are the Hash Algorithm for Secure DHCPv6 table and the Signature Algorithm for Secure DHCPv6 table.

Initial values for these registries are given below. Future assignments are to be made through Standards Action [RFC5226]. Assignments for each registry consist of a name, a value and a RFC number where the registry is defined.

Hash Algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for Hash Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
-----	-----	-----
Reserved	0x0000	this document
SHA-1	0x0001	this document
SHA-256	0x0002	this document

Signature Algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for Signature Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
-----	-----	-----
Reserved	0x0000	this document
RSASSA-PKCS1-v1_5	0x0001	this document

9. Acknowledgements

The authors would like to thank Bernie Volz, Ted Lemon, Ralph Droms, Jari Arkko, Sean Turner, Stephen Kent, Thomas Huth, David Schumacher, Dacheng Zhang, Francis Dupont and other members of the IETF DHC working groups for their valuable comments.

This document was produced using the xml2rfc tool [RFC2629].

10. Change log [RFC Editor: Please remove]

draft-jiang-dhc-sedhcpv6-01: removed protection between relay agent and server due to complexity, following the comments from Ted Lemon, Bernie Volz. 2013-10-16.

draft-jiang-dhc-sedhcpv6-01: update according to review comments from Ted Lemon, Bernie Volz, Ralph Droms. Separated Public Key/Certificate option into two options. Refined many detailed processes. 2013-10-08.

draft-jiang-dhc-sedhcpv6-00: original version, this draft is a replacement of draft-ietf-dhc-secure-dhcpv6, which reached IESG and dead because of consideration regarding to CGA. The authors followed the suggestion from IESG making a general public key based mechanism. 2013-06-29.

11. References

11.1. Normative References

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

- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.
- [RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

11.2. Informative References

- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.
- [RFC4270] Hoffman, P. and B. Schneier, "Attacks on Cryptographic Hashes in Internet Protocols", RFC 4270, November 2005.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6273] Kukec, A., Krishnan, S., and S. Jiang, "The Secure Neighbor Discovery (SEND) Hash Threat Analysis", RFC 6273, June 2011.
- [RSA] RSA Laboratories, "RSA Encryption Standard, Version 2.1, PKCS 1", November 2002.

Authors' Addresses

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

Email: jiangsheng@huawei.com

Sean Shen
CNNIC
4, South 4th Street, Zhongguancun
Beijing 100190
P.R. China

Email: shenshuo@cnnic.cn

DHC WG
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2014

S. Jiang
B. Liu
Huawei Technologies Co., Ltd
February 14, 2014

Stateless Reconfiguration in Dynamic Host Configuration Protocol for
IPv6 (DHCPv6)
draft-jiang-dhc-stateless-reconfiguration-01

Abstract

This document defines a mechanism to propagate reconfigure messages towards stateless configured DHCPv6 clients.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. Requirements Language	4
3. Stateless Reconfiguration Use Cases	4
4. New DHCPv6 Specifications	5
4.1. Multicast Address	5
4.2. Stateless Reconfigure Message	5
5. Stateless Reconfiguration Procedure	5
5.1. Server Behavior	6
5.2. Relay Agent Behavior	7
5.3. Client Behavior	8
6. Security Considerations	8
7. IANA Considerations	8
8. Acknowledgements	9
9. References	9
9.1. Normative References	9
9.2. Informative References	9
Authors' Addresses	10

1. Introduction

[RFC3736] defines a stateless configuration procedure using DHCPv6. With it, the network configure information, such as the addresses of DNS recursive name servers, can be propagated to nodes, which have obtained their IPv6 addresses through some other mechanism. The basic scenario is that a newly online client initiates an information request to DHCPv6 server, then the server responses with requested configuration information. This mechanism is called the Stateless DHCPv6 services, because DHCPv6 servers do not maintain any dynamic state for individual clients, including the unicast addresses of clients.

However, the specification of stateless DHCPv6 service lacks a mechanism to inform these configured clients if some configuration information is changed. Transplanting Reconfigure message of [RFC3315] into stateless DHCPv6 services does not solve the issue, because in stateful DHCPv6, servers send Reconfigure messages to clients using their unicast addresses.

The lifetime option for DHCPv6 [RFC4242] assigns a lifetime to configuration information obtained through DHCPv6. At the expiration of the lifetime, the host contacts the DHCPv6 server to obtain updated configuration information. This lifetime gives the network administrator another mechanism to configure hosts with new configuration by controlling the time at which the host refreshes the list. However, such mechanism is not flexible enough: one aspect is the minimum of refresh time is 10 minutes, which is so long that it

might not be suitable for unplanned configuration changes; the other aspect is, in order to update the configuration quickly, the short time setting would cause un-necessary refresh all the time.

This document defines a mechanism to propagate a newly defined Stateless-Reconfigure message towards stateless configured DHCPv6 clients. It requests a mechanism for the DHCPv6 server to be aware of all relay agent destinations.

{Question to WG No.1} There are three potential mechanisms to create relay agent destinations on the DHCPv6 server:

a) Static configuration

Network administrators manually configure static unicast addresses of all relay agents on the DHCPv6 server.

Pros: no need to update any protocol/function implementation in relays; allows fast deployment in current network.

Cons: cost significant human management burden; error-prone, mistakenly configuring the relay addresses or leaving out some relays are expected.

b) Define a new ALL_RELAY_AGENT multicast address

The DHCPv6 server could send the stateless reconfiguration messages directly to the new multicast address.

Pros: a solid coverage of all relays.

Cons: network administrators need to maintain an all-relay-agent multicast group; all relays and DHCPv6 servers need to be updated to know the new multicast address.

c) DHCPv6 server dynamic learning

the DHCPv6 server dynamically records unicast addresses of all relay agents from client Information-request messages and maintains the relay addresses list. A keepalive mechanism is needed between relay agents and servers to track the availability of the list entries.

Pros: automatic processing without human intervene.

Cons: requires more function update to the DHCPv6 server; the keepalive mechanism requires more function/protocol burden to the whole DHCP system.

[Editor Notes] the current form of this document is based on only the first mechanism of above three. If the WG decided to change or include other mechanism, the document would be updated accordingly.

The document newly defines a new link-scope well-known all-client multicast address and a new DHCPv6 message type for stateless reconfiguration. Correspondent server behavior, agent behavior and client behavior are specified in details.

The design of new DHCPv6 elements and procedures obey the recommendations and guidance of [I-D.ietf-dhc-option-guidelines].

2. Requirements Language

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

3. Stateless Reconfiguration Use Cases

This section described scenarios where stateless reconfiguration are expected.

- Configuration error

Configuration errors, either caused by human or program, are hard to be immune in networks. Especially, human errors is identified as one of the top reasons of network failure. In stateless DHCPv6, if the administrators/program accidentally mis-configure the parameters (e.g. DNS), then significant network failure would be expected. Current protocols just lack the ability to eliminate the configuration errors when such accident happens. The hosts configured with wrong parameters can only wait until the wrong parameters lifetime expired then to refresh them. This would not be acceptable especially when the lifetime was long. The stateless reconfiguration mechanism could be highly expected in this scenario.

- Emergent event

The network needs to initially update the already configured parameters within a short period due to some emergent events; and waiting the clients to refresh the parameters according to the lifetime is just un-acceptable. These scenarios would also require stateless reconfiguration.

4. New DHCPv6 Specifications

This section define new DHCPv6 elements requested by the stateless configuration mechanism, including multicast address constant, and message type.

4.1. Multicast Address

`ALL_CLIENT_MULTICAST_ADDRESS` (FF02::xxxx, TBD1) A link-scoped multicast address used by a DHCPv6 server or relay agent to communicate with neighboring (i.e., on-link) clients. All clients are members of this multicast group

4.2. Stateless Reconfigure Message

A new Stateless-Reconfigure message, which is mainly based on server to clients advertise model, is defined in order to distinguish from the existing Reconfigure message, which is mainly based on server/client one-to-one model.

[Editor Notes] According to the results of Qestion 2 and Question 4 (in Section 5 & 7 below), there might be two new messages needed. Current document uses the alternative of one new message.

`STATELESS-RECONFIGURE-TRIGGER` Message type value is TBD2. It follows the message format specification, defined in Section 6 of [RFC3315]. A server sends a Stateless-Reconfigure message to a client to inform the client that the server has new or updated configuration parameters, and that the client is to initiate an Information-Request transaction with the server in order to receive the updation.

5. Stateless Reconfiguration Procedure

{Question to WG No.2} There could be two kind of stateless DHCPv6 reconfiguration modes as the following, which one is proper? Or we should support both?

- Trigger mode

The server sends out a multicast Stateless-Reconfiguration message to `ALL_CLIENT_MULTICAST_ADDRESS`. As response, every client is requested to initiate an Information-Request message back to the server. The server can then inform the changed configuration information to clients.

This mode is similar with stateful DHCPv6 reconfiguration and also provide the potential possibility that the server response to information-request differently according to various user policies.

- Push mode

The server sends out a multicast Stateless-Reconfiguration message to ALL_CLIENT_MULTICAST_ADDRESS to directly advertise new configuration to the clients. The clients then update the parameters accordingly.

Trigger mode requires every client to initial individual request to servers. This is reasonable for the stateful information that need to be maintained and tracked in the servers, e.g. clients' IP addresses. But for the stateless information shared among clients (such as DNS), it might not necessary. Some resource constrained networks (e.g. a 802.15.4e/g based mesh network) might have efficiency problem with the trigger mode. These scenarios might significantly benefit from the push mode stateless reconfiguration mechanism. However, push mode seems breaking the traditional behavior model of DHCP. Whether it is a good break needs further discussion.

[Editor Notes] the current form of this document is based on triggering client information-request model, which complies the traditional behavior model of DHCPv6. If the WG chooses to directly advertise new configuration, the document would be updated accordingly.

5.1. Server Behavior

When the network configuration information on a stateless DHCPv6 server changes, the server creates and transmit a new Stateless-Reconfigure message towards all clients following the below steps:

- o The server sets the "msg-type" field to STATELESS-RECONFIGURE. The server sets the transaction-id field to 0. The server MUST include a Server Identifier option containing its DUID in the Reconfigure message.
- o The server MAY include an Option Request option to inform the client of what information has been changed or new information that has been added.
- o The server MUST NOT include a Reconfigure Message option (defined in section 22.19 of [RFC3315]), which is mandated in Reconfigure message to indicate the client to respond a Renew or an Information-Request message. It is because there is only one possible response on the client follow a Stateless-Reconfigure message - an Information-request message.

- o The server MUST NOT include any other options in the Reconfigure except as specifically allowed in the definition of individual options.
- o The server sends Stateless-Reconfigure message to its direct local link using ALL_CLIENT_MULTICAST_ADDRESS.
- o Simultaneously, the server uses a Relay-Reply message (as described in Section 20.3 of [RFC3315]) to send the Stateless-Reconfigure message to all relay agents in its static relay-agent-destination record using the unicast address of these relay agents. The peer-address of the Relay-Reply message MUST be filled by Relay-Reply message ALL_CLIENT_MULTICAST_ADDRESS.

Notes: since there is no previous Relay-Forward message that went through multiple relay agents and the server has to send the Relay-Reply message through the return same path, the server should be able to send the Relay-Reply message to the relay agent that direct connects with clients. Consequently, the Relay-Reply message SHOULD NOT contain another Relay-Reply message.

The below is an example of a typical Relay-Reply message that contains a Stateless-Reconfigure message:

```
msg-type: RELAY-REPLY
hop-count: 0
link-address: 0
peer-address: ALL_CLIENT_MULTICAST_ADDRESS
Relay Message option: <Stateless-Reconfigure message>
```

Servers MUST discard any received Stateless-Reconfigure messages.

5.2. Relay Agent Behavior

The relay agent extracts the Stateless-Reconfigure message from the Relay Message option and relays it to all clients. If the relay agent is attached to multiple links, it MUST broadcast the Stateless-Reconfigure message on every links. This behavior is compliance with normal behavior of relaying a Relay-reply message, defined in Section 20.2 of [RFC3315].

Relay agents MUST discard any received Stateless-Reconfigure messages. By design, relay agents do not process any directly received Stateless-Reconfigure messages.

The result is that the relay agent sends out a Stateless-Reconfigure message towards all client on the local link using ALL_CLIENT_MULTICAST_ADDRESS.

5.3. Client Behavior

Clients MUST discard any Stateless-Reconfigure messages that meets any of the following conditions:

- o the message was not multicast to the client using ALL_CLIENT_MULTICAST_ADDRESS.
- o the message does not include a Server Identifier option.
- o the message contains a Reconfigure Message Option, defined in Section 22.19 of [RFC3315].

Upon receipt of a valid Stateless-Reconfigure message, after a random delay time, the client responds with an Information-request message. The random delay time is designed to avoid congested Information-request on the server. While the transaction is in progress, the client silently discards any Stateless-Reconfigure messages it receives.

{Question to WG No.3} Should we define a maximum time of random delay time? If yes, should it come from server by a new option?

6. Security Considerations

Malicious server sends Stateless Reconfigure message to cause all clients response. There is the risk of denial of service attacks against DHCP clients and server. {Current auth mechanism cannot work in this broadcast model, server public key model maybe work.}

Since the clients response to Information-Request using the standard mechanism, defined in [RFC3315], the chance that receive wrong configuration information from malicious attackers does not raise.

7. IANA Considerations

Per this document, IANA has assigned one new well-known Multicast Address in the "IPv6 Multicast Address Space Registry" registry (currently located at <http://www.iana.org/assignments/ipv6-multicast-addresses>) for the following attribute:

ALL_CLIENT_MULTICAST_ADDRESS: (FF02::xxxx, TBD1).

Per this document, IANA has assigned one new DHCPv6 message type in the "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)" registry (currently located at <http://www.iana.org/assignments/dhcpv6-parameters>) for the following attribute:

STATELESS-RECONFIGURE Message Type, TBD2.

{Question to WG No.4} As raised in Question 2, if we support both Trigger mode and Push mode, then there should be two kind of corresponding messages. We could use two message types to distinguish them; or use a flag in one message type. Which is better?

8. Acknowledgements

The authors would like to thanks the valuable comments made by Suresh Krishnan, Ted Lemon, Bernie Voltz and other members of DHC WG.

This document was produced using the xml2rfc tool [RFC2629].

9. References

9.1. Normative References

- [I-D.ietf-dhc-option-guidelines]
Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", draft-ietf-dhc-option-guidelines-17 (work in progress), January 2014.
- [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.
- [RFC3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, April 2004.

9.2. Informative References

- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.
- [RFC4242] Venaas, S., Chown, T., and B. Volz, "Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 4242, November 2005.

Authors' Addresses

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

Email: jiangsheng@huawei.com

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

Email: leo.liubing@huawei.com

HOMENET
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2015

D. Migault (Ed)
Orange
W. Cloetens
SoftAtHome
C. Griffiths
Dyn
R. Weber
Nominum
July 2, 2014

DHCP Options for Homenet Naming Architecture
draft-mglt-homenet-naming-architecture-dhc-options-02.txt

Abstract

CPEs are usually constraint devices with reduced network and CPU capacities. As such, a CPE hosting on the Internet the authoritative naming service for its home network may become vulnerable to resource exhaustion attacks. One way to avoid exposing CPE is to outsource the authoritative service to a third party. This third party can be the ISP or any other independent third party.

Outsourcing the authoritative naming service to a third party requires setting up an architecture which may be unappropriated for most end users. To leverage this issue, this document proposes DHCP Options so any agnostic CPE can automatically proceed to the appropriated configuration and outsource the authoritative naming service for the home network. This document shows that in most cases, these DHCP Options make outsourcing to a third party (be it the ISP or any ISP independent service provider) transparent for the end user.

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 3, 2015.

