

DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2016

Y. Cui
H. Wang
L. Sun
Tsinghua University
T. Lemon
Nominum
I. Farrer
S. Zoric
Deutsche Telekom AG
March 21, 2016

YANG Data Model for DHCPv6 Configuration
draft-ietf-dhc-dhcpv6-yang-01

Abstract

This document describes a YANG data model [[RFC6020](#)] for the configuration and management of DHCPv6 servers, relays and clients.

Requirements Language

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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

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

This Internet-Draft will expire on September 22, 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	2
1.1.	Terminology	3
2.	DHCPv6 Tree Diagrams	3
2.1.	DHCPv6 Server Tree Diagrams	3
2.2.	DHCPv6 Relay Tree Diagrams	12
2.3.	DHCPv6 Client Tree Diagrams	15
2.4.	Notifications Mechanism for DHCPv6	20
3.	DHCPv6 YANG Model	22
4.	Security Considerations (TBD)	86
5.	IANA Considerations (TBD)	87
6.	Acknowledgements	87
7.	References	87
7.1.	Normative References	87
7.2.	Informative References	87
	Authors' Addresses	89

[1.](#) Introduction

DHCPv6 [[RFC3315](#)] is widely used for supplying configuration and other relevant parameters to clients in IPv6 networks. This document defines a DHCPv6 YANG data model, containing sub-modules for the configuration and management of DHCPv6 servers, relays and clients. A single YANG model covering all of these elements provides an operator with a common interface for the management of the entire DHCPv6 deployment in their network.

Since the publication of the original DHCPv6 specification, there have been a large number of additional documents that update the protocol's operation, add new functions and define new options. The YANG model described in this document incorporates all relevant changes. A full list of the documents which have been considered in the development of this model is included in [Appendix A](#).

IF - Comment - Does anyone have this list?

It is worth noting that as DHCPv6 is itself a device configuration protocol, it is not the intention of this document to replace the configuration of DHCPv6 options and parameters using the DHCPv6 protocol with the configuration of DHCPv6 options using NETCONF/YANG. The DHCPv6 client model is intended for the configuration of the DHCPv6 client function and also for obtaining read-only state data from the client which has been learnt via the normal DHCPv6 message flow. This gives an operator a better method for managing DHCPv6 clients and simplifies troubleshooting.

1.1. Terminology

The reader should be familiar with the terms defined in DHCPv6 [\[RFC3315\]](#) and other relevant documents.

The DHCPv6 tree diagrams provide a concise representation of a YANG module to help the reader understand the module structure. The meaning of the symbols in these diagrams are as follows:

- o Brackets "[" and "]" enclose list keys.
- o Braces "{" and "}" enclose feature content.
- o Parentheses "(" and ")" enclose choice and case nodes, and case nodes are also marked with a colon (":").
- o Symbols after data node names: "?" means an optional node, and "*" denotes a list and leaf-list.
- o Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).

2. DHCPv6 Tree Diagrams

2.1. DHCPv6 Server Tree Diagrams

```
+--rw dhcpv6
  +--rw server
    +--rw serv-attributes
      |  +--rw name                string
      |  +--rw duid              duidtype
      |  +--rw enable            boolean
      |  +--rw interfaces-config* string
      |  +--rw ipv6-address?     inet:ipv6-address
      |  +--rw description?      string
      |  +--rw pd-function        boolean
      |  +--rw stateless-service  boolean
      |  +--rw rapid-commit       boolean
```



```

|   +--rw vendor-info
|   |   +--rw ent-num          uint32
|   |   +--rw data*           string
+--rw option-sets
|   +--rw option-set* [option-set-id]
|   |   +--rw option-set-id    -> /dhcpv6/server/option-sets/
option-set
|   |   +--rw new-or-standard-option* [option-code]
|   |   |   +--rw option-code    uint16
|   |   |   +--rw option-name    string
|   |   |   +--rw option-description    string
|   |   |   +--rw option-reference?    string
|   |   |   +--rw option-value    string
|   |   +--rw user-class-value?    string
|   |   +--rw enterprise-number?    uint32
|   |   +--rw store-client-link-layer?    boolean
|   |   +--rw preference-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw preference-value    uint8
|   |   +--rw sip-server-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw sip-server* [sip-serv-id]
|   |   |   |   +--rw sip-serv-id    uint8
|   |   |   |   +--rw sip-serv-domain-name    string
|   |   |   |   +--rw sip-serv-addr    inet:ipv6-address
|   |   +--rw dns-config-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw dns-server* [dns-serv-id]
|   |   |   |   +--rw dns-serv-id    uint8
|   |   |   |   +--rw dns-serv-addr    inet:ipv6-address
|   |   |   +--rw domain-search-list    string
|   |   +--rw nis-config-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw nis-server* [nis-serv-id]
|   |   |   |   +--rw nis-serv-id    uint8
|   |   |   |   +--rw nis-serv-addr    inet:ipv6-address
|   |   +--rw nis-plus-config-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw nis-plus-server* [nis-plus-serv-id]
|   |   |   |   +--rw nis-plus-serv-id    uint8
|   |   |   |   +--rw nis-plus-serv-addr    inet:ipv6-address
|   |   +--rw info-refresh-time-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw info-refresh-time    yang:timeticks
|   |   +--rw cli-fqdn-option
|   |   |   +--rw enable          boolean
|   |   |   +--rw server-initiate-update    boolean
|   |   |   +--rw client-initiate-update    boolean

```

| | +--rw modify-name-from-cli boolean

```
|  +---rw timezone-option
|  |  +---rw enable                boolean
|  |  +---rw tz-posix              string
|  |  +---rw tz-database           string
|  +---rw ntp-server-option
|  |  +---rw enable                boolean
|  |  +---rw ntp-server* [ntp-serv-id]
|  |  |  +---rw ntp-serv-id        uint8
|  |  |  +---rw ntp-serv-addr      inet:ipv6-address
|  |  |  +---rw ntp-serv-mul-addr  inet:ipv6-address
|  |  |  +---rw ntp-serv-fqdn      string
|  +---rw sntp-server-option
|  |  +---rw enable                boolean
|  |  +---rw sntp-server* [sntp-serv-id]
|  |  |  +---rw sntp-serv-id        uint8
|  |  |  +---rw sntp-serv-addr      inet:ipv6-address
|  +---rw network-boot-option
|  |  +---rw enable                boolean
|  |  +---rw boot-file* [boot-file-id]
|  |  |  +---rw boot-file-id        uint8
|  |  |  +---rw suitable-arch-type* uint16
|  |  |  +---rw suitable-net-if*    uint32
|  |  |  +---rw boot-file-url       string
|  |  |  +---rw boot-file-paras* [para-id]
|  |  |  |  +---rw para-id          uint8
|  |  |  |  +---rw parameter        string
|  +---rw dslite-option
|  |  +---rw enable                boolean
|  |  +---rw dslite-aftr-name      string
|  +---rw kerberos-option
|  |  +---rw enable                boolean
|  |  +---rw default-realm-name     string
|  |  +---rw kdc-info* [kdc-id]
|  |  |  +---rw kdc-id              uint8
|  |  |  +---rw priority            uint16
|  |  |  +---rw weight              uint16
|  |  |  +---rw transport-type      uint8
|  |  |  +---rw port-number         uint16
|  |  |  +---rw kdc-ipv6-addr       inet:ipv6-address
|  |  |  +---rw realm-name          string
|  +---rw addr-selection-option
|  |  +---rw enable                boolean
|  |  +---rw a-bit-set             boolean
|  |  +---rw p-bit-set             boolean
|  |  +---rw policy-table* [policy-id]
|  |  |  +---rw policy-id           uint8
|  |  |  +---rw label               uint8
|  |  |  +---rw precedence          uint8
```



```

|   |   +--rw prefix-len          uint8
|   |   +--rw prefix              inet:ipv6-prefix
| +--rw sol-max-rt-option
|   |   +--rw enable              boolean
|   |   +--rw sol-max-rt-value    yang:timeticks
| +--rw inf-max-rt-option
|   |   +--rw enable              boolean
|   |   +--rw inf-max-rt-value    yang:timeticks
| +--rw pcp-server-option
|   |   +--rw enable              boolean
|   |   +--rw pcp-server* [pcp-serv-id]
|   |       +--rw pcp-serv-id      uint8
|   |       +--rw pcp-serv-addr*   inet:ipv6-address
| +--rw s46-rule-option
|   |   +--rw enable              boolean
|   |   +--rw s46-rule* [rule-id]
|   |       +--rw rule-id          uint8
|   |       +--rw rule-type        enumeration
|   |       +--rw ea-len           uint8
|   |       +--rw prefix4-len      uint8
|   |       +--rw ipv4-prefix      inet:ipv4-prefix
|   |       +--rw prefix6-len      uint8
|   |       +--rw ipv6-prefix      inet:ipv6-prefix
|   |       +--rw port-parameter
|   |           +--rw offset        uint8
|   |           +--rw psid-len      uint8
|   |           +--rw psid          uint16
| +--rw s46-br-option
|   |   +--rw enable              boolean
|   |   +--rw br* [br-id]
|   |       +--rw br-id            uint8
|   |       +--rw br-ipv6-addr     inet:ipv6-address
| +--rw s46-dmr-option
|   |   +--rw enable              boolean
|   |   +--rw dmr* [dmr-id]
|   |       +--rw dmr-id            uint8
|   |       +--rw dmr-prefix6-len   inet:ipv6-address
|   |       +--rw dmr-ipv6-prefix   inet:ipv6-prefix
| +--rw operator-option-ipv6-address
|   |   +--rw enable              boolean
|   |   +--rw operator-ipv6-addr* [operator-ipv6-addr-id]
|   |       +--rw operator-ipv6-addr-id uint8
|   |       +--rw operator-ipv6-addr  inet:ipv6-address
|   |       +--rw operator-option-single-flag
|   |   +--rw enable              boolean
|   |   +--rw flag* [flag-id]
|   |       +--rw flag-id           uint8
|   |       +--rw flag-value        boolean

```



```

|   +---rw operator-option-ipv6-prefix
|   |   +---rw enable                boolean
|   |   +---rw operator-ipv6-prefix* [operator-ipv6-prefix-id]
|   |   |   +---rw operator-ipv6-prefix-id    uint8
|   |   |   +---rw operator-ipv6-prefix6-len  uint8
|   |   |   +---rw operator-ipv6-prefix      inet:ipv6-prefix
|   |   |   +---rw operator-option-int32
|   |   +---rw enable                boolean
|   |   +---rw int32val* [int32val-id]
|   |   |   +---rw int32val-id            uint8
|   |   |   +---rw int32val                uint32
|   +---rw operator-option-int16
|   |   +---rw enable                boolean
|   |   +---rw int16val* [int16val-id]
|   |   |   +---rw int16val-id            uint8
|   |   |   +---rw int16val                uint16
|   +---rw operator-option-int8
|   |   +---rw enable                boolean
|   |   +---rw int8* [int8val-id]
|   |   |   +---rw int8val-id            uint8
|   |   |   +---rw int8val                uint8
|   +---rw operator-option-uri
|   |   +---rw enable                boolean
|   |   +---rw uri* [uri-id]
|   |   |   +---rw uri-id                uint8
|   |   |   +---rw uri                    string
|   +---rw operator-option-textstring
|   |   +---rw enable                boolean
|   |   +---rw textstring* [textstring-id]
|   |   |   +---rw textstring-id          uint8
|   |   |   +---rw textstring              string
|   +---rw operator-option-var-data
|   |   +---rw enable                boolean
|   |   +---rw var-data* [var-data-id]
|   |   |   +---rw var-data-id            uint8
|   |   |   +---rw var-data                binary
|   +---rw operator-option-dns-wire
|   |   +---rw enable                boolean
|   |   +---rw operator-option-dns-wire* [operator-option-dns-wire-id]
|   |   |   +---rw operator-option-dns-wire-id  uint8
|   |   |   +---rw operator-option-dns-wire      string
|   +---rw s46-v4-v6-binding-option
|   |   +---rw enable                boolean
|   |   +---rw ce* [ce-id]
|   |   |   +---rw ce-id                uint8
|   |   |   +---rw ipv4-addr            inet:ipv4-address
|   |   |   +---rw bind-prefix6-len      uint8
|   |   |   +---rw bind-ipv6-prefix      inet:ipv6-prefix

```



```

|         +--rw port-parameter
|         +--rw offset                uint8
|         +--rw psid-len              uint8
|         +--rw psid                  uint16
+--rw network-ranges
|  +--rw option-set [option-set-id]
|  +--rw network-range* [network-range-id]
|    +--rw network-range-id          uint8
|    +--rw network-description        string
|    +--rw network-prefix             inet:ipv6-prefix
|    +--rw inherit-option-set         boolean
|    +--rw option-set [option-set-id]
|    +--rw address-pools
|      | +--rw address-pool* [pool-id]
|      | | +--rw pool-id              uint8
|      | | +--rw pool-prefix          inet:ipv6-prefix
|      | | +--rw start-address        inet:ipv6-address
|      | | +--rw end-address          inet:ipv6-address
|      | | +--rw renew-time           yang:timeticks
|      | | +--rw rebind-time          yang:timeticks
|      | | +--rw preferred-lifetime   yang:timeticks
|      | | +--rw valid-lifetime       yang:timeticks
|      | | +--ro total-ipv6-count      uint64
|      | | +--ro used-ipv6-count       uint64
|      | | +--rw utilization-ratio     threshold
|      | | +--rw inherit-option-set    boolean
|      | | +--rw option-set [option-set-id]
|      | | +--rw reserved-addresses
|      | |   +--rw static-binding* [cli-id]
|      | |     | +--rw cli-id          uint32
|      | |     | +--rw duid            duidtype
|      | |     | +--rw reserv-addr*    inet:ipv6-address
|      | |     +--rw other-reserv-addr* inet:ipv6-address
|      | +--ro binding-info* [cli-id]
|      |   +--ro cli-id                uint32
|      |   +--ro duid                  duidtype
|      |   +--ro cli-ia* [iaid]
|      |     +--ro ia-type              string
|      |     +--ro iaid                 uint32
|      |     +--ro cli-addr*            inet:ipv6-address
|      |     +--ro pool-id?             uint8
+--rw prefix-pools
|  | +--rw prefix-pool* [pool-id]
|  | | +--rw pool-id                  uint8
|  | | +--rw prefix                    inet:ipv6-prefix
|  | | +--rw prefix-length             uint8
|  | | +--rw renew-time               yang:timeticks
|  | | +--rw rebind-time              yang:timeticks