Copyright Notice

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

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

Table of Contents

1. Requirements notation	3
2. Terminology	3
3. Introduction	4
4. Protocol Overview	6
5. Securing the exchanges	8
6. DNS Zones Update Mechanisms	9
6.1. Data subjected to update	9
6.2. Master / Slave Synchronization versus DNS Update	9
6.3. Setting Master / Slave Synchronization	10
6.4. Master / Slave Synchronization: CPE / Public Authoritative Name Server Set	10
6.5. Master / Slave Synchronization: CPE / Reverse Public Authoritative Name Server Set	11
7. DNS Zone Update Data	11
7.1. DNS Homenet Zone Template	11
7.2. DNS Homenet Zone	12
8. Payload Description	12
8.1. Security Field	12
8.2. Update Field	13
8.3. DHCP Public Key Option	14
8.4. DHCP Zone Template Option	14
8.5. DHCP Public Authoritative Name Server Set Option	15
8.6. DHCP Reverse Public Authoritative Name Server Set Option	16
9. DHCP Behavior	17
9.1. DHCPv6 Server Behavior	17
9.2. DHCPv6 Client Behavior	17
9.3. DHCPv6 Relay Behavior	17
10. IANA Considerations	17
11. Security Considerations	18

11.1. DNSSEC is recommended to authenticate DNS hosted data .	18
11.2. Channel between the CPE and ISP DHCP Server MUST be secured	18
11.3. CPEs are sensitive to DoS	18
12. Acknowledgment	18
13. References	18
13.1. Normative References	18
13.2. Informational References	20
Appendix A. Scenarios and impact on the End User	20
A.1. Base Scenario	20
A.2. Third Party Registered Homenet Domain	21
A.3. Third Party DNS Infrastructure	22
Appendix B. Document Change Log	23
Authors' Addresses	24

1. Requirements notation

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

2. Terminology

- Customer Premises Equipment: (CPE) is the router providing connectivity to the home network. It is configured and managed by the end user. In this document, the CPE might also hosts services such as DHCPv6. This device might be provided by the ISP.
- Public Key: designates a public Key generated by the CPE. This key is used as an authentication credential for the CPE.
- Registered Homenet Domain: is the Domain Name associated to the home network.
- DNS Homenet Zone: is the DNS zone associated to the home network. This zone is set by the CPE and essentially contains the bindings between names and IP addresses of the nodes of the home network. In this document, the CPE does neither perform any DNSSEC management operations such as zone signing nor provide an authoritative service for the zone. Both are delegated to the Public Authoritative Server. The CPE synchronizes the DNS Homenet Zone with the Public Authoritative Server via a hidden master / slave architecture. The Public Authoritative Server might use specific servers for the synchronization of the DNS Homenet Zone: the Public Authoritative Name Server Set.

- DNS Homenet Zone Template: The template used as a basis to generate the DNS Homenet Zone.
- DNS Template Server: The DNS server that hosts the DNS Homenet Zone Template.
- DNS Homenet Reverse Zone: The reverse zone file associated to the DNS Homenet Zone.
- Public Authoritative Master(s): are the visible name server hosting the DNS Homenet Zone. End users' resolutions for the Homenet Domain are sent to this server, and this server is a master for the zone.
- Public Authoritative Name Server Set: is the server the CPE synchronizes the DNS Homenet Zone. It is configured as a slave and the CPE acts as master. The CPE sends information so the DNSSEC zone can be set and served.
- Reverse Public Authoritative Master(s): are the visible name server hosting the DNS Homenet Reverse Zone. End users' resolutions for the Homenet Domain are sent to this server, and this server is a master for the zone.
- Reverse Public Authoritative Name Server Set: is the server the CPE synchronizes the DNS Homenet Reverse Zone. It is configured as a slave and the CPE acts as master. The CPE sends information so the DNSSEC zone can be set and served.

3. Introduction

CPEs are usually constraint devices with reduced network and CPU capacities. As such, a CPE hosting on the Internet the authoritative naming service for its home network may become vulnerable to resource exhaustion attacks. One way to avoid exposing CPE is to outsource the authoritative service to a third party. This third party can be the ISP or any other independent third party.

Outsourcing the authoritative naming service to a third party requires setting up an architecture which may be unappropriated for most end users. To leverage this issue, this document proposes DHCP Options so any agnostic CPE can automatically proceed to the appropriated configuration and outsource the authoritative naming service for the home network. This document shows that in most cases, these DHCP Options make outsourcing to a third party (be it the ISP or any ISP independent service provider) transparent for the end user.

When the CPE is plugged, the DHCP Options described in the document enable the CPE:

- 1. To build the DNS Homenet Zone: Building the DNS Homenet Zone requires filling the zone with appropriated bindings like name / IP addresses of the different devices in the home networks. Such information can be provided for example by the DHCP Server hosted on the CPE. On the other hand, it also requires configuration parameters like the name of the Registered Domain Name associated to the home network or the Public Authoritative Master(s) the DNS Homenet Zone is outsourced to. These configuration parameters are stored in the DNS Homenet Zone Template. This document describes the DHCP Zone Template Option. This option carries a DNS Homenet Zone Template FQDN. In order to retrieve the DNS Homenet Zone Template, the CPE sends a query of type AXFR [RFC1034] [RFC5936] for the DNS Homenet Zone Template FQDN.
- 2. To upload the DNS(SEC) Homenet Zone to the appropriated server: This server is designated as the Public Authoritative Name Server Set. It is in charge of publishing the DNS(SEC) Homenet Zone on the Public Authoritative Master(s). This document describes the DHCP Public Authoritative Name Server Set Option that provides the FQDN of the appropriated server. Note that, in the document we do not consider whether the DNS(SEC) Homenet Zone is signed or not and if signed who signs it. Such questions are out of the scope of the current document.
- 3. To upload the DNS Homenet Reverse Zone to the appropriated server: This server is designated as the Reverse Public Authoritative Name Server Set. It is in charge of publishing the DNS Homenet Reverse Zone on the Reverse Public Authoritative Master(s). This document describes the DHCP Reverse Public Authoritative Name Server Set Option that provides the FQDN of the appropriated server. Similarly to item 2., we do not consider in this document if the DNS Homenet Reverse Zone is signed or not, and if signed who signs it.
- 4. To provide authentication credential (a public key) to the DHCP Server: Information stored in the DNS Homenet Zone Template, the DNS(SEC) Homenet Zone and DNS Homenet Reverse Zone belongs to the CPE, and only the CPE should be able to update or upload these zones. To authenticate the CPE, this document defines the DHCP Public Key Option. This option is sent by the CPE to the DHCP Server and provides the Public Key the CPE uses to authenticate itself. The DHCP Server is then responsible to provide the Public Key to the various DNS servers.

As a result, the DHCP Options described in this document enable an agnostic CPE to outsource its naming infrastructure without any configuration from the end user. The main reason no configuration is required by the end user is that there are privilege links first between the CPE and the DHCP Server and then between the DHCP Server and the various DNS servers (DNS Homenet Zone Server, the Reverse Public Authoritative Name Server Set, Public Authoritative Name Server Set). This enables the CPE to send its authentication credentials (a Public Key) to the DHCP Server that in turn forward it to the various DNS servers. With the authentication credential on the DNS servers set, the CPE is able to update the various zones in a secure way.

If the DHCP Server cannot provide the public key to one of these servers (most likely the Public Authoritative Name Server Set) and the CPE needs to interact with the server, then, the end user is expected to provide it manually or using other mechanisms. Such mechanisms are outside the scope of this document. In that case, the authentication credentials need to be provided every time the key is modified. Appendix A provides more details on how different scenarios impact the end users.

4. Protocol Overview

This section illustrates how a CPE configures its naming infrastructure to outsource its authoritative naming service. All configurations and settings are performed using DHCP Options. In this section, for the sake of simplicity, we consider that the DHCP Server is able to communicate to the various DNS servers and provide them the public key associated to the CPE. Once each server got the credentials, the CPE can proceed to updates in a authenticated and secure way.

This scenario has been chosen as it is believed to be the most popular scenario. This document does not ignore that scenarios where the DHCP Server does not have privilege relations with the Public Authoritative Name Server Set must be considered. These cases are discussed latter in Appendix A. Such scenario does not necessarily require configuration for the end user and can also be Zero Config.

The scenario is represented in Figure 1.

- 1: The CPE provides its Public Key to the DHCP Server using a DHCP Public Key Option (OPTION_PUBLIC_KEY) and sends a DHCP Option Request Option (ORO) for the DHCP Zone Template Option (OPTION_DNS_ZONE_TEMPLATE), the DHCP Public Authoritative Name Server Set Option (OPTION_NAME_SERVER_SET) and the DHCP Reverse

Public Authoritative Name Server Set Option
(OPTION_REVERSE_NAME_SERVER_SET).

- 2: The DHCP Server makes the Public Key available to the DNS servers, so the CPE can secure its DNS transactions. Note that the Public Key alone is not sufficient to perform the authentication and the key should be, for example, associated with an identifier, or the concerned domain name. How the binding is performed is out of scope of the document. It can be a centralized database or various bindings may be sent to the different servers. Figure 1 represents the specific case where the DHCP Server forwards the set (Public Key, Zone Template FQDN) to the DNS Template Server, the set (Public Key, IPv6 subnet) to the Reverse Public Authoritative Name Server Set and the set (Public Key, Registered Homenet Domain) to the Public Authoritative Name Server Set.
- 3.: The DHCP Server responds to the CPE with the requested DHCP Options, i.e. the DHCP Public Key Option (OPTION_PUBLIC_KEY), DHCP Zone Template Option (OPTION_DNS_ZONE_TEMPLATE), DHCP Public Authoritative Name Server Set Option (OPTION_NAME_SERVER_SET), DHCP Reverse Public Authoritative Name Server Set Option (OPTION_REVERSE_NAME_SERVER_SET).
- 4.: Upon receiving the DHCP Zone Template Option (OPTION_DNS_ZONE_TEMPLATE), the CPE performs an AXFR DNS query for the Zone Template FQDN. The exchange is secured according to the security protocols defined in the Security field of the DHCP option. Once the CPE has retrieved the DNS Zone Template, the CPE can build the DNS Homenet Zone and the DNS Homenet Reverse Zone. Eventually the CPE signs these zones.
- 5.: Once the DNS(SEC) Homenet Reverse Zone has been set, the CPE uploads the zone to the Reverse Public Authoritative Name Server Set. The DHCP Reverse Public Authoritative Name Server Set Option (OPTION_REVERSE_NAME_SERVER_SET) provides the Reverse Public Authoritative Name Server Set FQDN as well as the upload method, and the security protocol to secure the upload.
- 6.: Once the DNS(SEC) Homenet Zone has been set, the CPE uploads the zone to the Public Authoritative Name Server Set. The DHCP Public Authoritative Name Server Set Option (OPTION_NAME_SERVER_SET) provides the Public Authoritative Name Server Set FQDN as well as the upload method and the security protocol to secure the upload.

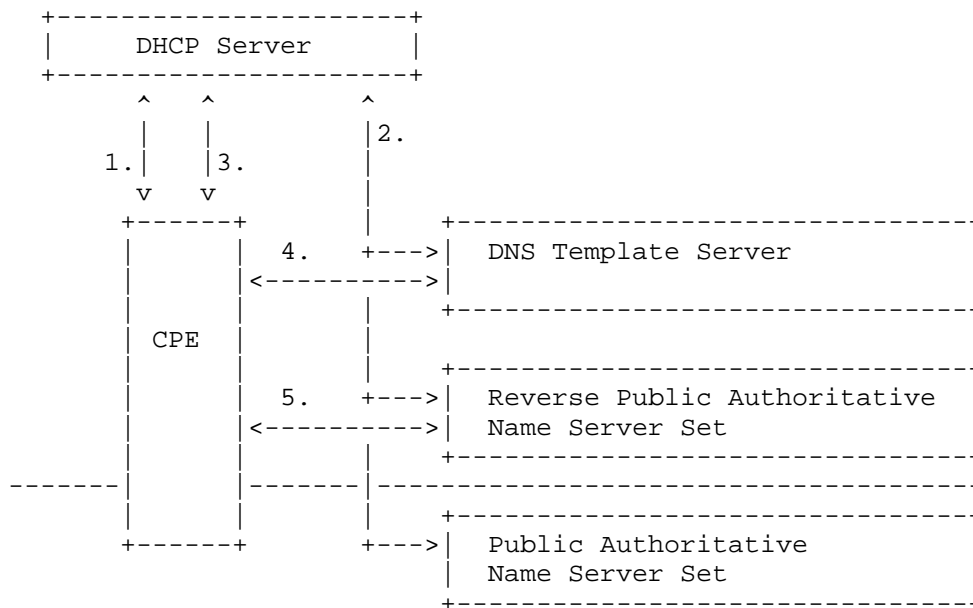


Figure 1: Protocol Overview

5. Securing the exchanges

Multiple protocols like IPsec [RFC4301] or TLS / DTLS [RFC5246] / [RFC6347] may be used to secure DNS transactions between the CPE and the DNS servers. This document restricts the scope of security protocols to those that have been designed specifically for DNS. This includes DNSSEC [RFC4033], [RFC4034], [RFC4035] that authenticates and provides integrity protection of DNS data, TSIG [RFC2845], [RFC2930] that use a shared secret to secure a transaction between two end points and SIG(0) [RFC2931] authenticates the DNS packet exchanged.

The key issue with TSIG is that a shared secret must be negotiated between the CPE and the server. On the other end, TSIG performs symmetric cryptography which is light in comparison with asymmetric cryptography used by SIG(0). As a result, over large zone transfer, TSIG may be preferred to SIG(0).

This document does not provides means to distribute shared secret for example using a specific DHCP Option. The only assumption made is that the CPE generates or is assigned a public key.

As a result, when the document specifies the transaction is secured with TSIG, it means that either the CPE and the DNS Server have been

manually configured with a shared secret, or the shared secret has been negotiated using TKEY [RFC2930], and the TKEY exchanged are secured with SIG(0).

Exchange with the DNS Template Server to retrieve the DNS Homenet Zone Template may be protected by SIG(0), TSIG or DNSSEC. When DNSSEC is used, it means the DNS Template Server only provides integrity protection, and does not necessarily prevents someone else to query the DNS Homenet Zone Template. In addition, DNSSEC is only a way to protect the communication of AXFR queries, in other words, DNSSEC cannot be used to secure updates. If DNSSEC is used to provide integrity protection for the AXFR response, the CPE should proceed to the DNSSEC signature checks. If signature check fails, it MUST reject the response. If the signature check succeeds, the CPE removes all DNSSEC related RRsets (DNSKEY, RRSIG, NSEC* ...) before building the DNS Homenet Zone. In fact, these DNSSEC related fields are associated to the DNS Homenet Zone Template and not the DNS Homenet Zone.

Any update exchange should use SIG(0) or TSIG to authenticate the exchange.

6. DNS Zones Update Mechanisms

6.1. Data subjected to update

The CPE is likely to update various DNS contents:

- DNS Homenet Zone Template: may be updated by the CPE if the configuration of the zone may be changed. This can include additional Public Authoritative Master(s), a different Registered Homenet Domain as the one initially proposed, or a redirection to another domain.
- DNS Homenet Reverse Zone: may be updated every time a new device is connected or dis-connected.
- DNS Homenet Zone: may be updated every time a new device is connected, dis-connected.

6.2. Master / Slave Synchronization versus DNS Update

As updates only concern DNS zones, this document only considers DNS update mechanisms such as DNS update [RFC2136] [RFC3007] or a master / slave synchronization.

The DNS Homenet Zone Template can only be updated with DNS update. The reason is that the DNS Homenet Zone Template contains static configuration data that is not expected to evolve over time.

The DNS Homenet Reverse Zone and the DNS Homenet Zone can be updated either with DNS update or using a master / slave synchronization. As these zones may be large, with frequent updates, we recommend to use the master / slave architecture as described in [I-D.mglt-homenet-front-end-naming-delegation]. The master / slave mechanism is preferred as it better scales and avoids DoS attacks: First the master notifies the slave the zone must be updated, and leaves the slave to proceed to the update when possible. Then, the NOTIFY message sent by the master is a small packet that is less likely to load the slave. At last, the AXFR query performed by the slave is a small packet sent over TCP (section 4.2 [RFC5936]) which makes unlikely the slave to perform reflection attacks with a forged NOTIFY. On the other hand, DNS updates can use UDP, packets require more processing than a NOTIFY, and they do not provide the server the opportunity to post-pone the update.

6.3. Setting Master / Slave Synchronization

The master / slave architecture is described in [I-D.mglt-homenet-front-end-naming-delegation]. The CPE is configured as a master whereas the DNS Server is configured as a slave. The DNS Server represents the Public Authoritative Name Server Set or the Reverse Public Authoritative Name Server Set.

When the CPE is plugged its IP address may be unknown to the slave. The section details how the CPE or master communicate the necessary information to set up the slave.

In order to set the master / slave configuration, both master and slaves must agree on 1) the zone to be synchronized, 2) the IP address used by both master and slave. In this document we assume that synchronization is performed on both side on port 53.

[QUESTION Do we have to consider different port of port 53 is fine. I guess it is fine.]

6.4. Master / Slave Synchronization: CPE / Public Authoritative Name Server Set

The CPE knows the zone to be synchronized by reading the Registered Homenet Domain in the DNS Homenet Zone Template provided by the DHCP Zone Template Option (OPTION_DNS_ZONE_TEMPLATE). The IP address of the slave is provided by the DHCP Public Authoritative Name Server Set Option (OPTION_NAME_SERVER_SET).

The Public Authoritative Name Server Set has been configured with the Registered Homenet Domain and the Public Key that identifies the CPE. The only thing missing is the IP address of the CPE. This IP address is provided by the CPE by sending a NOTIFY [RFC1996].

When the CPE has built its DNS Homenet Zone, it sends a NOTIFY message to the Public Authoritative Name Server Sets. Upon receiving the NOTIFY message, the slave reads the Registered Homenet Domain and checks the NOTIFY is sent by the authorized master. This can be done using the shared secret (TSIG) or the public key (SIG(0)). Once the NOTIFY has been authenticated, the Public Authoritative Name Server Sets might consider the source IP address of the NOTIFY query to configure the masters attributes.

6.5. Master / Slave Synchronization: CPE / Reverse Public Authoritative Name Server Set

The CPE knows the zone to be synchronized by looking at its assigned prefix. The IP address of the slave is provided by the DHCP Reverse Public Authoritative Name Server Set Option (OPTION_REVERSE_NAME_SERVER_SET).

Configuration of the slave is performed as illustrated in Section 6.4.

7. DNS Zone Update Data

7.1. DNS Homenet Zone Template

The DNS Homenet Zone Template contains at least the related fields of the Public Authoritative Master(s) as well as the Homenet Registered Domain, that is SOA, and NS fields. This template might be generated automatically by the owner of the DHCP Server. For example, an ISP might provide a default Homenet Registered Domain as well as default Public Authoritative Master(s). This default settings should provide the CPE the necessary pieces of information to set the homenet naming architecture.

If the DNS Homenet Zone Template is not subject to modifications or updates, the owner of the template might only use DNSSEC to enable integrity check.

The DNS Homenet Zone Template might be subject to modification by the CPE. The advantage of using the standard DNS zone format is that standard DNS update mechanism can be used to perform updates. These updates might be accepted or rejected by the owner of the DNS Homenet Zone Template. Policies that defines what is accepted or rejected is out of scope of this document. However, in this document we assume

the Registered Homenet Domain is used as an index by the Public Authoritative Name Server Set, and SIG(0), TSIG are used to authenticate the CPE. As a result, the Registered Homenet Domain should not be modified unless the Public Authoritative Name Server Set can handle with it.

7.2. DNS Homenet Zone

The DNS Homenet Zone might be generated from the DNS Homenet Zone Template. How the DNS Homenet Zone is generated is out of scope of this document. In some cases, the DNS Homenet Zone might be the exact copy of the DNS Homenet Zone Template. In other cases, it might be generated from the DNS Homenet Zone Template with additional RRsets. In some other cases, the DNS Homenet Zone might be generated without considering the DNS Homenet Zone Template, but only considering specific configuration rules.

In the current document the CPE only sets a single zone that is associated with one single Homenet Registered Domain. The domain might be assigned by the owner of the DNS Homenet Zone Template. This constrain does not prevent the CPE to use multiple domain names. How additional domains are considered is out of scope of this document. One way to handle these additional zones is to configure static redirections to the DNS Homenet Zone using CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsextn-cname-dname].

8. Payload Description

8.1. Security Field

The Security Field of the DHCP Option is represented in Figure 2. It indicates the security mechanism supported by the DNS Server. One of these mechanism MUST be chosen by the CPE in order to perform a transaction with the DNS server. See Section 5 for more details.

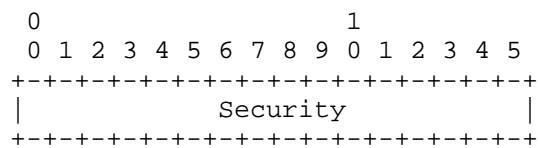


Figure 2: Security Field

- DNS (Bit 0): indicates, when set to 1, that DNS without any security extension is supported.

- DNSSEC (Bit 1): indicates, when set to 1, that DNSSEC provides integrity protection. This can only be used for read operations like retrieving the DNS Homenet Zone Template.
- SIG(0) (Bit 2): indicates, when set to 1, that transaction protected by SIG(0) are supported.
- TSIG (Bit 3): indicates, when set to 1, that transaction using TSIG is supported. Note that if a shared secret has not been previously negotiated between the two party, it should be negotiated using TKEY. The TKEY exchanges MUST be protected with SIG(0) even though SIG(0) is not supported.
- Remaining Bits (Bit 4-15): MUST be set to 0 by the DHCP Server and ignored by the DHCP Client.

A Security field with all bits set to zero indicates the operation is not permitted. The Security field may be set to zero when updates operations are not permitted for the DNS Homenet Template. In any other case this is an error.

8.2. Update Field

The Update Field of the DHCP Option is represented in Figure 3. It indicates the update mechanism supported by the DNS server. See Section 6 for more details.

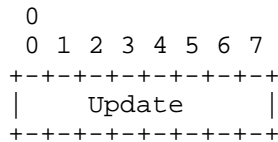


Figure 3: Update Field

- Master / Slave (Bit 0): indicates, when set to 1, that DNS Server supports data synchronization using a Master / Slave mechanism.
- DNS Update (Bit 1): indicates, when set to 1, that DNS Server supports data synchronization using DNS Updates.
- Remaining Bits (Bit 2-7): MUST be set to 0 by the DHCP Server and ignored by the DHCP Client.

8.3. DHCP Public Key Option

The DHCP Public Key Option (OPTION_PUBLIC_KEY) indicates the Public Key that is used to authenticate the CPE.

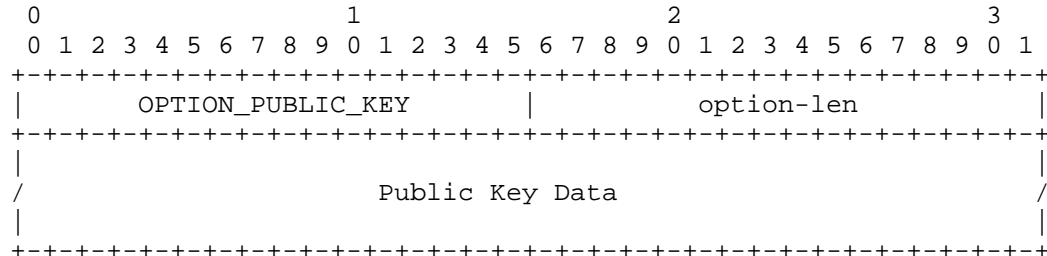


Figure 4: DHCP Public Key Option

- OPTION_PUBLIC_KEY (variable): the option code for the DHCP Public Key Option.
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Public Key Data: contains the Public Key. The format is the DNSKEY RDATA format as defined in [RFC4034].

8.4. DHCP Zone Template Option

The DHCP Zone Template Option (OPTION_DNS_ZONE_TEMPLATE) Option indicates the CPE how to retrieve the DNS Homenet Zone Template. It provides a FQDN the CPE SHOULD query with a DNS query of type AXFR. The option also specifies which security protocols are available on the authoritative server. DNS Homenet Zone Template update, if permitted MUST use the DNS Update mechanism.

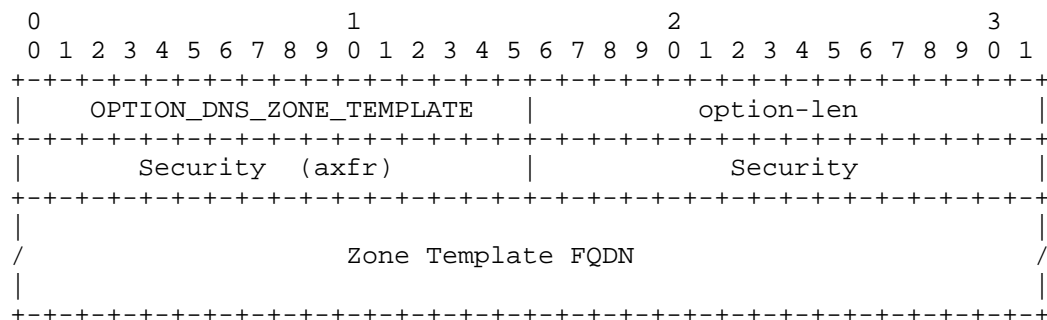


Figure 5: DHCP Zone Template Option

- `OPTION_DNS_ZONE_TEMPLATE` (variable): the option code for the DHCP Zone Template Option.
- `option-len` (16 bits): length in octets of the option-data field as described in [RFC3315].
- `Security (axfr)` (16 bits): defines which security protocols are supported by the DNS server. This field concerns the AXFR and consultation queries, not the update queries. See Section 8.1 for more details.
- `Security` (16 bits): defines which security protocols are supported by the DNS server. This field concerns the update. See Section 8.1 for more details.
- `Zone Template FQDN` (variable): the FQDN of the DNS server hosting the DNS Homenet Zone Template.

8.5. DHCP Public Authoritative Name Server Set Option

The DHCP Public Authoritative Name Server Set Option (`OPTION_NAME_SERVER_SET`) provides information so the CPE can upload the DNS Homenet Zone to the Public Authoritative Name Server Set. Finally, the option provides the security mechanisms that are available to perform the upload. The upload is performed via a DNS master / slave architecture or DNS updates.

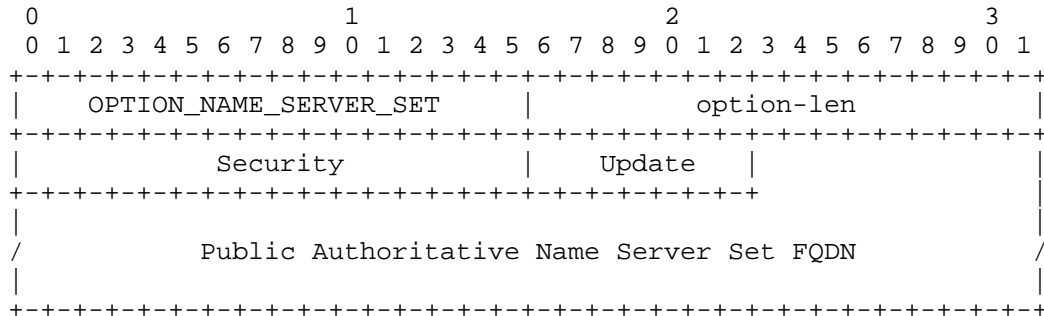


Figure 6: DHCP Public Authoritative Name Server Set Option

- `OPTION_NAME_SERVER_SET` (16 bits): the option code for the DHCP Public Authoritative Name Server Set Option.
- `option-len` (16 bits): length in octets of the option-data field as described in [RFC3315].

- Security (16 bits): defines which security protocols are supported by the DNS server. See Section 8.1 for more details.
- Update (8 bits): defines which update mechanisms are supported by the DNS server. See Section 6 for more details.
- Public Authoritative Name Server Set FQDN (variable): the FQDN of the Public Authoritative Name Server Set.

8.6. DHCP Reverse Public Authoritative Name Server Set Option

The DHCP Reverse Public Authoritative Name Server Set Option (OPTION_REVERSE_NAME_SERVER_SET) provides information so the CPE can upload the DNS Homenet Zone to the Public Authoritative Name Server Set. The option provides the security mechanisms that are available to perform the upload. The upload is performed via a DNS master / slave architecture or DNS updates.

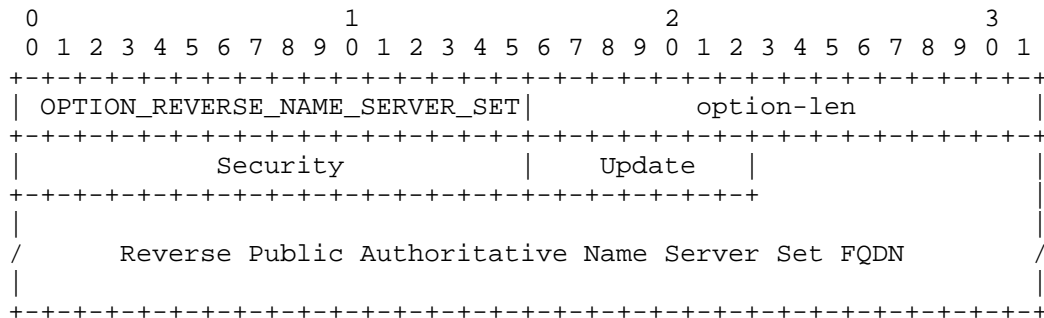


Figure 7: DHCP Reverse Public Authoritative Name Server Set Option

- OPTION_REVERSE_NAME_SERVER_SET (16 bits): the option code for the DHCP Reverse Public Authoritative Name Server Set Option.
- option-len (16 bits): length in octets of the option-data field as described in [RFC3315].
- Security (16 bits): defines which security protocols are supported by the DNS server. See Section 8.1 for more details.
- Update (8 bits): defines which update mechanisms are supported by the DNS server. See Section 6 for more details.
- Reverse Public Authoritative Name Server Set FQDN (variable): The FQDN of the Reverse Public Authoritative Name Server Set.

9. DHCP Behavior

9.1. DHCPv6 Server Behavior

The DHCP Server sends the DHCP Zone Template Option (OPTION_DNS_ZONE_TEMPLATE), DHCP Public Authoritative Name Server Set Option (OPTION_NAME_SERVER_SET), DHCP Reverse Public Authoritative Name Server Set Option (OPTION_REVERSE_NAME_SERVER_SET) upon request by the DHCP Client.

The DHCP Server MAY receive a DHCP Public Key Option (OPTION_PUBLIC_KEY) from the CPE. Upon receipt of this DHCP Option, the DHCP Server is expected to communicate this credential to the available DNS Servers like the DNS Template Server, the Public Authoritative Name Server Set and the Reverse Public Authoritative Name Server Set.

9.2. DHCPv6 Client Behavior

The DHCP Client MAY send a DHCP Public Key Option (OPTION_PUBLIC_KEY) to the DHCP Server. This Public Key authenticates the CPE.

The DHCP Client sends a DHCP Option Request Option (ORO) with the necessary DHCP options.

A CPE SHOULD only send the an ORO request for DHCP Options it needs or for information that needs to be up-to-date.

Upon receiving a DHCP option described in this document, the CPE SHOULD retrieve or update DNS zones using the associated security and update protocols.

9.3. DHCPv6 Relay Behavior

DHCP Relay behavior are not modified by this document.

10. IANA Considerations

The DHCP options detailed in this document is:

- OPTION_DNS_ZONE_TEMPLATE: TBD
- OPTION_NAME_SERVER_SET: TBD
- OPTION_REVERSE_NAME_SERVER_SET: TBD
- OPTION_PUBLIC_KEY: TBD

11. Security Considerations

11.1. DNSSEC is recommended to authenticate DNS hosted data

It is recommended that the (Reverse) DNS Homenet Zone is signed with DNSSEC. The zone may be signed by the CPE or by a third party. We recommend the zone to be signed by the CPE, and that the signed zone is uploaded.