```



```

|   |   | +--rw preferred-lifetime      yang:timeticks
|   |   | +--rw valid-lifetime          yang:timeticks
|   |   | +--rw utilization-ratio       threshold
|   |   | +--rw inherit-option-set      boolean
|   |   | +--rw option-set [option-set-id]
|   |   | +--rw reserved-prefixes
|   |   |   +--rw static-binding* [cli-id]
|   |   |     | +--rw cli-id            uint32
|   |   |     | +--rw duid              duidtype
|   |   |     | +--rw reserv-prefix-len uint8
|   |   |     | +--rw reserv-prefix     inet:ipv6-prefix
|   |   |   +--rw exclude-prefix-len   uint8
|   |   |   +--rw exclude-prefix       inet:ipv6-prefix
|   |   |   +--rw other-reserv-prefix* [reserv-id]
|   |   |     +--rw reserv-id           uint8
|   |   |     +--rw prefix-len          uint8
|   |   |     +--rw prefix              inet:ipv6-prefix
|   |   +--ro binding-info* [cli-id]
|   |     +--ro cli-id                 uint32
|   |     +--ro duid                   duidtype
|   |     +--ro cli-iapd* [iaid]
|   |       +--ro iaid                 uint32
|   |       +--ro cli-prefix*          uint32
|   |       +--ro cli-prefix-len*      uint8
|   |       +--ro pool-id?             uint8
|   +--rw hosts
|     +--rw host* [cli-id]
|       +--rw cli-id                  uint32
|       +--rw duid                    duidtype
|       +--rw inherit-option-set      boolean
|       +--rw option-set [option-set-id]
|       +--rw nis-domain-name?        string
|       +--rw nis-plus-domain-name?   string
+--rw relay-opaque-paras
|   +--rw relays* [relay-name]
|     +--rw relay-name                string
|     +--rw interface-info* [if-name]
|       | +--rw if-name                string
|       | +--rw interface-id           string
|     +--rw subscribers* [subscriber]
|       | +--rw subscriber              uint8
|       | +--rw subscriber-id           string
|     +--rw remote-host* [ent-num]
|       +--rw ent-num                  uint32
|       +--rw remote-id                string
+--rw rsoo-enabled-options
|   +--rw rsoo-enabled-option* [option-code]
|     +--rw option-code                uint16

```



```

|      +--rw description                string
+--ro packet-stats
  +--ro solicit-count                  uint32
  +--ro request-count                  uint32
  +--ro renew-count                    uint32
  +--ro rebind-count                   uint32
  +--ro decline-count                  uint32
  +--ro release-count                  uint32
  +--ro info-req-count                 uint32
  +--ro advertise-count                uint32
  +--ro confirm-count                  uint32
  +--ro reply-count                    uint32
  +--ro reconfigure-count              uint32
  +--ro relay-forward-count            uint32
  +--ro relay-reply-count              uint32

```

Figure 1: DHCPv6 Data Model Structure

Introduction of important nodes:

- o serv-attributes: This container contains basic attributes of a DHCPv6 server such as DUID, server name and so on. Some optional functions that can be provided by the server is also included.
- o duid: Each server and client has only one DUID (DHCP Unique Identifier). The DUID here identifies a unique DHCPv6 server for clients. DUID consists of a two-octet type field and an arbitrary length (no more than 128 bytes) content field.
- o pd-function: Whether the server can act as a delegating router to perform prefix delegation ([[RFC3633](#)]).
- o operator-option-ipv6-address, operator-option-single-flag, operator-option-ipv6-prefix, operator-option-int32, operator-option-int16, operator-option-int8, operator-option-uri, operator-option-textstring, operator-option-var-data, operator-option-dns-wire: are generic option formats described in ([[RFC7227](#)]).
- o stateless-service: A boolean value specifies whether the server support client-server exchanges involving two messages defined in ([[RFC3315](#)]).
- o rapid-commit: Setting the value to '1' represents the server support the Solicit-Reply message exchange. '0' means the server will simply ignore the Rapid Commit option in Solicit message.
- o interfaces-config: A leaf list to denote which one or more interfaces the server should listen on. The default value is to

listen on all the interfaces. This node is also used to set a unicast address for the server to listen with a specific interface. For example, if people want the server to listen on a unicast address with a specific interface, he can use the format like "eth1/2001:db8::1".

- o option-sets: DHCPv6 employs various options to carry additional information and parameters in DHCP messages. This container defines all the possible options that need to be configured at the server side. The relevant RFCs that define those options include: [\[RFC3315\]](#), [\[RFC3319\]](#), [\[RFC3646\]](#), [\[RFC3898\]](#), [\[RFC4242\]](#), [\[RFC4704\]](#), [\[RFC4833\]](#), [\[RFC5908\]](#), [\[RFC5970\]](#), [\[RFC4075\]](#), [\[RFC6334\]](#), [\[RFC6784\]](#), [\[RFC7078\]](#), [\[RFC7083\]](#), [\[RFC7291\]](#), [\[RFC7598\]](#).
- o option-set: A server may allow different option sets to be configured for different conditions (i.e. different networks, clients and etc). This "option-set" list enables various sets of options being defined and configured in a single server. Different sets are distinguished by the key called "option-set-id". All the possible options discussed above are defined in the list and each option is corresponding to a container. Since all the options in the list are optional, each container in this list has a boolean parameter called "enable" to indicate whether this option (container) will be included in the current option set or not. With the "new-or-standard-option" list, it is easy to extend the model when new options are defined. We could also use the "new-or-standard-option" list to define an IETF standard option.
- o network-ranges: This model supports a hierarchy to achieve dynamic configuration. That is to say we could configure the server at different levels through this model. The top level is a global level which is defined as the container "network-ranges". The following levels are defined as sub-containers under it. The "network-ranges" contains the parameters (e.g. option-sets) that would be allocated to all the clients served by this server.
- o network-range: Under the "network-ranges" container, a "network-range" list is defined to configure the server at a network level which is also considered as the second level. Different network are identified by the key "network-range-id". This is because a server may have different configuration parameters (e.g. option sets) for different networks.
- o address-pools: Under the "network-range" list, a container describes the DHCPv6 server's address pools for a specific network is defined. This container supports the server to be configured at a pool level.

- o address-pool: A DHCPv6 server can be configured with several address pools for a specific network. This list defines such address pools which are distinguish by the key called "pool-id".
- o binding-info: A list records a binding information for each DHCPv6 client that has already been allocated IPv6 addresses.
- o prefix-pools: If a server supports prefix delegation function, this container under the "network-range" list will be valid to define the delegating router's prefix pools for a specific network. This container also supports the server to be configured at a pool level.
- o prefix-pool: Similar to server's address pools, a delegating router can also be configured with multiple prefix pools specified by a list called "prefix-pool".
- o binding-info: A list records a binding information for each DHCPv6 requesting router that has already been configured IPv6 prefixes.
- o hosts: A server may also desire to be configured at a host level under some circumstances. This container include a list called "host" to allow the server carrying different parameters (e.g. option sets) for different hosts.
- o relay-opaque-paras: This container contains some opaque values in Relay Agent options that need to be configured on the server side only for value match. Such Relay Agent options include Interface-Id option, Remote-Id option and Subscriber-Id option.
- o rsoo-enabled-options: [[RFC6422](#)] requires that the server SHOULD have an administrator-configurable list of RS00-enabled options. This container include a list called "rsoo-enabled-option" to allow new RS00-enabled options to be defined at the server side.
- o packet-stats: A container presents the packet statistics related to the DHCPv6 server.