11.2. Channel between the CPE and ISP DHCP Server MUST be secured

The document considers that the channel between the CPE and the ISP DHCP Server is trusted. More specifically, the CPE is authenticated and the exchanged messages are protected. The current document does not specify how to secure the channel. [RFC3315] proposes a DHCP authentication and message exchange protection, [RFC4301], [RFC5996] propose to secure the channel at the IP layer.

In fact, the channel MUST be secured because the CPE provides authentication credentials. Unsecured channel may result in CPE impersonation attacks.

11.3. CPEs are sensitive to DoS

CPE have not been designed for handling heavy load. The CPE are exposed on the Internet, and their IP address is publicly published on the Internet via the DNS. This makes the Home Network sensitive to Deny of Service Attacks. The resulting outsourcing architecture is described in [I-D.mglt-homenet-front-end-naming-delegation]. This document shows how the outsourcing architecture can be automatically set.

12. Acknowledgment

We would like to thank Tomasz Mrugalski, Marcin Siodelski and Bernie Volz for their comments on the design of the DHCP Options. We would also like to thank Mark Andrews, Andrew Sullivan and Lorenzo Colliti for their remarks on the architecture design. The designed solution has been largely been inspired by Mark Andrews's document [I-D.andrews-dnsop-pd-reverse] as well as discussions with Mark.

13. References

13.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

- [RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, August 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.
- [RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.
- [RFC2930] Eastlake, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, September 2000.
- [RFC2931] Eastlake, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, September 2000.
- [RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, November 2000.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

- [RFC5936] Lewis, E. and A. Hoenes, "DNS Zone Transfer Protocol (AXFR)", RFC 5936, June 2010.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.
- [RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, June 2012.

13.2. Informational References

- [I-D.andrews-dnsop-pd-reverse]
Andrews, M., "Automated Delegation of IP6.ARPA reverse zones with Prefix Delegation", draft-andrews-dnsop-pd-reverse-02 (work in progress), November 2013.
- [I-D.mglt-homenet-front-end-naming-delegation]
Migault, D., Cloetens, W., Griffiths, C., and R. Weber, "IPv6 Home Network Naming Delegation", draft-mglt-homenet-front-end-naming-delegation-03 (work in progress), October 2013.
- [I-D.sury-dnsexst-cname-dname]
Sury, O., "CNAME+DNAME Name Redirection", draft-sury-dnsexst-cname-dname-00 (work in progress), April 2010.

Appendix A. Scenarios and impact on the End User

This section details various scenarios and discuss their impact on the end user.

A.1. Base Scenario

The base scenario is the one described in Section 4. It is typically the one of an ISP that manages the DHCP Server, and all DNS servers.

The end user subscribes to the ISP (foo), and at subscription time registers for example.foo as its Registered Homenet Domain example.foo. Since the ISP knows the Registered Homenet Domain and the Public Authoritative Master(s) the ISP is able to build the DNS Homenet Zone Template.

The ISP manages the DNS Template Server, so it is able to load the DNS Homenet Zone Template on the DNS Template Server.

When the CPE is plugged (at least the first time), it provides its Public Key to the DHCP Server. In this scenario, the DHCP Server and the DNS Servers are managed by the ISP so the DHCP Server can provide authentication credentials of the CPE to enable secure authenticated transaction between the CPE and these DNS servers. More specifically, credentials are provided to:

- Public Authoritative Name Server Set
- Reverse Public Authoritative Name Server Set
- DNS Template Server

The CPE can update the zone using DNS update or a master / slave configuration in a secure way.

The main advantage of this scenario is that the naming architecture is configured automatically and transparently for the end user.

The drawbacks are that the end user uses a Registered Homenet Domain managed by the ISP and that it relies on the ISP naming infrastructure.

A.2. Third Party Registered Homenet Domain

This section considers the case when the end user wants its home network to use example.com as a Registered Homenet Domain instead of example.foo that has been assigned by the ISP. We also suppose that example.com is not managed by the ISP.

This can also be achieved without any configuration. When the end user buys the domain name example.com, it may request to redirect the name example.com to example.foo using static redirection with CNAME [RFC2181], [RFC1034], DNAME [RFC6672] or CNAME+DNAME [I-D.sury-dnsex-cname-dname].

This configuration is performed once when the domain name example.com is registered. The only information the end user needs to know is the domain name assigned by the ISP. Once this configuration is done no additional configuration is needed anymore. More specifically, the CPE may be changed, the zone can be updated as in Appendix A.1 without any additional configuration from the end user.

The main advantage of this scenario is that the end user benefits from the Zero Configuration of the Base Scenario Appendix A.1. Then, the end user is able to register for its home network an unlimited number of domain names provided by an unlimited number of different third party providers.

The drawback of this scenario may be that the end user still rely on the ISP naming infrastructure. Note that the only case this may be inconvenient is when the DNS Servers provided by the ISPs results in high latency.

A.3. Third Party DNS Infrastructure

This scenario considers that the end user uses example.com as a Registered Homenet Domain, and does not want to rely on the authoritative servers provided by the ISP.

In this section we limit the outsourcing to the Public Authoritative Name Server Set and Public Authoritative Master(s) to a third party. All other DNS Servers DNS Template Server, Reverse Public Authoritative Master(s) and Reverse Public Authoritative Name Server Set remain managed by the ISP. The reason we consider that Reverse Public Authoritative Masters(s) and Reverse Public Authoritative Name Server Set remains managed by the ISP are that the prefix is managed by the ISP, so outsourcing these resources requires some redirection agreement with the ISP. More specifically the ISP will need to configure the redirection on one of its Reverse DNS Servers. That said, outsourcing these resources is similar as outsourcing Public Authoritative Name Server Set and Public Authoritative Master(s) to a third party. Similarly, the DNS Template Server can be easily outsourced as detailed in this section

Outsourcing Public Authoritative Name Server Set and Public Authoritative Master(s) requires:

- 1) Updating the DNS Homenet Zone Template: this can be easily done as detailed in Section 6 as the DNS Template Server is still managed by the ISP. Such modification can be performed once by any CPE. Once this modification has been performed, the CPE can be changed, the Public Key of the CPE may be changed, this does not need to be done another time. One can imagine a GUI on the CPE asking the end user to fill the field with Registered Homenet Domain, optionally Public Authoritative Master(s), with a button "Configure DNS Homenet Zone Template".
- 2) Updating the DHCP Server Information. In fact the Reverse Public Authoritative Name Server Set returned by the ISP is modified. One can imagine a GUI interface that enables the end user to modify its profile parameters. Again, this configuration update is done once-for-ever.
- 3) Upload the authentication credential of the CPE, that is the Public Key of the CPE, to the third party. Unless we use specific mechanisms, like communication between the DHCP Server

and the third party, or a specific token that is plugged into the CPE, this operation is likely to be performed every time the CPE is changed, and every time the Public Key generated by the CPE is changed.

The main advantage of this scenario is that the DNS infrastructure is completely outsourced to the third party. Most likely the Public Key that authenticate the CPE need to be configured for every CPE. Configuration is expected to be CPE live-long.

Appendix B. Document Change Log

[RFC Editor: This section is to be removed before publication]

-03: Working Version Major modifications are:

- Redesigning options/scope: according to feed backs received from the IETF89 presentation in the dhc WG.
- Redesigning architecture: according to feed backs received from the IETF89 presentation in the homenet WG, discussion with Mark and Lorenzo.

-02: Working Version Major modifications are:

- Redesigning options/scope: As suggested by Bernie Volz

-01: Working Version Major modifications are:

- Remove the DNS Zone file construction: As suggested by Bernie Volz
- DHCPv6 Client behavior: Following options guide lines
- DHCPv6 Server behavior: Following options guide lines

-00: version published in the homenet WG. Major modifications are:

- Reformatting of DHCP Options: Following options guide lines
- DHCPv6 Client behavior: Following options guide lines
- DHCPv6 Server behavior: Following options guide lines

-00: First version published in dhc WG.

Authors' Addresses

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

Phone: +33 1 45 29 60 52
Email: daniel.migault@orange.com

Wouter Cloetens
SoftAtHome
vaartdijk 3 701
3018 Wijkmaal
Belgium

Email: wouter.cloetens@softathome.com

Chris Griffiths
Dyn
150 Dow Street
Manchester, NH 03101
US

Email: cgriffiths@dyn.com
URI: <http://dyn.com>

Ralf Weber
Nominum
2000 Seaport Blvd #400
Redwood City, CA 94063
US

Email: ralf.weber@nominum.com
URI: <http://www.nominum.com>

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 24, 2014

M. Boucadair
France Telecom
L. Yeh
Freelancer Technologies
A. Petrescu
CEA
October 21, 2013

Route Problem at Relay during DHCPv6 Prefix Delegation
draft-petrescu-relay-route-problem-00.txt

Abstract

The operation of a prefix delegation procedure with the DHCPv6 protocol may need route setup and maintenance at the Delegating Router, Requesting Router and on the entity on which the Relay agent is implemented. This document describes the problem of routing during DHCPv6 prefix delegation, and is illustrated by ADSL-type and cellular-type of topologies which may use Relays; we refer to section 14 of RFC 3633 which mentions the need of 'a protocol or other out of band communication to add routing information for delegated prefixes'. Based on this problem, a number of requirements from the service providers are described.

A small set of documented solutions are separately mentioned (snooping, route injection, etc.), together with their pros and cons according to a particular judgment.

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 24, 2014.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
2. Terminology	3
3. Problem and the Related Topologies	3
3.1. Problem	3
3.2. Relay vs. non-Relay Topologies	4
3.3. Relay Topologies for Large-scale Deployments	6
4. Issues and Requirements	8
5. Potential Solutions, Solution Space	10
5.1. DHCPv6 message snooping for PD-prefix	10
5.2. ietf-dhc-dhcpv6-agentopt-delegate for PD-prefix	10
5.3. draft-ietf-dhc-dhcpv6-prefix-pool-opt-03 for prefix pool of PD	11
5.4. draft-joshi-dhc-dhcpv6-aggr-route-opt for aggregation route of PD	11
6. Security Considerations	11
7. IANA Considerations	12
8. Acknowledgements	12
9. References	12
9.1. Normative References	12
9.2. Informative References	12
Appendix A. ChangeLog	13
Appendix B. Software	13
Appendix C. draft-stenberg-pd-route-maintenance-00	13
Appendix D. Snooping only	13
Appendix E. ICMP Redirect	14
Authors' Addresses	14

1. Introduction

The operation of a prefix delegation procedure with the DHCPv6 protocol may need route setup and maintenance at the Delegating Router, Requesting Router and on the entity on which the Relay agent is implemented. This document describes the problem of routing during DHCPv6 prefix delegation, and is illustrated by ADSL-type and cellular-type of topologies which may use Relays; we refer to section 14 of RFC 3633 which mentions the need of 'a protocol or other out of band communication to add routing information for delegated prefixes'. Based on this problem, a number of requirements from the service providers are described.

A small set of documented solutions are separately mentioned (snooping, route injection, etc.), together with their pros and cons according to a particular judgment.

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].

PE router stands for 'Provider Edge' router. It is a router in the provider's network, situated at its edge. It is connected directly to the CE router ('Customer's Edge') which is situated within the customer's network, at its edge.

3. Problem and the Related Topologies

3.1. Problem

This is a problem mentioned in the context of the specification of the Relay Agent behaviour, in DHCPv6 Prefix Delegation [RFC3633]. If the network topology in which running Prefix Delegation is run involves a Relay Agent, then the delegating router may need a protocol or other out-of-band communication to add routing information for delegated prefixes into any router through which the requesting router may forward traffic. That protocol or out-of-band communication are left unspecified.

A more detailed interpretation of that problem is described next.

Intuitively, the Prefix Delegation operation consists in a request and a delegation phase (more precisely, a prefix allocation is performed by the software in the Server, and messages such as

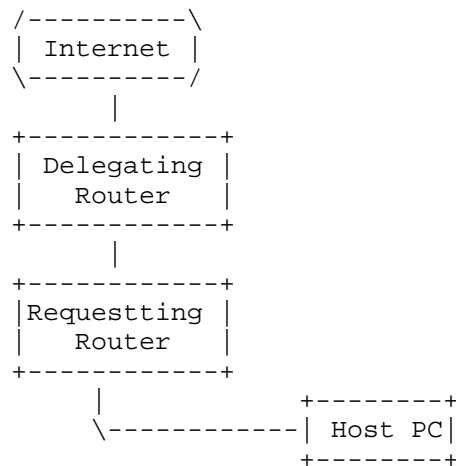
Solicit/Advertise/Request/Reply are used, see [RFC3315]). In the request phase, the Requesting Router requests an IPv6 prefix (not just an IPv6 address). In the delegation phase, the Delegating Router allocates (reserves) a prefix for use by the Requesting Router; this prefix is then sent by the Delegating Router to the Requesting Router.

Allocating a prefix is an operation different than allocating simply an address. In IPv6, one of the differences relates to the way in which the routing is set up with respect to the allocated parameter. The routing for the allocated address is pre-set ((1) the address is part of a prefix, and the routing is pre-set for that prefix and (2) the address is allocated by the Server and may be resolved on-link); whereas the routing for a prefix can not be pre-set ((1) it is next to impossible to pre-aggregate several /64 allocated prefixes into a single pre-set /64 prefix and (2) the IP address of the next-hop is not known at the Server during the prefix allocation phase, being pre-configured on the Client, and not relayed in the messages sent by the Relay; yet this address is needed for route setup). The concepts of numbered and unnumbered interface may also play a distinctive role.

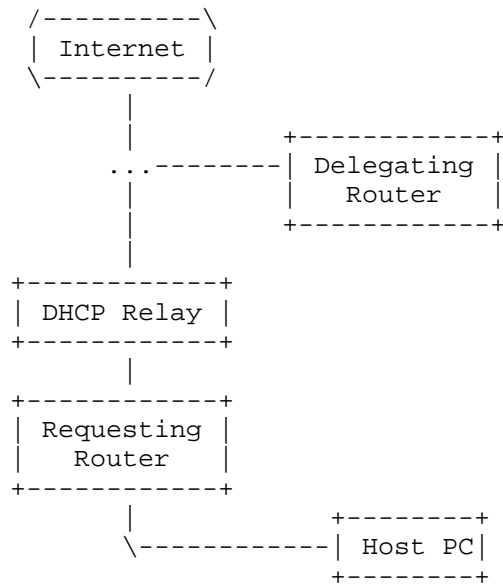
An alternative explanation: in the case of allocating an address, the routing is pre-set for one particular prefix, during network setup operation. Due to the imposed length of the Interface ID on the widely used Ethernet-type of links, that pre-set prefix has length 64. That particular prefix covers the entire set of addresses which may be dynamically allocated by DHCP. On the contrary, dynamically allocating a prefix does not benefit of this pre-setting of routing, because of that 64 limit. One can not pre-set a prefix of length 64 which covers other prefixes of same length /64. There is thus a need to dynamically set a route for the allocated prefix (because it is not possible to pre-set routes for allocated prefixes).

3.2. Relay vs. non-Relay Topologies

In practice, some topologies may accommodate easily the deployment of Prefix Delegation, yet other topologies may pose problems with respect to PD. A topology where the Requesting Router is a neighbor to the Delegating Router, and the Requesting Router's default route is the Delegating Router, may easily accommodate the Prefix Delegation operation (in this case too, the delegating router needs the operation of route set-up for network reachability). This topology is pictured below.



On another hand, a topology where the Requesting Router is not an immediate IP neighbor to the Delegating Router, and/or RR's default route is not the DR, the operation of allocating a prefix must necessarily involve an operation of route set up. This topology is illustrated below.