2.2. DHCPv6 Relay Tree Diagrams

```
+--rw dhcpv6
  +-- ...
  |
  +--rw relay
    +--rw relay-attributes
      | +--rw name                string
      | +--rw enable              boolean
      | +--rw description?        string
```



```
| +--rw dest-addr*                               inet:ipv6-address
| +--rw subscribers* [subscriber]
| | +--rw subscriber                             uint8
| | +--rw subscriber-id                         string
| +--rw remote-host* [entNum]
| | +--rw ent-num                               uint32
| | +--rw remote-id                             string
| +--rw vendor-info
| | +--rw ent-num                               uint32
| | +--rw data*                                 string
+--rw relay-supplied-options-option
| +--rw rsoo-set* [rsoo-set-id]
| | +--rw rsoo-set-id                           uint8
| | +--rw erp-local-domain-name-option
| | | +--rw enable                             boolean
| | | +--rw erp-for-client* [cli-id]
| | | | +--rw cli-id                           uint32
| | | | +--rw duid                             duidtype
| | | | +--rw erp-name                         string
+--rw relay-if* [if-name]
| +--rw if-name                                 string
| +--rw enable                                 boolean
| +--rw ipv6-address?                          ietf:ipv6-address
| +--rw interface-id?                         string
| +--rw rsoo-set [rsoo-set-id]
| +--rw pd-route* [pd-route-id]
| | +--rw pd-route-id                         uint8
| | +--rw requesting-router-id                uint32
| | +--rw delegating-router-id                uint32
| | +--rw next-router                         inet:ipv6-address
| | +--rw last-router                         inet:ipv6-address
| +--rw next-entity* [dest-addr]
| | +--rw dest-addr                           inet:ipv6-address
| | +--rw available                           boolean
| | +--rw multicast                           boolean
| | +--rw server                              boolean
| +--ro packet-stats
| | +--ro cli-packet-rvd-count                uint32
| | +--ro solicit-rvd-count                   uint32
| | +--ro request-rvd-count                   uint32
| | +--ro renew-rvd-count                     uint32
| | +--ro rebind-rvd-count                     uint32
| | +--ro decline-rvd-count                   uint32
| | +--ro release-rvd-count                   uint32
| | +--ro info-req-rvd-count                   uint32
| | +--ro relay-for-rvd-count                 uint32
| | +--ro relay-rep-rvd-count                 uint32
| | +--ro packet-to-cli-count                 uint32
```



```
|      +--ro adver-sent-count      uint32
|      +--ro confirm-sent-count    uint32
|      +--ro reply-sent-count      uint32
|      +--ro reconfig-sent-count   uint32
|      +--ro relay-for-sent-count   uint32
|      +--ro relay-rep-sent-count   uint32
+--ro relay-stats
    +--ro cli-packet-rvd-count      uint32
    +--ro relay-for-rvd-count       uint32
    +--ro relay-rep-rvd-count       uint32
    +--ro packet-to-cli-count       uint32
    +--ro relay-for-sent-count       uint32
    +--ro relay-rep-sent-count       uint32
    +--ro discarded-packet-count    uint32
```

Introduction of important nodes:

- o relay-attributes: A container describes some basic attributes of the relay agent including some relay agent specific options data that need to be configured previously. Such options include Remote-Id option and Subscriber-Id option.
- o dest-addrs: Each DHCPv6 relay agent may be configured with a list of destination addresses. This node defines such a list of IPv6 addresses that may include unicast addresses, multicast addresses or other addresses.
- o relay-supplied-options-option: DHCPv6 relay agent could provide some information that would be useful to DHCPv6 client. Since relay agent cannot provide options directly to the client, [\[RFC6422\]](#) defines RS00-enabled options to propose options for the server to send to the client. This container models such RS00-enabled options.
- o rs00-set: This list under the "relay-supplied-options-option" container is similar to the "option-set" defined in server feature. It allows the relay to implement several sets of RS00-enabled options for different interfaces. The list only include the EAP Re-authentication Protocol (ERP) Local Domain Name DHCPv6 Option defined in [\[RFC6440\]](#), since it is the only one RS00-enabled options accepted by IANA so far.
- o relay-if: A relay agent may have several interfaces, we should provide a way to configure and manage parameters on the interface-level. A list that describes specific interfaces and their corresponding parameters is employed to fulfil the configuration. Here we use a string called "if-name" as the key of list.

- o pd-route: A sub-container of "relay-if" which describes the route for delegated prefixes into the provider edge router.
- o next-entity: This node defines a list that is used to describe the next hop entity of this relay agent. Different entities are distinguished by their addresses.
- o packet-stats: A container shows packet state information of a specific data communication.
- o relay-stats: The "relay-stats" container records and presents the overall packet statistics of the relay agent.

2.3. DHCPv6 Client Tree Diagrams

```

+--rw dhcpv6
  +-- ...
  |
  +--rw client
    +--rw client-if* [if-name]
      +--rw if-name          string
      +--rw cli-id           uint32
      +--rw duid             duidtype
      +--rw enable           boolean
      +--rw description?     string
      +--rw pd-function      boolean
      +--rw rapid-commit     boolean
      +--rw mo-tab
      | +--rw m-tab          boolean
      | +--rw o-tab          boolean
      +--rw oro-options
      | +--rw oro-option* [option-code]
      |   +--rw option-code  uint16
      |   +--rw description  string
      +--rw client-configured-options
      | +--rw new-or-standard-cli-option* [option-code]
      | | +--rw option-code    uint16
      | | +--rw option-name    string
      | | +--rw option-description string
      | | +--rw option-reference? string
      | | +--rw option-value   string
      | +--rw user-class-option
      | | +--rw enable         boolean
      | | +--rw user-class* [user-class-id]
      | |   +--rw user-class-id  uint8
      | |   +--rw user-class-info string
      | +--rw vendor-class-option
      | | +--rw enable         boolean

```



```

| | +--rw ent-num                uint32
| | +--rw data*                  string
| +--rw client-fqdn-option
| | +--rw enable                  boolean
| | +--rw fqdn                    string
| | +--rw server-initiate-update boolean
| | +--rw client-initiate-update boolean
| +--rw client-architecture-type-option
| | +--rw enable                  boolean
| | +--rw architecture-types* [type-id]
| | | +--rw type-id              uint16
| | | +--rw most-preferred       boolean
| +--rw client-network-interface-option
| | +--rw enable                  boolean
| | +--rw type                    uint8
| | +--rw major                   uint8
| | +--rw minor                   uint8
| +--rw kerberos-principal-name-option
| | +--rw enable                  boolean
| | +--rw principal-name          string
| +--rw client-link-layer-addr-option
| | +--rw enable                  boolean
| | +--rw link-layer-type         uint16
| | +--rw link-layer-addr         string
+--ro identity-associations
| +--ro identity-association* [iaid]
| | +--ro iaid                    uint32
| | +--ro ia-type                  string
| | +--ro ipv6-addr*               inet:ipv6-address
| | +--ro ipv6-prefix*             inet:ipv6-prefix
| | +--ro prefix-length*           uint8
| | +--ro t1-time                  yang:timeticks
| | +--ro t2-time                  yang:timeticks
| | +--ro preferred-lifetime       yang:timeticks
| | +--ro valid-lifetime           yang:timeticks
+--ro if-other-paras
| +--ro uni-dhcpv6-serv-addr       inet:ipv6-address
| +--ro dns-paras
| | +--ro domain-search-list       string
| | +--ro dns-servers* [dns-serv-id]
| | | +--ro dns-serv-id            uint8
| | | +--ro dns-serv-addr          inet:ipv6-address
| +--ro sip-paras
| | +--ro sip-servers* [sip-serv-id]
| | | +--ro sip-serv-id            uint8
| | | +--ro sip-serv-addr          inet:ipv6-address
| | | +--ro sip-serv-domain-name   string
| +--ro nis-paras