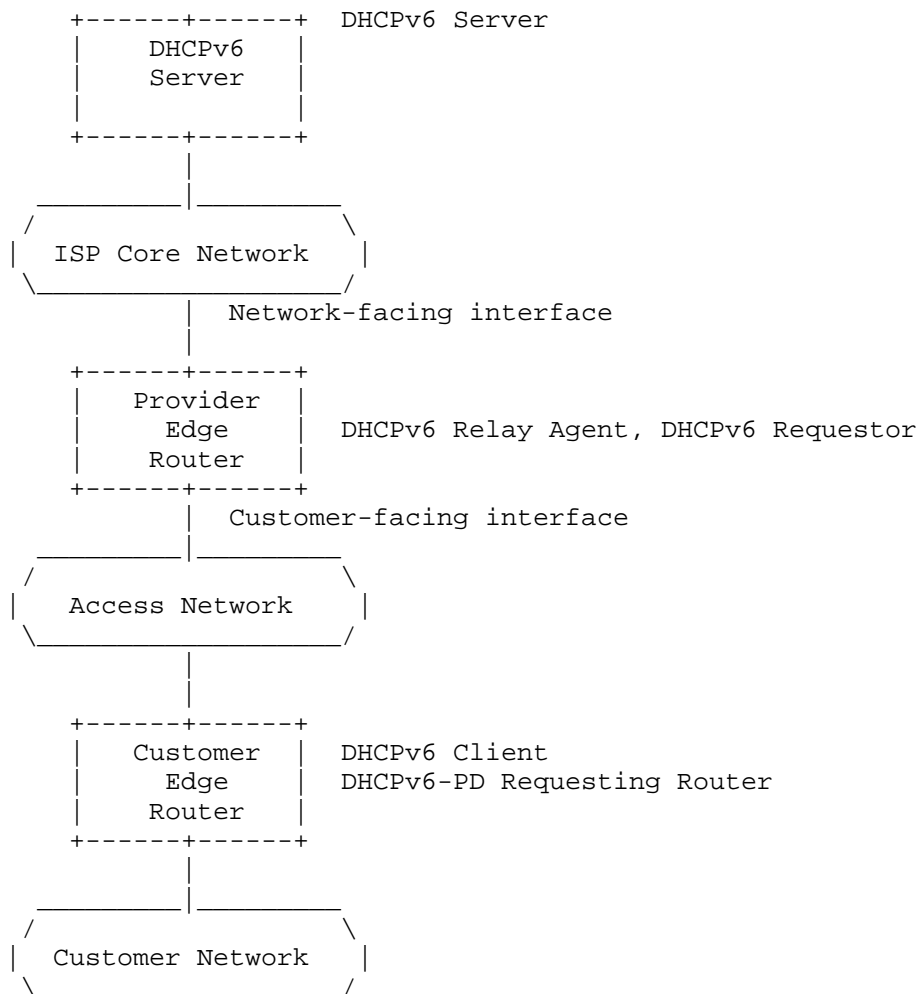
The operation of Prefix Delegation in the topology illustrated above needs to provoke the setting up of a route at the DHCP Relay during the Prefix Delegation operation. Otherwise, the DHCP Relay will not be able to forward packets addressed to Host PC (which is configured with an address in the range of the allocated prefix).

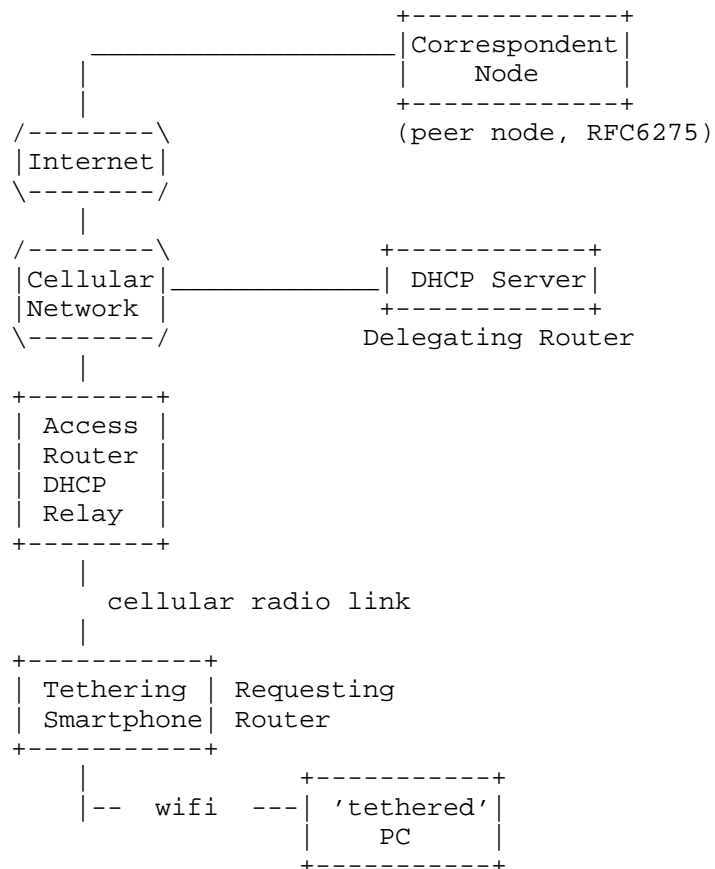
This problem statement is related exclusively to the operation of the Relay Agent and the computer (Router or Host) on which this Agent runs. These entities are direct neighbors (in the sense of the Neighbor Discovery protocol) with the interface on which the Requesting Router emits the DHCPv6 Request message. On another hand, a similar problem, is considered in a more generic manner: upon delegation, use a routing protocol to maintain routing state at Delegating Routers, at other intermediary routers (for routers not necessarily direct neighbors to the RR), and at the Requesting Router; this is described in section 2.3.4 of [I-D.stenberg-pd-route-maintenance].

3.3. Relay Topologies for Large-scale Deployments

The Figure 1 in [RFC3633] illustrates a network architecture in which prefix delegation could be used. That architecture is relating directly to the use of DSL deployments.

The following topologies pertinent for large-scale deployments are considered. A topology for home deployments, and a topology of mobile hotspots (tethering smartphone) are pictured.





In the above figure, the cellular network is considered to be similar to an ISP network.

4. Issues and Requirements

As a reminder, the main requirement from service providers is the following:

- o Need for deterministic and dynamic means to drive route aggregates and associated route announcement actions to be undertaken by PE routers while ensuring a consistency with prefix assignment states. In particular:
 - * Optimizing the size of routing and forwarding tables must be supported. As such, route aggregation must be supported by

these routers.

- * Failures to deliver incoming packets to a customer serviced behind a given PE router must be avoided. Dedicated routing actions must be achieved to ensure incoming traffic will be forwarded to the appropriate customer's device. Nevertheless, triggers to drive the level of route aggregation and required route announcement actions must be supported.
- * Prefix assignments and routing actions must be correlated otherwise delivery of connectivity service will fail.

These requirements can be fulfilled by a variety of solutions that have their limits. For instance:

- o Current practices rely on static configuration. This practice is prone to errors.
- o The level of route aggregation cannot be driven by PE routers without any hint(s) from an entity that has the visibility on aggregation policies and the status of prefixes, etc.
- o Relying on proprietary means to trigger the injection of routing entries may lead to undesired behavior: increase the size of routing table and forwarding table due to injecting very specific routes, etc.

Note:

- o Prefix assignment policies can be configured to DHCP servers including topological aware considerations (e.g., regional-based assignment, per-service assignment, etc.). Refer to Section 4 of [I-D.lemon-dhc-topo-conf].
- o Status of active (prefix) states is maintained by DHCP servers.
- o The use of DHCP for this purpose does not require an additional interface to pass the state maintained by the DHCP server to a routing controller which will then undertake appropriate actions.

Finally, it is worth mentioning that other standards development organizations consider the use of DHCPv6 Prefix Delegation in particular contexts:

- o at the Broadband Forum ('BBF'), a technical report titled "IPv6 in the context of TR-101", dated November 2010, [bf], lists a number of requirements derived from the problem that "hosts receiving IPv6 addresses from the RR are not known to the BNG, i.e. the BNG

is not aware of what addresses/prefixes are assigned to hosts attached to the RG acting as RR."

- o at 3GPP, the specification [3gpp-ref] describes the use of DHCPv6 prefix delegation using S2c. It is for a "User Equipment acting as a Mobile Router".

5. Potential Solutions, Solution Space

5.1. DHCPv6 message snooping for PD-prefix

The Provider Edge (PE) router acting as relay snoops every reply message from the server with valid lease. Per the parameters, such as valid-lifetime and preferred-lifetime, shown the IA_PD option, PE router knows the lease of each delegated prefix. Then the PE can add and withdraw the associated route per the lease of each delegated prefix.

Pros: no new messages and options defined, no additional function on the relay.

Cons: but PE router need to snoop each DHCPv6-PD message, then take action for routing per the lease of each delegated prefix.

In the real deployed network, PE router always need to handle each protocol message in the control plane including DHCPv6 message. That makes snooping sounds not a problem for PE router. Almost every implementation of PE router today in the Telecom's network adopts the method of snooping to get the lease of each delegated prefix.

5.2. ietf-dhc-dhcpv6-agentopt-delegate for PD-prefix

[I-D.ietf-dhc-dhcpv6-agentopt-delegate].

The relay use OPTION_ORO to request the assigned address or the delegated prefixes for the client from the server. But the draft has not mentioned in which DHCPv6 message the OPTION_ORO will always be employed.

Pros: no new messages, just define a new RAAN option (OPTION_AGENT_NOTIFY) to convey OPTION_IAADDR and OPTION_IAPREFIX, sounds it can de-couple with the DHCPv6-PD mechanism.

Cons: sounds only for the route @ relay (or PE router) co-related with the delegated prefix, have no route aggregation.

Due to PE router need to interpret and handle each protocol message

in the control plane, it can get the assigned address or the delegated prefixes directly from the DHCPv6 message. That makes RAAN option unnecessary.

5.3. draft-ietf-dhc-dhcpv6-prefix-pool-opt-03 for prefix pool of PD

[I-D.ietf-dhc-dhcpv6-prefix-pool-opt].

PE router acting as relay use OPTION_ORO in the relay-forward of DHCPv6-PD message to request the information about the prefix pool (and its status). After the PE got the OPTION_PREFIX_POOL in the Relay-reply message, it can add the aggregation route per the status (or lease) of the prefix pool. The aggregation route can dramatically reduce the size of the routing table in the ISP network.

Pros: no new messages, just define a new option (OPTION_PREFIX_POOL) to convey the information about prefix pools.

Cons: lightweight but piggyback on the UDP-based DHCPv6 message.

This draft hasn't achieve Consensus yet, but may need to simplify the mechanism to gain more support.

5.4. draft-joshi-dhc-dhcpv6-aggr-route-opt for aggregation route of PD

[I-D.joshi-dhc-dhcpv6-aggr-route-opt]

PE router acting as relay tries to employ new messages exchange between relay and server, including new function of information-request, renew and reply, reconfigure. That make the communication between the relay and server to be the communication between hosts.

Pros: de-coupled from the DHCPv6-PD mechanism, communication between hosts could employ the reliable TCP.

Cons: introduce new functionality defined for the relay and server, define new messages and option (OPTION_AGGR_ROUTE), have problem of synchronization for the status of aggregation route or for the leases of delegated prefixes.

This draft may be incomplete work, maybe further discussion may be needed.

6. Security Considerations

7. IANA Considerations

8. Acknowledgements

The authors would like to acknowledge Mikael Abrahamsson

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.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

9.2. Informative References

- [3gpp-ref] "3GPP TS 23.402 V12.2.0 (2013-09), Technical Specification, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Architecture enhancements for non-3GPP accesses (Release 12), http://www.3gpp.org/ftp/Specs/archive/23_series/23.402/23402-c20.zip accessed on October 7th, 2013".
- [I-D.ietf-dhc-dhcpv6-agentopt-delegate] Droms, R., Volz, B., and O. Troan, "DHCPv6 Relay Agent Assignment Notification (RAAN) Option", draft-ietf-dhc-dhcpv6-agentopt-delegate-04 (work in progress), July 2009.
- [I-D.ietf-dhc-dhcpv6-prefix-pool-opt] leaf.yeh.sdo@gmail.com, l., Lemon, T., and M. Boucadair, "Prefix Pool Option for DHCPv6 Relay Agent on the Provider Edge Routers", draft-ietf-dhc-dhcpv6-prefix-pool-opt-03 (work in progress), April 2013.

- [I-D.joshi-dhc-dhcpv6-aggr-route-opt]
Joshi, S., "Aggregate Route Option for Dynamic Host Control Protocol version 6 (DHCPv6)", draft-joshi-dhc-dhcpv6-aggr-route-opt-01 (work in progress), September 2011.
- [I-D.lemon-dhc-topo-conf]
Lemon, T., "Customizing DHCP Configuration on the Basis of Network Topology", draft-lemon-dhc-topo-conf-01 (work in progress), April 2013.
- [I-D.stenberg-pd-route-maintenance]
Stenberg, M. and O. Troan, "IPv6 Prefix Delegation routing state maintenance approaches", draft-stenberg-pd-route-maintenance-00 (work in progress), October 2006.
- [bf] "Broadband Forum, Technical Report, TR-177, "IPv6 in the Context of TR-101, Issue: 1, Issue date: November 2010", document freely available at the URL <http://www.broadband-forum.org/technical/download/TR-177.pdf> accessed on October 4th, 2013."

Appendix A. ChangeLog

The changes are listed in reverse chronological order, most recent changes appearing at the top of the list.

From draft-relay-route-pd-problem-00.txt to
draft-authors-relay-route-pd-problem-00.txt:

- o first version.

Appendix B. Software

Prototype implementations.

Appendix C. draft-stenberg-pd-route-maintenance-00

Appendix D. Snooping only

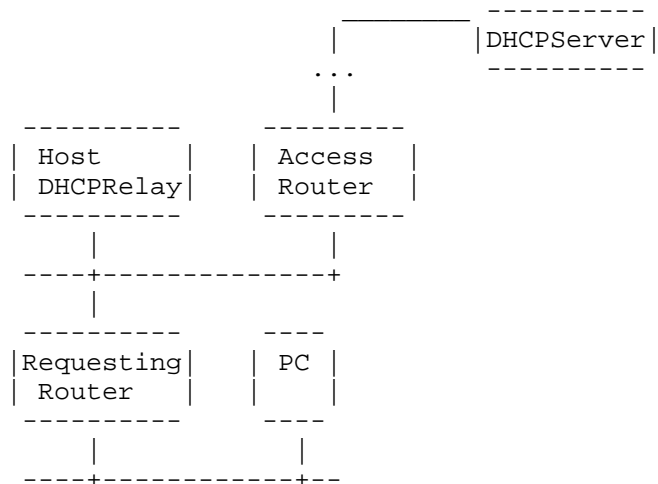
not sure.

Appendix E. ICMP Redirect

One possible solution to inform an entity in the network about a better route to a destination is to use the message ICMPv6 Redirect, as specified in [RFC4861]. This message is used by Routers to inform hosts about better routes.

Some DHCPv6 Prefix Delegation deployments consider that the DHCPv6 Relay functionality is co-located within a Router. In this case, it is not possible to use the ICMP Redirect message.

However, in other possible deployments, the DHCPv6 Relay functionality is co-located in a Host; in this case the use of ICMPv6 Redirect may be possible. An example topology of this use of DHCP Relay on a Host is depicted in the following figure:



Authors' Addresses

Mohamed Boucadair
France Telecom
Rennes, 35000
France

Email: mohamed.boucadair@orange.com

Leaf Y. Yeh
Freelancer Technologies
P. R. China

Email: leaf.yeh.sdo@gmail.com

Alexandru Petrescu
CEA
France

Phone:
Email: Alexandru.Petrescu@cea.fr

DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 7, 2014

D. Raghuvanshi
D. Kukrety
Cisco Systems, Inc.
October 4, 2013

DHCPv6 Active Leasequery
draft-raghuvanshi-dhc-dhcpv6-active-leasequery-00.txt

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) has been extended with a Leasequery capability that allows a client to request information about DHCPv6 bindings. That mechanism is limited to queries for DHCPv6 binding data updates until the time DHCPv6 server receive the Leasequery request. Continuous update of an external client with Leasequery data is sometimes desired. This document expands on the DHCPv6 Leasequery protocol, and allows for active transfer of real-time DHCPv6 binding information data via TCP. This document also extends DHCPv6 Bulk Leasequery by adding new options.