```



```
| | +--ro nis-domain-name      string
| | +--ro nis-server* [nis-serv-id]
| |   +--ro nis-serv-id      uint8
| |   +--ro nis-serv-addr    inet:ipv6-address
| +--ro nis-plus-paras
| |   +--ro nis-plus-domain-name  string
| |   +--ro nis-plus-server* [nis-plus-serv-id]
| |     +--ro nis-plus-serv-id    uint8
| |     +--ro nis-plus-serv-addr  inet:ipv6-address
| +--ro info-refresh-time      yang:timeticks
| +--ro time-zone-paras
| |   +--ro tz-posix          string
| |   +--ro tz-database       string
| +--ro cli-fqdn              string
| +--ro ntp-paras
| |   +--ro ntp-server* [ntp-serv-id]
| |     +--ro ntp-serv-id      uint8
| |     +--ro ntp-serv-addr    inet:ipv6-address
| |     +--ro ntp-serv-mul-addr inet:ipv6-address
| |     +--ro ntp-serv-fqdn    string
| +--ro sntp-paras
| |   +--ro sntp-server* [sntp-serv-id]
| |     +--ro sntp-serv-id      uint8
| |     +--ro sntp-serv-addr    inet:ipv6-address
| +--ro network-boot-paras
| |   +--ro boot-file* [boot-file-id]
| |     +--ro boot-file-id      uint8
| |     +--ro suitable-arch-type* uint16
| |     +--ro suitable-net-if*  uint32
| |     +--ro boot-file-url     string
| |     +--ro boot-file-paras* [para-id]
| |       +--ro para-id         uint8
| |       +--ro parameter       string
| +--ro kerberos-paras
| |   +--ro default-realm-name  string
| |   +--ro kdc-info* [kdc-id]
| |     +--ro kdc-id           uint8
| |     +--ro priority         uint16
| |     +--ro weight           uint16
| |     +--ro transport-type    uint8
| |     +--ro port-number       uint16
| |     +--ro kdc-ipv6-addr     inet:ipv6-address
| |     +--ro realm-name       string
| +--ro addr-selection-paras
| |   +--ro automatic-row-add   boolean
| |   +--ro prefer-temporary-addr boolean
| |   +--ro policy-table* [policy-id]
| |     +--ro policy-id        uint8
```



```

| |      +--ro label                               uint8
| |      +--ro precedence                           uint8
| |      +--ro prefix-len                           uint8
| |      +--ro prefix                               inet:ipv6-prefix
| +--ro sol-max-rt                                 yang:timeticks
| +--ro inf-max-rt                                 yang:timeticks
| +--ro pcg-paras
| |  +--ro pcg-server* [pcg-serv-id]
| |      +--ro pcg-serv-id                           uint8
| |      +--ro pcg-serv-addr                         inet:ipv6-address
| +--ro s46-rule-paras
| |  +--ro s46-rule* [rule-id]
| |      +--ro rule-id                               uint8
| |      +--ro rule-type                             enumeration
| |      +--ro ea-len                                 uint8
| |      +--ro prefix4-len                           uint8
| |      +--ro ipv4-prefix                           inet:ipv4-prefix
| |      +--ro prefix6-len                           uint8
| |      +--ro ipv6-prefix                           inet:ipv6-prefix
| |      +--ro port-parameter
| |          +--ro offset                             uint8
| |          +--ro psid-len                           uint8
| |          +--ro psid                               uint16
| +--ro s46-br-paras
| |  +--ro br* [br-id]
| |      +--ro br-id                                 uint8
| |      +--ro br-ipv6-addr                         inet:ipv6-address
| +--ro s46-dmr-paras
| |  +--ro dmr* [dmr-id]
| |      +--ro dmr-id                                 uint8
| |      +--ro dmr-prefix6-len                       uint8
| |      +--ro dmr-ipv6-prefix                       inet:ipv6-prefix
| +--ro s46-v4-v6-binding-paras
| |  +--ro ipv4-addr                                 inet:ipv6-address
| |  +--ro bind-prefix6-len                           uint8
| |  +--ro port-parameter
| |      +--ro offset                                 uint8
| |      +--ro psid-len                               uint8
| |      +--ro psid                                 uint16
| +--ro erp-local-domain-name                       string
+--ro supported-options
|  +--ro supported-option* [option-code]
|      +--ro option-code                             uint16
|      +--ro description                             string
+--ro packet-stats
    +--ro solicit-count                             uint32
    +--ro request-count                             uint32
    +--ro renew-count                               uint32

```


++ro rebind-count	uint32
++ro decline-count	uint32
++ro release-count	uint32
++ro info-req-count	uint32
++ro advertise-count	uint32
++ro confirm-count	uint32
++ro reply-count	uint32
++ro reconfigure-count	uint32

Introduction of important nodes:

- o client-if: A client may have several interfaces, it is more reasonable to configure and manage parameters on the interface-level. The list defines a specific client interface and its data. Different interfaces are distinguished by the "ifName" key which is a configurable string value.
- o duid: Each server and client has only one DUID (DHCP Unique Identifier). The DUID here will be carried in the Client ID option to identify a specific DHCPv6 client. This leaf are same as the "duid" leaf in "dhcpv6-server" feature.
- o pd-function: Whether the client can act as a requesting router to request prefixes using prefix delegation ([RFC3633](#)).
- o rapid-commit: '1' indicates a client can initiate a Solicit-Reply message exchange by adding a Rapid Commit option in Solicit message. '0' means the client is not allowed to add a Rapid Commit option to request addresses in a two-message exchange pattern.
- o mo-tab: The management tab label indicates the operation mode of the DHCPv6 client. 'm'=1 and 'o'=1 indicate the client will use DHCPv6 to obtain all the configuration data. 'm'=1 and 'o'=0 are a meaningless combination. 'm'=0 and 'o'=1 indicate the client will use stateless DHCPv6 to obtain configuration data apart from addresses/prefixes data. 'm'=0 and 'o'=0 represent the client will not use DHCPv6 but use SLAAC to achieve configuration.
- o oro-options: This container provide a way to configure the list of options that the client will request in its ORO option.
- o client-configured-options: Similar to the server, the client also need to configure some options to fulfil some desired functions. This container include all the potential options that need to be configured at the client side. The relevant RFCs that define those options include: [RFC3315](#), [RFC4704](#), [RFC5970](#), [RFC6784](#), [RFC6939](#).

- o identity-association: IA is a construct through which a server and a client can identify, group, and manage a set of related IPv6 addresses. The key of the "identity-association" list is a 4-byte number IAID defined in [[RFC3315](#)].
- o if-other-paras: A client can obtain extra configuration data other than address and prefix information through DHCPv6 options. This container describes such data the client was configured through DHCPv6. The potential configuration data may include DNS server parameters, SIP server parameters and etc.
- o supported-options: This state data container defines a list of options supported by the client for administrator to interrogate a client's capabilities.
- o packet-stats: A container records all the packet status information of a specific interface.

[2.4.](#) Notifications Mechanism for DHCPv6


```

+--rw dhcpv6
  +-- ...
  |
  +--n notifications
    +--n dhcpv6-server-event
      | +--n pool-running-out
      | | +--ro utilization-ratio          uint16
      | | +--ro duid                      duidtype
      | | +--ro serv-name?                 string
      | | +--ro pool-name                  string
      | +--n invalid-client-detected
      |   +--ro duid                      duidtype
      |   +--ro description?              string
    +--n dhcpv6-relay-event
      | +--n topo-changed
      |   +--ro relay-if-name              string
      |   +--ro first-hop                  boolean
      |   +--ro last-entity-addr           inet:ipv6-address
    +--n dhcpv6-client-event
      +--n ia-lease-event
        | +--ro event-type                 enumeration
        | +--ro duid                      duidtype
        | +--ro ia-id                      uint32
        | +--ro serv-name?                 string
        | +--ro description?               string
      +--n invalid-ia-detected
        | +--ro duid                      duidtype
        | +--ro ia-id                      uint32
        | +--ro serv-name?                 string
        | +--ro description?               string
      +--n retransmission-failed
        | +--ro duid                      duidtype
        | +--ro description                 enumeration
      +--n failed-status-turn-up
        +--ro duid                      duidtype
        +--ro status-code                 enumeration

```

Introduction of notifications:

- o pool-running-out: raised when the address/prefix pool is going to run out. A threshold for utilization ratio of the pool has been defined in the server feature so that it will notify the administrator when the utilization ratio reaches the threshold, and such threshold is a settable parameter.
- o invalid-client-detected: raised when the server has found a client which can be regarded as a potential attacker. Some description could also be included.

- o topo-changed: raised when the topology of the relay agent is changed.
- o ia-lease-event: raised when the client was allocated a new IA from the server or it renew/rebind/release its current IA.
- o invalid-ia-detected: raised when the identity association of the client can be proved to be invalid. Possible condition includes duplicated address, illegal address, etc.
- o retransmission-failed: raised when the retransmission mechanism defined in [[RFC3315](#)] is failed.
- o failed-status-turn-up: raised when the client receives a message includes an unsuccessful Status Code option.