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 7, 2014.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	3
2. Terminology	3
3. Protocol Overview	4
4. Interaction Between Active Leasequery and Bulk Leasequery . .	6
5. Extension to DHCPv6 Bulk Leasequery	7
6. Message and Option Definitions	7
6.1. Message Framing for TCP	7
6.2. Messages	7
6.2.1. ACTIVELEASEQUERY	7
6.3. Options	8
6.3.1. OPTION_LQ_BASE_TIME	8
6.3.2. OPTION_LQ_START_TIME	9
6.3.3. OPTION_LQ_END_TIME	10
6.4. Connection and Transmission Parameters	10
7. Information Communicated by Active Leasequery	11
8. Requestor Behavior	12
8.1. Connecting and General Processing	12
8.2. Forming an Active Leasequery	12
8.3. Processing Active Replies	13
8.3.1. Processing Replies from a Request Containing a OPTION_LQ_START_TIME	15
8.4. Processing Time Values in Leasequery messages	17
8.5. Examples	18
8.5.1. Query Failure	18
8.5.2. Data Missing on Server	18
8.5.3. Successful Query	19
8.6. Closing Connections	19
9. Server Behavior	19
9.1. Accepting Connections	20
9.2. Replying to an Active Leasequery	20
9.3. Multiple or Parallel Queries	22
9.4. Closing Connections	22
10. Security Considerations	22
11. IANA Considerations	24
12. Acknowledgements	24
13. Modification History	24
14. References	24
14.1. Normative References	24
14.2. Informative References	25
Authors' Addresses	25

1. Introduction

The DHCPv6 [RFC3315] protocol specifies a mechanism for the assignment of IPv6 address and configuration information to IPv6 nodes. IPv6 Prefix Delegation for DHCPv6 (PD) [RFC3633] specifies a mechanism for DHCPv6 delegation of IPv6 prefixes and related data. DHCPv6 servers maintain authoritative information including binding information for delegated IPv6 prefixes.

Requirements exist for external entities to keep up to date on the correspondence between DHCPv6 clients and their bindings. These requirements often stem from regulatory requirements placed on service providers by governmental agencies.

These entities need to keep up with the current binding activity of the DHCPv6 server. Keeping up with these binding activity is termed "active" leasequery.

The DHCPv6 Bulk Leasequery [RFC5460] capability can be used to recover useful information from a DHCPv6 server when some external entity starts up. This entity could be one which is directly involved in the DHCPv6 client - server transactions (e.g., a relay agent), or it could be an external process which needs information present in the DHCPv6 server's lease state database.

The Active Leasequery capability documented here is designed to allow an entity not directly involved in DHCPv6 client - server transactions to nevertheless keep current with the state of the DHCPv6 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 [RFC2119].

DHCPv6 terminology is defined in [RFC3315]. Terminology specific to DHCPv6 Active Leasequery can be found below:

- o "Absolute Time"

A 32-bit quantity containing the number of seconds since January 1, 2000.

- o "Active Leasequery"

Keeping up to date in real-time (or near real-time) with DHCPv6

binding activity.

- o "Bulk Leasequery"

Requesting and receiving the existing DHCPv6 binding information in an efficient manner.

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

If a DHCPv6 Active Leasequery requestor sends `OPTION_LQ_START_TIME` option in a `ACTIVELEASEQUERY` message, the DHCPv6 server will attempt to send the requestor the information that changed since the time specified in the `OPTION_LQ_START_TIME` option. The 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.

- o "clock skew"

The difference between the absolute time on a DHCPv6 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 DHCPv6 server, a more accurate value is derived from continuously examining the instantaneous value developed from each packet received from a DHCPv6 server and using it to make small adjustments to the existing value held in the requestor.

- o "Transaction ID"

An opaque value used to match responses with queries initiated by an Active Leasequery client.

3. Protocol Overview

The Active Leasequery mechanism is modeled on the existing DHCPv6 Bulk Leasequery [RFC5460]; most differences arise from the long term nature of the TCP connection required for Active Leasequery. In addition, a DHCPv6 server which supports Active Leasequery MUST support Bulk Leasequery [RFC5460] as well.

An Active Leasequery client opens a TCP connection to a DHCPv6 Server, using the DHCPv6 port 547. 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 DHCPv6 server. No relaying for Active Leasequery is specified.

After establishing a connection, the client sends an ACTIVELEASEQUERY message over the connection. In response, the server sends updates to the requestor using LEASEQUERY-REPLY and LEASEQUERY-DATA messages. This response procedure is identical to [RFC5460], except that in case of Active Leasequery server sends updates whenever some activity occurs to change binding state - thus the need for long lived connection. Active Leasequery is designed to provide continuous updates of DHCPv6 IPv6 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 IPv6 bindings 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 DHCPv6 server. Since that isn't sufficiently robust, much of the mechanism in this document is designed to deal efficiently with situations that occur when the Active Leasequery requestor becomes disconnected from the DHCPv6 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 DHCPv6 server will determine whether or not data was missed while the Active Leasequery requestor was not connected.

The DHCPv6 server processing the Active Leasequery request may limit the amount of data saved, and methods exist for the DHCPv6 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 [RFC5460] to recover information not available through an Active Leasequery.

DHCPv6 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 DHCPv6 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 binding information. In addition, it would use Bulk Leasequery to recover missed information in the event that its connection with the DHCPv6 server was lost for a longer time than the DHCPv6 server would keep track of the specific changes to the IPv6 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 `OPTION_CLIENT_DATA` and additional options discussed in the Bulk Leasequery specification [RFC5460] 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 [RFC5460]. The format of packets returned to an Active Leasequery requestor are identical to that defined for the Bulk Leasequery protocol [RFC5460].

Applications which employ Active Leasequery to keep a database up to date with respect to the DHCPv6 server's lease state database will usually use an initial Bulk Leasequery to bring their database into equivalence with that of the DHCPv6 server, and then use Active Leasequery to keep that database current with respect to the DHCPv6 server's lease state database.

There are several differences between the Active and Bulk Leasequery protocols. Active Leasequery defines a new message (`ACTIVELEASEQUERY`) to send Active Leasequery request to DHCPv6 server. An Active Leasequery connection sends all available updates to the requestor, based on `OPTION_LQ_QUERY` option (see Section 6.2.1).

An Active Leasequery connection does not ever "complete", though the DHCPv6 server may drop the connection for a variety of reasons associated with some sort of exception condition.

5. Extension to DHCPv6 Bulk Leasequery

This document extends to the capabilities of DHCPv6 Bulk Leasequery protocol [RFC5460] by defining new options. More details about these options are specified in Section 6.3.

6. Message and Option Definitions

6.1. Message Framing for TCP

The use of TCP for the Active Leasequery protocol permits one or more DHCPv6 messages to be sent at a time. The receiver needs to be able to determine how large each message is. The same message framing technique used for DHCPv6 Bulk Leasequery [RFC5460] is used for Active Leasequery as well.

The intent in using the same format is that code which currently knows how to deal with a message returned from DHCPv6 Bulk Leasequery [RFC5460] will be able to deal with the message held inside of the TCP framing.

6.2. Messages

The LEASEQUERY-REPLY message is defined in [RFC5007]. The LEASEQUERY-DATA and LEASEQUERY-DONE messages are defined in [RFC5460].

In a Active Leasequery exchange, a single LEASEQUERY-REPLY message is used to indicate the success or failure of a query, and to carry data that do not change in the context of a single query and answer, such as the Server-ID and Client-ID options. If a query is successful, only a single LEASEQUERY-REPLY message MUST appear. If the server is returning binding data, the LEASEQUERY-REPLY also contains the first client's binding data in an OPTION_CLIENT_DATA option. Additional binding data is returned using LEASEQUERY-DATA message as explained in DHCPv6 Bulk Leasequery [RFC5460]. In case of failure query, single LEASEQUERY-REPLY message is returned without any binding data.

6.2.1. ACTIVELEASEQUERY

The new message type (ACTIVELEASEQUERY) is designed to assist requestor keeping up to date in real-time (or near real-time) with DHCPv6 bindings. It asks the server to return DHCPv6 bindings activity that occurs subsequent to the receipt of the Active Leasequery request.

An ACTIVELEASEQUERY request MUST contain a transaction-id, and that

transaction-id MUST BE locally unique to the TCP connection to the DHCPv6 server.

While sending Active Leasequery request, requestor MAY include OPTION_LQ_START_TIME option in ACTIVELEASEQUERY request. In this case, DHCPv6 server returns all the bindings changed on or after the OPTION_LQ_START_TIME.

If the requestor is interested in receiving all binding updates from DHCPv6 server, it MUST NOT include OPTION_LQ_QUERY option in ACTIVELEASEQUERY message. But if the requestor is only interested in specific binding updates, it MAY include OPTION_LQ_QUERY option along with query-types defined in [RFC5007] and [RFC5460].

Other DHCPv6 options used in LEASEQUERY message (as specified in [RFC5460]) can also be used in ACTIVELEASEQUERY request.

6.3. Options

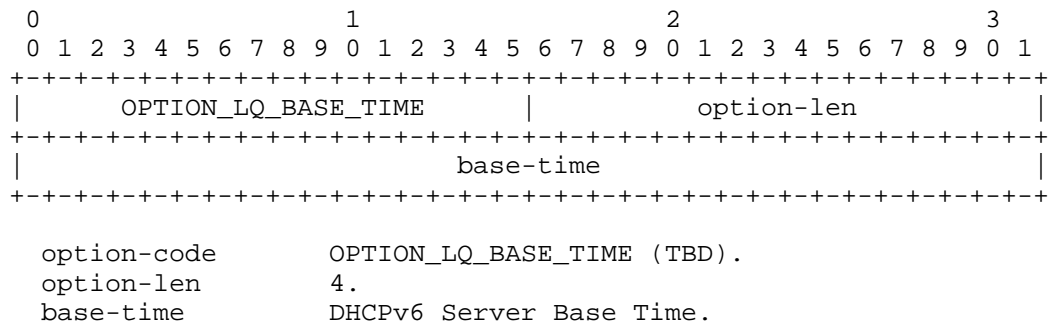
New options (OPTION_LQ_BASE_TIME, OPTION_LQ_START_TIME and OPTION_LQ_END_TIME) are defined as an extension to DHCPv6 Bulk Leasequery [RFC5460]. DHCPv6 server sends OPTION_LQ_BASE_TIME option in Active or Bulk Leasequery response if requestor ask for the same in Active or Bulk Leasequery request. OPTION_LQ_START_TIME can be used in Active or Bulk Leasequery request made to DHCPv6 server. OPTION_LQ_END_TIME can be used in Bulk Leasequery request made to DHCPv6 server. The reply messages for Active Leasequery uses the options defined in [RFC3315], [RFC5007] and [RFC5460].

6.3.1. OPTION_LQ_BASE_TIME

The OPTION_LQ_BASE_TIME option is the current time the message was created to be sent by the DHCPv6 server to the requestor of the Active or 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 OPTION_CLT_TIME [RFC5007]. This time is in the context of the DHCPv6 server who placed this option in a message.

This is an unsigned integer in network byte order.

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



6.3.2. OPTION_LQ_START_TIME

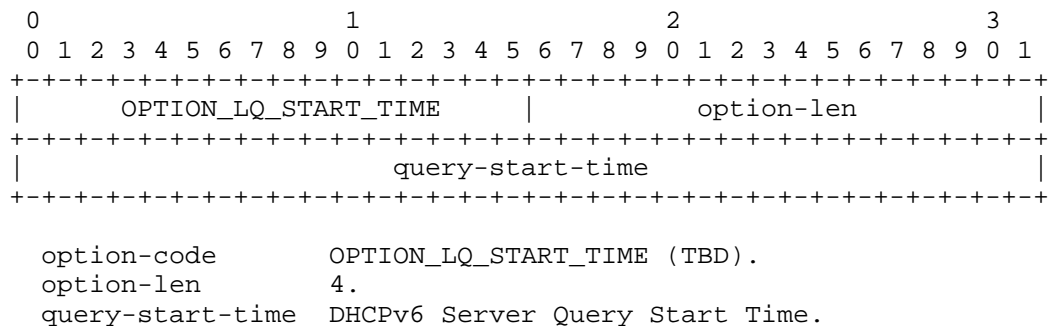
The `OPTION_LQ_START_TIME` option specifies a query start time to the DHCPv6 server. If specified, only bindings that have changed on or after the `OPTION_LQ_START_TIME` should be included in the response to the query.

The requestor **MUST** determine the `OPTION_LQ_START_TIME` using lease information it has received from the DHCPv6 server. This **MUST** be an absolute time in the DHCPv6 server's context (see Section 8.4).

Typically (though this is not a requirement) the `OPTION_LQ_START_TIME` option will contain the value most recently received in a `OPTION_LQ_BASE_TIME` option by the requestor, as this will indicate the last successful communication with the DHCPv6 server.

This is an unsigned integer in network byte order.

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



6.3.3. OPTION LO END TIME

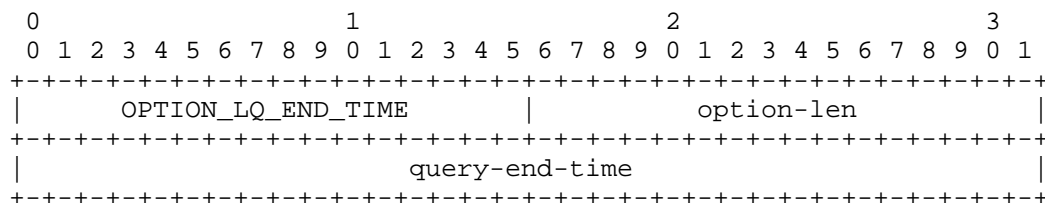
The `OPTION_LQ_END_TIME` option specifies a query end time to the DHCPv6 server. If specified, only bindings that have changed on or before the `OPTION_LQ_END_TIME` should be included in the response to the query. This option MAY be used in Bulk Leasequery request. But it MUST NOT be used in Active Leasequery request.

The requestor **MUST** determine the `OPTION_LQ_END_TIME` based on lease information it has received from the DHCPv6 server. This **MUST** be an absolute time in the context of the DHCPv6 server.

In the absence of information to the contrary, the requestor SHOULD assume that the time context of the DHCPv6 server is identical to the time context of the requestor (see Section 8.4).

This is an unsigned integer in network byte order.

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



```
option-code      OPTION_LQ_END_TIME (TBD).
option-len       4.
query-end-time   DHCPv6 Server Query End Time.
```

6.4. Connection and Transmission Parameters

Active Leasequery uses the same port configuration as DHCPv6 Bulk Leasequery [RFC5460]. It also uses the other transmission parameters (BULK LQ DATA TIMEOUT and BULK LQ MAX CONNS) as defined in [RFC5460].

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
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

7. Information Communicated by Active Leasequery

While the information communicated by a DHCPv6 Bulk Leasequery [RFC5460] is taken directly from the DHCPv6 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 binding in the DHCPv6 server's lease state database.

This is of significance, because if the Active Leasequery requestor runs slowly or the requestor disconnects from the DHCPv6 server and then reconnects with a `OPTION_LQ_START_TIME` (signalling a catch-up operation), the information communicated to the Active Leasequery requestor is only the most current information from the DHCPv6 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 DHCPv6 server is only required to transmit information about a binding 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 DHCPv6 server is waiting to send information down the connection, when the connection becomes available to be written the DHCPv6 server **MAY** create the packet to send at this time. The current state of the binding 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 DHCPv6 server are tuned to take into account the requirements of an Active Leasequery requestor. For instance, the period after the expiration or release of a binding 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 binding.