3. DHCPv6 YANG Model

This module imports typedefs from [[RFC6991](#)].

```
<CODE BEGINS> file "ietf-dhcpv6@2015-10-16.yang"
module ietf-dhcpv6 {
  namespace "urn:ietf:params:xml:ns:yang:ietf-dhcpv6";
  prefix "dhcpv6";

  import ietf-inet-types {
    prefix inet;
    revision-date "2013-07-15";
  }
  import ietf-yang-types {
    prefix yang;
    revision-date "2013-07-15";
  }

  organization "dhc wg";
  contact "yong@csnet1.cs.tsinghua.edu.cn
    wangh13@mails.tsinghua.edu.cn
    lh.sunlinh@gmail.com
    Ted.Lemon@nominum.com
    ian.farrer@telekom.de";

  description "This model defines a YANG data model that can be
    used to configure and manage DHCPv6 server, DHCPv6 relay and
    DHCPv6 client.";

  revision 2015-10-16 {
    description "version05: Omit the feature statement. Modify
      model according to comments from Huawei. Also correct some
```



```
        grammar errors.";

        reference "I-D: draft-ietf-dhc-dhcpv6-yang-00";
    }
    revision 2015-09-25 {
        description "version04: Correct duid and grammar errors.";

        reference "I-D: draft-cui-dhc-dhcpv6-yang-04";
    }
    revision 2015-07-01 {
        description "version03: Correct grammar errors.";

        reference "I-D: draft-cui-dhc-dhcpv6-yang-03";
    }
    revision 2015-04-13 {
        description "version02: Correct grammar errors.";

        reference "I-D: draft-cui-dhc-dhcpv6-yang-02";
    }
    revision 2015-04-02 {
        description "version01: Correct grammar errors, Reuse
            groupings, Update 'dhcpv6-relay' feature, Add
            notifications.";

        reference "I-D: draft-cui-dhc-dhcpv6-yang-01";
    }
    revision 2015-03-04 {
        description "version00: Initial revision.";

        reference "I-D: draft-cui-dhc-dhcpv6-yang-00";
    }
}

/*
 * Typedef
 */

typedef duidtype {
    type union {
        type uint16;
        type string {
            pattern '([0-9a-fA-F]{2}){2,128}';
        }
    }
    description "the type defined for duid";
}

typedef threshold {
```



```
    type union {
      type uint16 {
        range 0..100;
      }
      type enumeration {
        enum "disabled" {
          description "No threshold";
        }
      }
    }
    description "Threshold value in percent";
  }
}

/*
 * Grouping
 */

grouping vendor-infor {
  description "vendor info";
  container vendor-info {
    description "";
    leaf ent-num {
      type uint32;
      mandatory true;
      description "enterprise number";
    }
    leaf-list data {
      type string;
      description "specific vendor info";
    }
  }
}

grouping portset-para {
  description "portset parameters";
  container port-parameter {
    description "port parameter";
    leaf offset {
      type uint8;
      mandatory true;
      description "offset in a port set";
    }
    leaf psid-len {
      type uint8;
      mandatory true;
      description "length of a psid";
    }
    leaf psid {
```



```
        type uint16;
        mandatory true;
        description "psid value";
    }
}

/*
 * Data Nodes
 */

container server {
    description "dhcpv6 server portion";
    container serv-attributes {
        description "This container contains basic attributes
        of a DHCPv6 server such as DUID, server name and so
        on. Some optional functions that can be provided by
        the server is also included.";
        leaf name {
            type string;
            mandatory true;
            description "server's name";
        }
        leaf duid {
            type duidtype;
            mandatory true;
            description "DHCP Unique Identifier";
        }
        leaf enable {
            type boolean;
            mandatory true;
            description "whether to enable the server";
        }
        leaf ipv6-address {
            type inet:ipv6-address;
            description "server's IPv6 address";
        }
        leaf description {
            type string;
            description "description of the server";
        }
        leaf pd-function {
            type boolean;
            mandatory true;
            description "Whether the server can act as a
            delegating router to perform prefix delegation
            ([RFC3633]).";
        }
    }
}
```



```
leaf stateless-service {
    type boolean;
    mandatory true;
    description "A boolean value specifies whether
the server support client-server exchanges
involving two messages defined in ([RFC3315]).";
}
leaf rapid-commit {
    type boolean;
    mandatory true;
    description "A boolean value specifies whether
the server support client-server exchanges
involving two messages defined in ([RFC3315]).";
}
leaf-list interfaces-config {
    type string;
    description "A leaf list to denote which one or
more interfaces the server should listen on. The
default value is to listen on all the interfaces.
This node is also used to set a unicast address
for the server to listen with a specific interface.
For example, if people want the server to listen
on a unicast address with a specific interface, he
can use the format like 'eth1/2001:db8::1'.";
}
uses vendor-infor;
}
container option-sets {
    description "DHCPv6 employs various options to carry
additional information and parameters in DHCP messages.
This container defines all the possible options that
need to be configured at the server side.";
    list option-set {
        key option-set-id;
        description "A server may allow different option
sets to be configured for different conditions
(i.e. different networks, clients and etc). This
'option-set' list enables various sets of options
being defined and configured in a single server.
Different sets are distinguished by the key called
'option-set-id'. All the possible options discussed
above are defined in the list and each option is
corresponding to a container.";
        leaf option-set-id {
            type leafref; {
                path "/dhcpv6/server/option-sets/option-set/option-set-id";
            }
            mandatory true;
            description "the option-set-id key";
        }
    }
}
```



```
    }  
  }  
  list new-or-standard-option {  
    key option-code;  
    description "new or standard option";  
    leaf option-code {  
      type uint16;  
      mandatory true;  
      description "the option code key";  
    }  
    leaf option-name {  
      type string;  
      mandatory true;  
      description "the new option's name";  
    }  
    leaf option-description {  
      type string;  
      mandatory true;  
      description "description of new option";  
    }  
    leaf option-reference {  
      type string;  
      description "reference to the  
specification";  
    }  
    leaf option-value {  
      type string;  
      mandatory true;  
      description "the new option's value";  
    }  
  }  
  leaf user-class-value {  
    type string;  
    description "use class option's value";  
  }  
  leaf enterprise-number {  
    type uint32;  
    description "enterprise number";  
  }  
  leaf store-client-link-layer {  
    type boolean;  
    description "whether to store client's  
link layer address";  
  }  
  container preference-option {  
    description "preference option";  
    leaf enable {  
      type boolean;
```



```
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    leaf preference-value {
        type uint8;
        mandatory true;
        description "the value for this option";
    }
}
container sip-server-option {
    description "sip server option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    list sip-server {
        key sip-serv-id;
        description "sip server info";
        leaf sip-serv-id {
            type uint8;
            mandatory true;
            description "sip server id";
        }
        leaf sip-serv-domain-name {
            type string;
            mandatory true;
            description "sip serevr domain
            name";
        }
        leaf sip-serv-addr {
            type inet:ipv6-address;
            mandatory true;
            description "sip server addr";
        }
    }
}
container dns-config-option {
    description "dns configuration option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be included in the
```



```
        option set";
    }
    list dns-server {
        key dns-serv-id;
        description "dns server info";
        leaf dns-serv-id {
            type uint8;
            mandatory true;
            description "dns server id";
        }
        leaf dns-serv-addr {
            type inet:ipv6-address;
            mandatory true;
            description "dns server addr";
        }
    }
    leaf domain-search-list {
        type string;
        mandatory true;
        description "search list";
    }
}
container nis-config-option {
    description "nis configurtaion option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list nis-server {
        key nis-serv-id;
        description "nis server info";
        leaf nis-serv-id {
            type uint8;
            mandatory true;
            description "nis server id";
        }
        leaf nis-serv-addr {
            type inet:ipv6-address;
            mandatory true;
            description "nis server addr";
        }
    }
}
container nis-plus-config-option {
    description "nisp configuration option";
```



```
    leaf enable {
      type boolean;
      mandatory true;
      description "indicate whether this
        option will be included in the
        option set";
    }
    list nis-plus-server {
      key nis-plus-serv-id;
      description "nisp server info";
      leaf nis-plus-serv-id {
        type uint8;
        mandatory true;
        description "nisp server id";
      }
      leaf nis-plus-serv-addr {
        type inet:ipv6-address;
        mandatory true;
        description "nisp server addr";
      }
    }
  }
  container info-refresh-time-option {
    description "info refresh time option";
    leaf enable {
      type boolean;
      mandatory true;
      description "indicate whether this
        option will be included in the
        option set";
    }
    leaf info-refresh-time {
      type yang:timeticks;
      mandatory true;
      description "the refresh time";
    }
  }
  container cli-fqdn-option {
    description "client fqdn option";
    leaf enable {
      type boolean;
      mandatory true;
      description "indicate whether this
        option will be included in the
        option set";
    }
    leaf server-initiate-update {
      type boolean;
```



```
        mandatory true;
        description "server initiate";
    }
    leaf client-initiate-update {
        type boolean;
        mandatory true;
        description "client initiate";
    }
    leaf modify-name-from-cli {
        type boolean;
        mandatory true;
        description "modify by client";
    }
}
container timezone-option {
    description "timezone option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    leaf tz-posix {
        type string;
        mandatory true;
        description "tz posix";
    }
    leaf tz-database {
        type string;
        mandatory true;
        description "tz database";
    }
}
container ntp-server-option {
    description "ntp server option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list ntp-server {
        key ntp-serv-id;
        description "ntp server info";
        leaf ntp-serv-id {
            type uint8;
```



```
        mandatory true;
        description "ntp server id";
    }
    leaf ntp-serv-addr {
        type inet:ipv6-address;
        mandatory true;
        description "ntp server addr";
    }
    leaf ntp-serv-mul-addr {
        type inet:ipv6-address;
        mandatory true;
        description "ntp server multicast
        addr";
    }
    leaf ntp-serv-fqdn {
        type string;
        mandatory true;
        description "ntp server fqdn";
    }
}

container sntp-server-option {
    description "sntp server option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    list sntp-server {
        key sntp-serv-id;
        description "sntp server info";
        leaf sntp-serv-id {
            type uint8;
            mandatory true;
            description "sntp server id";
        }
        leaf sntp-serv-addr {
            type inet:ipv6-address;
            mandatory true;
            description "sntp server addr";
        }
    }
}

container network-boot-option {
    description "network boot option";
    leaf enable {
```



```
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list boot-file {
        key boot-file-id;
        description "boot file info";
        leaf boot-file-id {
            type uint8;
            mandatory true;
            description "boot file id";
        }
        leaf-list suitable-arch-type {
            type uint16;
            description "architecture type";
        }
        leaf-list suitable-net-if {
            type uint32;
            description "network interface";
        }
        leaf boot-file-url {
            type string;
            mandatory true;
            description "url for boot file";
        }
        list boot-file-paras {
            key para-id;
            description "boot file parameters";
            leaf para-id {
                type uint8;
                mandatory true;
                description "parameter id";
            }
            leaf parameter {
                type string;
                mandatory true;
                description "parameter
value";
            }
        }
    }
}

container dslite-option {
    description "dslite option";
    leaf enable {
        type boolean;
```



```
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    leaf dslite-aftr-name {
        type string;
        mandatory true;
        description "aftr name";
    }
}
container kerberos-option {
    description "kerberos option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    leaf default-realm-name {
        type string;
        mandatory true;
        description "default realm name";
    }
}
list kdc-info {
    key kdc-id;
    description "kdc info";
    leaf kdc-id {
        type uint8;
        mandatory true;
        description "kdc id";
    }
    leaf priority {
        type uint16;
        mandatory true;
        description "priority";
    }
    leaf weight {
        type uint16;
        mandatory true;
        description "weight";
    }
    leaf transport-type {
        type uint8;
        mandatory true;
        description "transport type";
    }
}
```



```
    leaf port-number {
      type uint16;
      mandatory true;
      description "port number";
    }
    leaf kdc-ipv6-addr {
      type inet:ipv6-address;
      mandatory true;
      description "kdc ipv6 addr";
    }
    leaf realm-name {
      type string;
      mandatory true;
      description "realm name";
    }
  }
}

container addr-selection-option {
  description "address selection option";
  leaf enable {
    type boolean;
    mandatory true;
    description "indicate whether this
option will be included in the
option set";
  }
  leaf a-bit-set {
    type boolean;
    mandatory true;
    description "a bit";
  }
  leaf p-bit-set {
    type boolean;
    mandatory true;
    description "p bit";
  }
}

list policy-table {
  key policy-id;
  description "policy table";
  leaf policy-id {
    type uint8;
    mandatory true;
    description "policy id";
  }
  leaf label {
    type uint8;
    mandatory true;
    description "label";
  }
}
```



```
    }
    leaf precedence {
        type uint8;
        mandatory true;
        description "precedence";
    }
    leaf prefix-len {
        type uint8;
        mandatory true;
        description "prefix length";
    }
    leaf prefix {
        type inet:ipv6-prefix;
        mandatory true;
        description "prefix";
    }
}

}
container sol-max-rt-option {
    description "sol max rt option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    leaf sol-max-rt-value {
        type yang:timeticks;
        mandatory true;
        description "sol max rt value";
    }
}

}
container inf-max-rt-option {
    description "inf max rt option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be included in the
        option set";
    }
    leaf inf-max-rt-value {
        type yang:timeticks;
        mandatory true;
        description "inf max rt value";
    }
}

}
```



```
container pcg-server-option {
  description "pcg server option";
  leaf enable {
    type boolean;
    mandatory true;
    description "indicate whether this
option will be included in the
option set";
  }
  list pcg-server {
    key pcg-serv-id;
    description "pcg server info";
    leaf pcg-serv-id {
      type uint8;
      mandatory true;
      description "pcg server id";
    }
    leaf pcg-serv-addr {
      type inet:ipv6-address;
      mandatory true;
      description "pcg server addr";
    }
  }
}

container s46-rule-option {
  description "s46 rule option";
  leaf enable {
    type boolean;
    mandatory true;
    description "indicate whether this
option will be included in the
option set";
  }
  list s46-rule {
    key rule-id;
    description "s46 rule";
    leaf rule-id {
      type uint8;
      mandatory true;
      description "rule id";
    }
    leaf rule-type {
      type enumeration {
        enum "BMR" {
          description "BMR";
        }
        enum "FMR" {
          description "FMR";
        }
      }
    }
  }
}
```



```
    }
    }
    mandatory true;
    description "rule type";
  }
  leaf prefix4-len {
    type uint8;
    mandatory true;
    description "ipv4 prefix length";
  }
  leaf ipv4-prefix {
    type inet:ipv4-prefix;
    mandatory true;
    description "ipv4 prefix";
  }
  leaf prefix6-len {
    type uint8;
    mandatory true;
    description "ipv6 prefix length";
  }
  leaf ipv6-prefix {
    type inet:ipv6-prefix;
    mandatory true;
    description "ipv6 prefix";
  }
  }
  uses portset-para;
}

}
container s46-br-option {
  description "s46 br option";
  leaf enable {
    type boolean;
    mandatory true;
    description "indicate whether this
    option will be included in the
    option set";
  }
}
list br {
  key br-id;
  description "br info";
  leaf br-id {
    type uint8;
    mandatory true;
    description "br id";
  }
}
leaf br-ipv6-addr {
  type inet:ipv6-address;
  mandatory true;
```



```
        description "br ipv6 addr";
    }
}

container operator-option-ipv6-address {
    description "operator ipv6 address option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list operator-ipv6-addr {
        key operator-ipv6-addr-id;
        description "operator ipv6 address info";
        leaf operator-ipv6-addr-id {
            type uint8;
            mandatory true;
            description "operator ipv6 address id";
        }
        leaf operator-ipv6-addr {
            type inet:ipv6-address;
            mandatory true;
            description "operator ipv6 address id";
        }
    }
}

container operator-option-single-flag {
    description "operator single flag";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list flag {
        key flag-id;
        description "operator single flag info";
        leaf flag-id {
            type uint8;
            mandatory true;
            description "operator single flag id";
        }
        leaf flag-value{
            type boolean;
            mandatory true;
        }
    }
}
```



```
        description "operator single flag value";
    }
}

container operator-option-ipv6-prefix {
    description "operator ipv6 prefix option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list operator-ipv6-prefix{
        key operator-ipv6-prefix-id;
        description "operator ipv6 prefix info";
        leaf operator-ipv6-prefix-id {
            type uint8;
            mandatory true;
            description "operator ipv6 prefix id";
        }
        leaf operator-ipv6-prefix6-len{
            type uint8;
            mandatory true;
            description "operator ipv6 prefix length";
        }
        leaf operator-ipv6-prefix {
            type inet:ipv6-prefix;
            mandatory true;
            description "operator ipv6 prefix";
        }
    }
}

container operator-option-int32 {
    description "operator integer 32 option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list int32val{
        key int32val-id;
        description "operator integer 32 info";
        leaf int32val-id {
            type uint8;
            mandatory true;
        }
    }
}
```



```
        description "operator integer 32 id";
    }
    leaf int32val{
        type uint32;
        mandatory true;
        description "operator integer 32 value";
    }
}

container operator-option-int16 {
    description "operator integer 16 option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list int16val{
        key int16val-id;
        description "operator integer 16 info";
        leaf int16val-id {
            type uint8;
            mandatory true;
            description "operator integer 16 id";
        }
        leaf int16val{
            type uint16;
            mandatory true;
            description "operator integer 16 value";
        }
    }
}

container operator-option-int8 {
    description "operator integer 8 option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list int8val{
        key int32val-id;
        description "operator integer 8 info";
        leaf int8val-id {
            type uint8;
            mandatory true;
```



```
        description "operator integer 8 id";
    }
    leaf int8val{
        type uint8;
        mandatory true;
        description "operator integer 8 value";
    }
}
container operator-option-uri {
    description "operator uri option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list uri{
        key uri-id;
        description "operator uri info";
        leaf uri-id {
            type uint8;
            mandatory true;
            description "operator uri id";
        }
        leaf uri{
            type string;
            mandatory true;
            description "operator uri value";
        }
    }
}
container operator-option-textstring {
    description "operator itext string option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list textstring{
        key textstring-id;
        description "operator text string info";
        leaf textstring-id {
            type uint8;
            mandatory true;
        }
    }
}
```



```
        description "operator text string id";
    }
    leaf textstring{
        type string;
        mandatory true;
        description "operator text string value";
    }
}
container operator-option-var-data {
    description "operator variable length data option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list int32val{
        key var-data-id;
        description "operator ivariable length data info";
        leaf var-data-id {
            type uint8;
            mandatory true;
            description "operator variable length id";
        }
        leaf var-data{
            type binary;
            mandatory true;
            description "operator variable length value";
        }
    }
}
container operator-option-dns-wire {
    description "operator dns wire format domain name list
option";

    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list operator-option-dns-wire{
        key operator-option-dns-wire-id;
        description "operator dns wire format info";
        leaf operator-option-dns-wire-id {
            type uint8;
```