8. Requestor Behavior

8.1. Connecting and General Processing

A Requestor attempts to establish a TCP connection to a DHCPv6 Server in order to initiate a Active Leasequery exchange. If the attempt fails, the Requestor MAY retry.

If an Active Leasequery is terminated prematurely by a LEASEQUERY-DONE with a DHCPv6 status code (carried in an OPTION_STATUS_CODE option) 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 DHCPv6 server come as multiple responses to a single ACTIVELEASEQUERY message. Thus, each ACTIVELEASEQUERY 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 contain the same xid as the request. It is the xid which allows the data-streams of two or more different ACTIVELEASEQUERY requests to be de-multiplexed by the requestor.

A requestor MAY send a ACTIVELEASEQUERY request to a DHCPv6 server and immediately close the transmission side of its TCP connection, and then read the resulting response messages from the DHCPv6 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.

8.2. Forming an Active Leasequery

The Active Leasequery is designed to create a long lived connection between the requestor and the DHCPv6 server processing the active query. The DHCPv6 server will send binding information back across this connection with minimal delay after it learns of the binding information. It will learn about bindings either because it makes the bindings itself or because it has received information about a binding from another server.

To form the Active Leasequery, a DHCPv6 request is constructed with a message type of ACTIVELEASEQUERY. The DHCPv6 request MUST contain a transaction-id, and that transaction-id MUST BE locally unique to the TCP connection to the DHCPv6 server.

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 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 DHCPv6 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 (OPTION_LQ_BASE_TIME) received in Leasequery reply messages to use later in a subsequent Active Leasequery request.

This capability is enabled by the transmission of a 4 octet OPTION_LQ_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 DHCPv6 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 DHCPv6 server, the requestor will place this highest base-time value into a OPTION_LQ_START_TIME option in the new Active Leasequery request.

If the requestor doesn't wish to request an update of information missed when it was not connected to the DHCPv6 server, then it does not include the OPTION_LQ_START_TIME option in the Active Leasequery 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.

8.3. Processing Active Replies

The Requestor attempts to read a DHCPv6 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.

The requestor examines the LEASEQUERY-REPLY message, and determines how to proceed. Message validation rules are specified in DHCPv6 Leasequery [RFC5007] and DHCPv6 Bulk Leasequery [RFC5460]. If the reply contains an DHCPv6 status code (carried in an OPTION_STATUS_CODE option), the requestor follows the recommendations in [RFC5007].

Note that an Active Leasequery request specifically requests the

DHCPv6 server to create a long-lived connection which may not have data transferring continuously during its lifetime. Therefore the DHCPv6 server will send a LEASEQUERY-DATA message without binding data (OPTION_CLIENT_DATA) every ACTIVE_LQ_IDLE_TIMEOUT seconds (default 60) in order for the requestor to know that the connection remains alive. This approach is followed only when connection is idle (i.e. server has no binding data to send). During normal binding data exchange, receiving of LEASEQUERY-DATA message by requestor itself signifies that connection is active. 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 DHCPv6 server to send messages. Thus ACTIVE_LQ_RCV_TIMEOUT controls how sensitive the requestor is to be to delays by the DHCPv6 server in sending updates or LEASEQUERY-DATA messages.

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 ACTIVELEASEQUERY message from a single DHCPv6 server.

An Active Leasequery 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 OPTION_LQ_BASE_TIME option in a LEASEQUERY-REPLY or LEASEQUERY-DATA message may appear to be ordered, but the most recent change in the lease state data being returned is not related to the OPTION_LQ_BASE_TIME option value in the messages. Another way to say this is that the ordering of the updates sent by the DHCPv6 server during the catch-up phase is independent of the ordering in the changes in the lease state data. The OPTION_LQ_BASE_TIME option from messages during this phase MUST NOT be saved and used in a subsequent ACTIVELEASEQUERY message's OPTION_LQ_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 Active Leasequery request with no OPTION_LQ_START_TIME (and therefore no catch-up phase), the OPTION_LQ_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 DHCPv6 server) can be used in a subsequent Active Leasequery message's OPTION_LQ_START_TIME after a loss of the Active Leasequery connection.

The LEASEQUERY-DONE message MAY unilaterally terminate a successful Active Leasequery request which is currently in progress in the event that the DHCPv6 server determines that it cannot continue processing a ACTIVELEASEQUERY request. For example, when a server is requested to shut down it SHOULD send a LEASEQUERY-DONE message with a DHCPv6

status code of QueryTerminated and include OPTION_LQ_BASE_TIME option in the message. This SHOULD be the last message on that connection, and once the message has been transmitted, the server should close the connection.

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

8.3.1. Processing Replies from a Request Containing a OPTION_LQ_START_TIME

If the Active Leasequery was requested with a OPTION_LQ_START_TIME, the DHCPv6 server will attempt to send information about all bindings that changed since the time specified in the OPTION_LQ_START_TIME. This is the catch-up phase of the Active Leasequery processing. The DHCPv6 server MAY also begin immediate updates over the same connection of real-time binding information changes. Thus, the catch-up phase may run in parallel with the normal updates generated by the Active Leasequery request.

A DHCPv6 server MAY keep only a limited amount of time ordered information available to respond to an Active Leasequery request containing a OPTION_LQ_START_TIME. Thus, it is possible that the time specified in the OPTION_LQ_START_TIME represents a time not covered by the time ordered information kept by the DHCPv6 server. If this should occur, and there is not enough data saved in the DHCPv6 server to satisfy the request specified by the OPTION_LQ_START_TIME option, the DHCPv6 server will reply immediately with a LEASEQUERY-REPLY message with a DHCPv6 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 Active Leasequery request.

If there is enough data saved to satisfy the request, then LEASEQUERY-REPLY (with OPTION_STATUS_CODE of Success or reply without OPTION_STATUS_CODE option) and LEASEQUERY-DATA messages will begin arrive from the DHCPv6 server. Some of these messages will be related to the OPTION_LQ_START_TIME request and be part of the catch-up phase. Some of these messages will be real-time updates of binding changes taking place in the DHCPv6 server. In general, there is no way to determine the source of each message.

Until the catch-up phase is complete, the latest base-time value received from a DHCPv6 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 Active Leasequery were to terminate prior to the completion of the catch-up

phase.

The requestor will know that the catch-up phase is complete when the DHCPv6 server will transmit a LEASEQUERY-DATA message with the DHCPv6 status code of CatchUpComplete. Once this message is transmitted, all additional LEASEQUERY-DATA messages will relate to real-time ("new") binding changes in the DHCPv6 server.

As discussed in Section 8.3, the requestor SHOULD keep track of the latest base-time option value received over a particular connection, to be used in a subsequent Active Leasequery 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 LEASEQUERY-DONE 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 DHCPv6 server to begin to satisfy the request implied by the OPTION_LQ_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 DHCPv6 server will terminate the catch-up phase of processing immediately by sending a LEASEQUERY-DATA message with a DHCPv6 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 binding during the period from the time specified in the OPTION_LQ_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 binding which had one or more changes of state during the period specified by the OPTION_LQ_START_TIME and the current time. The last message for each binding will contain the state at the current time, and there may be one or more messages concerning a single binding during the catch-up phase of processing.

If a binding changed state multiple times during the time that the requestor was not connected (that is, during the time from the OPTION_LQ_START_TIME and the present), then only the current 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 OPTION_LQ_START_TIME to the present will be represented by an individual Leasequery message.

If the LEASEQUERY-REPLY or LEASEQUERY-DATA message containing a DHCPv6 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 bindings in the DHCPv6 server, then the requestor SHOULD issue a Bulk Leasequery request to recover the information missing from its database. This Bulk Leasequery request should include a `OPTION_LQ_START_TIME`, set to be the same as its `OPTION_LQ_START_TIME` previously included in the `ACTIVELEASEQUERY` responses from the DHCPv6 server, and a `OPTION_LQ_END_TIME` equal to the `OPTION_LQ_BASE_TIME` returned by the DHCPv6 server in the `LEASEQUERY-REPLY` or `LEASEQUERY-DATA` message with the DHCPv6 status code of `DataMissing`.

In the event that the requestor receives a `LEASEQUERY-REPLY` or `LEASEQUERY-DATA` message with a DHCPv6 status code of `DataMissing`, it is a reasonable assumption that it is interested in keeping its database up to date with respect to the DHCPv6 server's internal binding database or it would not have included the `OPTION_LQ_START_TIME` in the `ACTIVELEASEQUERY` message.

Typically, the requestor would have one connection open to a DHCPv6 server for a Active Leasequery request and possibly one additional connection open for a Bulk Leasequery request to the same DHCPv6 server to fill in the data that might have been missed prior to the initiation of the Active Leasequery. The Bulk Leasequery connection would typically run to completion and be closed, leaving one Active Leasequery connection open to a single DHCPv6 server. Alternatively, both requests could be issued over a single connection.

8.4. Processing Time Values in Leasequery messages

Active or Bulk Leasequery requests may be made to a DHCPv6 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 Active or Bulk Leasequery response.

If the requestor of a Active or 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 DHCPv6 server.

When receiving a Active or Bulk Leasequery reply message from the DHCPv6 server, the message will contain a `OPTION_LQ_BASE_TIME` option. The time contained in this `OPTION_LQ_BASE_TIME` option is in the context of the DHCPv6 server. As such, it is an ideal time to save and use as input to an Active or Bulk Leasequery message in the `OPTION_LQ_START_TIME` or `OPTION_LQ_END_TIME` option, should the requestor need to ever issue an Active or Bulk Leasequery message using these option as part of a later query, since these option requires a time in the context of the DHCPv6 server.

In addition to saving the `OPTION_LQ_BASE_TIME` for possible future use in `OPTION_LQ_START_TIME` or `OPTION_LQ_END_TIME` option, the `OPTION_LQ_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.

8.5. Examples

These examples illustrate what a series of queries and responses might look like. These are only examples -- there are no requirement that these sequence must be followed.

8.5.1. Query Failure

This example illustrates the message flows in case DHCPv6 server identifies that it can not accept and/or process Active Leasequery request from the client. This could be because of various reasons (i.e. `UnknownQueryType`, `MalformedQuery`, `NotConfigured`, `NotAllowed`).

Client	Server
-----	-----
ACTIVELEASEQUERY xid 1	----->
	<----- LEASEQUERY-REPLY xid 1 (w/error)

8.5.2. Data Missing on Server

This example illustrates the message flows in case DHCPv6 server identifies that it does not have enough data saved to satisfy the request specified by the `OPTION_LQ_START_TIME` option.

In this case the DHCPv6 server will reply immediately with a `LEASEQUERY-REPLY` message with a DHCPv6 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 Active Leasequery request.

```

Client
-----
ACTIVELEASEQUERY xid 2  ----->
                           <----- LEASEQUERY-REPLY xid 2 (w/error)
                           <----- LEASEQUERY-DATA xid 2
                           <----- LEASEQUERY-DATA xid 2
                           <----- LEASEQUERY-DATA xid 2

```

8.5.3. Successful Query

This example illustrates the message flows in case of successful query processing by DHCPv6 server.

In this case the DHCPv6 server will reply immediately with a LEASEQUERY-REPLY message (with OPTION_STATUS_CODE of Success or reply without OPTION_STATUS_CODE option), followed by binding data in LEASEQUERY-DATA messages. In case, DHCPv6 server wants to abort in-process request and terminate the connection due to some reason, it sends LEASEQUERY-DONE with error code present in OPTION_STATUS_CODE option.

```

Client
-----
ACTIVELEASEQUERY xid 3  ----->
                           <----- LEASEQUERY-REPLY xid 3
                           <----- LEASEQUERY-DATA xid 3
                           <----- LEASEQUERY-DATA xid 3
                           <----- LEASEQUERY-DATA xid 3
                           <----- LEASEQUERY-DATA xid 3
                           <----- LEASEQUERY-DATA xid 3
                           <----- LEASEQUERY-DONE xid 3 (w/error)

```

8.6. Closing Connections

The Requestor or DHCPv6 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.

9. Server Behavior

A DHCPv6 server which supports Active Leasequery MUST support DHCPv6 Bulk Leasequery [RFC5460] as well.

9.1. Accepting Connections

Servers that implement DHCPv6 Active Leasequery listen for incoming TCP connections. Approach used in accepting (or rejecting) the client connection is same as specified in DHCPv6 Bulk Leasequery [RFC5460].

9.2. Replying to an Active Leasequery

The DHCPv6 Leasequery [RFC5007] specification describes the initial construction of LEASEQUERY-REPLY messages. Use of the LEASEQUERY-REPLY and LEASEQUERY-DATA messages to carry multiple bindings is described in DHCPv6 Bulk Leasequery [RFC5460]. Message transmission and framing for TCP is described in Section 6.1.

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 DHCPv6 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 DHCPv6 server will drop the TCP connection.

If the DHCPv6 server encounters an error during initial processing of the ACTIVELEASEQUERY message, it SHOULD send a LEASEQUERY-REPLY message containing an error code of some kind in a DHCPv6 status code option. It SHOULD close the connection after this error is signalled.

If the DHCPv6 server encounters an error during later processing of the ACTIVELEASEQUERY message, it SHOULD send a LEASEQUERY-DONE containing an error code of some kind in a DHCPv6 status code option. It SHOULD close the connection after this error is signalled.

If the server finds any bindings satisfying a query, it sends each binding's data in a reply message. The first reply message is a LEASEQUERY-REPLY. The binding data is carried in an OPTION_CLIENT_DATA option, as specified in [RFC5007]. The server returns subsequent bindings in LEASEQUERY-DATA messages, which can avoid redundant data (such as the requestor's Client-ID).

Every reply to a Active Leasequery request MUST contain the information specified in replies to a DHCPv6 Bulk Leasequery request [RFC5460].

If an Active Leasequery or Bulk Leasequery request contains OPTION_LQ_BASE_TIME option code present in OPTION_ORO, the DHCPv6

server MUST include `OPTION_LQ_BASE_TIME` option in every reply for this request. The value for base-time option is current absolute time in the DHCPv6 server's context.

If an Active Leasequery request contains a `OPTION_LQ_START_TIME` option, it indicates that the requestor would like the DHCPv6 server to send it not only messages that correspond to DHCPv6 binding activity that occurs subsequent to the receipt of the Active Leasequery request, but also messages that correspond to DHCPv6 binding activity that occurred prior to the Active Leasequery request.

If `OPTION_LQ_END_TIME` option appears in a Active Leasequery request, the DHCPv6 server should send a `LEASEQUERY-REPLY` message with a DHCPv6 status code of `MalformedQuery` and terminate the connection.

In order to implement a meaningful response to this query, the DHCPv6 server MAY keep track of the binding activity and associate changes with particular base-time values from the messages. Then, when requested to do so by a Active Leasequery request containing a `OPTION_LQ_START_TIME` option, the DHCPv6 server can respond with replies for all binding activity occurring on that `OPTION_LQ_START_TIME` or later times.

These replies based on the `OPTION_LQ_START_TIME` MAY be interleaved with the messages generated due to current binding activity.

Once the transmission of the DHCPv6 Leasequery messages associated with the `OPTION_LQ_START_TIME` option are complete, a `LEASEQUERY-DATA` message MUST be sent with a DHCPv6 status code value of `CatchUpComplete`.

The DHCPv6 server SHOULD, but is not required to, keep track of a limited amount of previous binding activity. The DHCPv6 server MAY choose to only do this in the event that it has received at least one Active Leasequery request in the past, as to do so will almost certainly entail some utilization of resources which would be wasted if there are no Active Leasequery clients for this DHCPv6 server. The DHCPv6 server SHOULD make the amount of previous binding activity it retains configurable. There is no requirement on the DHCPv6 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 Active Leasequery request yielding a `LEASEQUERY-DONE` message, such as a server shutdown, there will be no `LEASEQUERY-DONE` message at the conclusion of the Active Leasequery processing because that processing will not conclude but will continue until either the

client or the server drops the connection.

9.3. Multiple or Parallel Queries

Requesters 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. But it is RECOMMENDED to use different connection in case of parallel Active and Bulk Leasequeries.

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 requesters seeking service.

9.4. Closing Connections

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 DHCPv6 status code option in a LEASEQUERY-DONE 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 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.

10. Security Considerations

The "Security Considerations" section of [RFC3315] details the general threats to DHCPv6. The DHCPv6 Leasequery specification [RFC5007] describes recommendations for the Leasequery protocol, especially with regard to relayed Leasequery messages, mitigation of packet-flooding denial-of-service (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 DHCPv6 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.

There are two specific issues regarding Active Leasequery security that deserve explicit mention. The first is preventing information that Active Leasequery can provide from reaching clients who are not authorized to receive such information. The second is ensuring that authorized clients of the Active Leasequery capability receive accurate information from the Server (and that this information is not disrupted in transit).

To prevent information leakage to unauthorized clients Servers SHOULD restrict Active Leasequery connections and ACTIVELEASEQUERY messages to certain requestors, either through explicit configuration of the Server itself or by employing external network elements to provide such restrictions. In particular, the typical DHCPv6 client SHOULD NOT be allowed to receive a response to an Active Leasequery request, and some technique MUST exist to allow prevention of such access in any environment where Active Leasequery is deployed.

Connections not from permitted requestors SHOULD be closed immediately, to avoid server connection resource exhaustion or alternatively, simply not be allowed to reach the server at all. Servers SHOULD have the capability to restrict certain requestors to certain query types. Servers MAY reply to queries that are not permitted with the LEASEQUERY-DONE message with a status-code option status of NotAllowed, or MAY simply close the connection.

To prevent disruption and malicious corruption of Active Leasequery data flows between the server and authorized clients these data flows SHOULD transit only secured networks. These data flows are typically infrastructure oriented, and there is usually no reason to have them flowing over networks where such attacks are likely. In the rare cases where these data flows might need to be sent through unsecured networks, they MUST sent over connections secured through means external to the DHCPv4/DHCPv6 server and its client(s) (e.g., through VPN's).

Authentication for DHCP Messages [RFC3315] MUST NOT be used to

attempt to secure transmission of the messages described in this document.

11. IANA Considerations

IANA is requested to assign new DHCPv6 Option Codes in the registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>:

```
OPTION_LQ_BASE_TIME
OPTION_LQ_START_TIME
OPTION_LQ_END_TIME
```

IANA is requested to assign new values in the registry of DHCPv6 Status Codes maintained in <http://www.iana.org/assignments/dhcpv6-parameters>:

```
DataMissing
CatchUpComplete
```

IANA is requested to assign value for the following new DHCPv6 Message type in the registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>:

```
ACTIVELEASEQUERY
```

12. Acknowledgements

Many of the ideas in this document were originally proposed by Kim Kinnear, Bernie Volz, Mark Stapp and John Brzozowski. Many of the concept and content, present in this document, are based on DHCPv4 Active Leasequery.

13. Modification History

14. References

14.1. 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.

- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [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.

14.2. Informative References

- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC4614] Duke, M., Braden, R., Eddy, W., and E. Blanton, "A Roadmap for Transmission Control Protocol (TCP) Specification Documents", RFC 4614, September 2006.

Authors' Addresses

Dushyant Raghuvanshi
Cisco Systems, Inc.
Cessna Business Park,
Varthur Hobli, Outer Ring Road,
Bangalore, Karnataka 560037
India

Phone: +91 080 4365 7476
Email: draghuva@cisco.com

Deepak Kukrety
Cisco Systems, Inc.
Cessna Business Park,
Varthur Hobli, Outer Ring Road,
Bangalore, Karnataka 560037
India

Phone: +91 080 4365 7474
Email: dkukrety@cisco.com

DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 17, 2014

P. Patil
Cisco
M. Boucadair
France Telecom
D. Wing
T. Reddy
Cisco
September 13, 2013

DHCPv6 Dynamic Reconfiguration
draft-wing-dhc-dns-reconfigure-02

Abstract

This specification extends DHCPv6 so that a DHCPv6 Relay Agent can dynamically indicate end host connectivity to a DHCPv6 Server. This information is also triggered by any change in connectivity type provided to the host. The DHCPv6 server uses this information as an input to its decision-making about configuration parameters to be conveyed to that host.

Status of This Memo

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

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

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

This Internet-Draft will expire on March 17, 2014.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	2
2. Terminology	3
3. Problem Statement: Focus on DNS Reconfiguration	3
4. Host Connectivity Status Option	4
5. DHCPv6 Relay Agent Behavior	5
5.1. Relay Forward	5
5.2. Reconfigure Request	6
6. DHCPv6 Server Behavior	6
6.1. Relay Forward	6
6.2. Reconfigure Request	6
7. Host Tracking	7
8. Security Considerations	7
9. IANA Considerations	7
10. References	7
10.1. Normative References	8
10.2. Informative References	8
Authors' Addresses	9

1. Introduction

Some networks are expected to support IPv4-only, dual-stack, and IPv6-only hosts at the same time. Due to devices capabilities and available connectivity types, providing generic configuration from a DHCP server to connected hosts is sub-optimal in most cases, and may even break functionality in some cases. Network infrastructure is usually well equipped to be aware of single/dual-stack nature of hosts. The network can also track and detect transitions from single to dual-stack or vice-versa.

This specification describes a DHCPv6 extension for relay agents to indicate host characteristics pertaining to host connectivity to DHCPv6 servers. The information passed by a relay is generic and a DHCPv6 server can interpret and process this information to make a more informed decision on the configuration parameters that a client is to receive.

The DHCPv6 server can either be configured or have built-in logic to use this information as desired, which is outside the scope of this document.

Section 3 describes a typical problem that can be addressed using the mechanism described in this specification. A DHCPv6 server makes a decision on priority of DNS servers to be sent back to the client based on host connectivity characteristics provided by the relay agent.

While the host stack can be upgraded to send this information to the DHCPv6 server on its own, a generalized upgrade of all DHCPv6 client implementations on all operating systems is extremely difficult.

[DISCUSSION NOTE: A companion solution could be to define a container that can be used to return per-AF specific configuration parameters to the client. In such a scheme, the server blindly returns all pieces of configuration and it is up to the client to make use of appropriate set of parameters according to its available connectivity. This alternative assumes an update of dhcp client. This approach can be seen as complimentary to the one defined in this specification. The document will be updated to reflect consensus of the WG on whether the additional option is to be specified.]

2. Terminology

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

Dual-Stack host: Denotes a host that is configured with both an IPv4 address and IPv6 prefix and is reachable using both IPv4 and IPv6 connectivity.

3. Problem Statement: Focus on DNS Reconfiguration

Default address selection rules specified in [RFC6724] prefers IPv6 over IPv4. If a dual-stack host is configured to use a DNS64 server [RFC6147], it will send its DNS queries to that DNS64 server which will synthesize a AAAA response if no A record is found. Thus, the dual-stack host will always use IPv6 if a DNS lookup was involved, even if IPv4 could have been used more optimally.

In some deployments, if NAT44 [RFC3022] and NAT64 [RFC6146] are deployed on the same network, it is preferable to use NAT44 over NAT64 because of scale, performance and application incompatibility issues (e.g., FTP) [RFC6384]. At the same time, native IPv6 can still be preferred over IPv4.

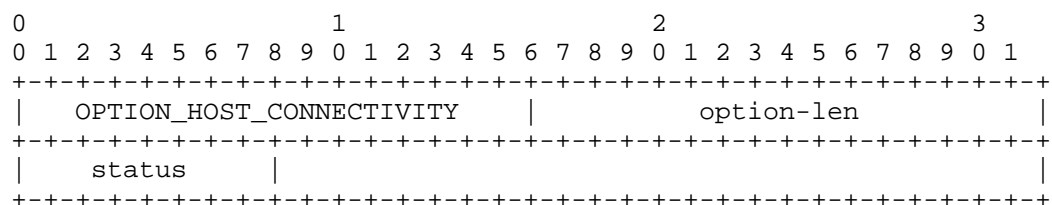
A DHCPv6 Relay Agent can observe host characteristics on a network to determine if a host is IPv4-only, dual-stack or IPv6-only and also

detect transitions from single to dual-stack or vice-versa. This information can be used by the DHCPv6 Relay Agent to influence the DHCPv6 Server to send appropriately prioritized DNS Servers to the client. The DHCPv6 server can implement the following based on connectivity information received from the relay agent.

- o IPv6-only transition to Dual-Stack: In case a host is IPv6-only, it is provided with a DNS64 server. When transitioning to dual-stack, an IPv4 DNS server is assigned as a consequence of obtaining an IPv4 Address. The DHCPv6 Relay Agent can detect this and send a RECONFIGURE_REQUEST message [RFC6977] to the DHCPv6 Server indicating that the host needs to be provided with a regular DNS server. In lieu of this mechanism, the host would continue to use the DNS64 server until the host stack reinitializes.
- o Dual-Stack to IPv6-only: In case a host is dual-stack, it is provided with a regular DNS server followed by DNS64 server. When transitioning to IPv6-only, the DHCPv6 Relay Agent can detect this change and send a RECONFIGURE_REQUEST message to the DHCPv6 server indicating that the host needs to be assigned a DNS64 server only. In lieu of this mechanism, the host would continue to use the regular DNS Server which is inaccessible and eventually time out to fail over to the DNS64 Server. The host will take additional time to fully initialize causing delays in connection.

4. Host Connectivity Status Option

The option (Figure 1) includes an 8-bit status code that indicates specific host connectivity characteristics. The option can be included by a DHCPv6 Relay Agent in RELAY-FORW and RECONFIGURE-REQUEST.



option-code	OPTION_HOST_CONNECTIVITY (TBA).
option-len	1.
status	8-bit integer value carrying the connectivity status of a host. The following codes are defined:

Value	Name
-------	------

	1		IPv4_TO_DUAL_STACK	
	2		IPv6_TO_DUAL_STACK	
	3		DUAL_STACK_TO_IPv4	
	4		DUAL_STACK_TO_IPv6	
+-----+-----+-----+-----+				

Figure 1: Relay Agent Host Connectivity Option message format

- o IPv4_TO_DUAL_STACK: Host is transitioning from IPv4-Only to Dual-Stack mode.
- o IPv6_TO_DUAL_STACK: Host is transitioning from IPv6-Only to Dual-Stack mode.
- o DUAL_STACK_TO_IPv4: Host is transitioning from Dual-Stack to IPv4-Only mode.
- o DUAL_STACK_TO_IPv6: Host is transitioning from Dual-Stack to IPv6-Only mode.

5. DHCPv6 Relay Agent Behavior

DHCPv6 relay agents that implement this specification MUST be configurable for tracking host connectivity and inserting the OPTION_HOST_CONNECTIVITY option in RELAY-FORW and RECONFIGURE-REQUEST messages.

To be able to notify details of hosts' connectivity, a relay agent must be able to track host connectivity. A Relay Agent can detect host connectivity type using mechanisms discussed in Section 7. The Relay Agent then includes this information in the appropriate DHCPv6 message.

Relay agents need to maintain connectivity state of each host it can track. This ensures that notifications to the DHCPv6 server, especially DHCPv6 RECONFIGURE-REQUEST, are accurately sent when there is a change in status. If a relay agent loses state due to some reason (e.g., during restart events), it will build state again using the mechanisms described in Section 7 and then send appropriate notifications to the server. Such notifications are redundant and a DHCPv6 Server can choose to ignore such redundant notifications from the relay agent. Redundant notifications are also possible when relay agents are deployed in fault tolerant mode.

5.1. Relay Forward

DHCPv6 relay agents that implement this specification MAY include the option `OPTION_HOST_CONNECTIVITY` in the `RELAY_FORW` to indicate status of host connectivity.

5.2. Reconfigure Request

DHCPv6 relay agents that implement this specification MUST be configurable for sending the `RECONFIGURE_REQUEST` message. The relay agent generates a Reconfigure-Request [RFC6977] anytime status of host connectivity changes by including `OPTION_HOST_CONNECTIVITY` in the request.

6. DHCPv6 Server Behavior

A DHCPv6 Server that supports `OPTION_HOST_CONNECTIVITY` may either have specific configuration or built-in logic to process information available in the option and send configuration parameters in DHCPv6 responses. How the server consumes and acts on the information obtained in the option are outside the scope of this document.

The DHCPv6 server may use this connectivity information, if available, in addition to other relay agent option data, other options included in the DHCPv6 client messages, server configuration, and physical network topology information in order to assign appropriate configuration to the client.

The server MUST ignore the option if it doesn't recognize the status in the `OPTION_HOST_CONNECTIVITY` option. The server SHOULD maintain the latest status received from the relay agent. The server can use this state to match against subsequent notifications and only further process if there is change in status. A relay agent could, for reasons such as restart, fault-tolerant mode etc, send redundant notifications and matching of status at the server will avoid unnecessary processing and message exchanges.

6.1. Relay Forward

Upon receiving a `RELAY-FORW` message containing `OPTION_HOST_CONNECTIVITY`, the server can send appropriate configuration in the `RELAY-REPLY` response. The server MUST NOT return this option in a `RELAY-REPLY` message.

6.2. Reconfigure Request

Upon receiving a `RECONIFURE-REQUEST` message containing an `OPTION_HOST_CONNECTIVITY` option, the server MUST follow the mechanism described in [RFC6977] to create and send Reconfigure message. The server MUST NOT return this option in a `RECONFIGURE-REPLY` message.

7. Host Tracking

Relay Agents can actively keep track of all IPv4/IPv6 addresses and associated lease times assigned to hosts via the respective DHCP servers. Relay Agents can therefore detect transitions from single to dual-stack and vice-versa efficiently. In addition to this technique, relay agents closest to the client can detect transitions using snooping mechanisms. Network devices today use mechanisms such as ARP and NDP snooping (bindings learnt by snooping all NDP traffic, NS, NA, RS, RA) to determine host characteristics such as IPv4/IPv6 - MAC - DUID bindings. IPv4/IPv6 and MAC counters are also used to determine host liveliness.

First hop devices that implement first hop security also track IP address bindings and determine binding updates such as temporary addresses, deprecated addresses, etc. Existing work such as [I-D.ietf-savi-dhcp] and [I-D.levy-abegnoli-savi-plbt] also aim to active current host bindings, all of which can be leveraged to track host addresses.

These mechanisms help determine if a particular IP address family is inactive, has reverted to using a single stack even though it initially had dual-stack capabilities and detect active dual-stack usage after long periods of single-stack activity.

Other techniques to track host connectivity can be envisaged. It is out of scope of this document to provide an exhaustive list of host tracking techniques.

8. Security Considerations

This document describes an application of the mechanism specified in [RFC6977]. Host tracking mechanisms MUST be reliable. If a relay is compromised, it may be used to force an uncompromised server abuse clients by triggering repetitive reconfigurations. Security considerations described in [RFC6977] are applicable to this mechanism.

9. IANA Considerations

IANA is requested to assign the following new DHCPv6 Option Code in the registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters>:

- o OPTION_HOST_CONNECTIVITY

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6384] van Beijnum, I., "An FTP Application Layer Gateway (ALG) for IPv6-to-IPv4 Translation", RFC 6384, October 2011.
- [RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.
- [RFC6977] Boucadair, M. and X. Pournard, "Triggering DHCPv6 Reconfiguration from Relay Agents", RFC 6977, July 2013.

10.2. Informative References

- [I-D.ietf-savi-dhcp] Bi, J., Wu, J., Yao, G., and F. Baker, "SAVI Solution for DHCP", draft-ietf-savi-dhcp-18 (work in progress), June 2013.
- [I-D.levy-abegnoli-savi-plbt] Levy-Abegnoli, E., "Preference Level based Binding Table", draft-levy-abegnoli-savi-plbt-02 (work in progress), March 2010.
- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.
- [RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [RFC3646] Droms, R., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, December 2003.
- [RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.
- [RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, April 2011.

Authors' Addresses

Prashanth Patil
Cisco Systems, Inc.
Bangalore
India

Email: praspati@cisco.com

Mohamed Boucadair
France Telecom
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: tiredddy@cisco.com