mandatory true;

Cui, et al.

Expires September 22, 2016

[Page 43]

```
        description "operator dns wire format id";
    }
    leaf operator-option-dns-wire{
        type binary;
        mandatory true;
        description "operator dns wire format value";
    }
}
container s46-dmr-option {
    description "s46 dmr option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list dmr {
        key dmr-id;
        description "dmr info";
        leaf dmr-id {
            type uint8;
            mandatory true;
            description "dmr id";
        }
        leaf dmr-prefix-len {
            type uint8;
            mandatory true;
            description "dmr prefix length";
        }
        leaf dmr-ipv6-prefix {
            type inet:ipv6-prefix;
            mandatory true;
            description "dmr ipv6 prefix";
        }
    }
}
container s46-v4-v6-binding-option {
    description "s46 binding option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be included in the
option set";
    }
    list ce {
```



```

        key ce-id;
        description "ce info";
        leaf ce-id {
            type uint8;
            mandatory true;
            description "ce id";
        }
        leaf ipv4-addr {
            type inet:ipv4-address;
            mandatory true;
            description "ce ipv4 addr";
        }
        leaf bind-prefix6-len {
            type uint8;
            mandatory true;
            description "bind ipv6 prefix
length";
        }
        leaf bind-ipv6-prefix {
            type inet:ipv6-prefix;
            mandatory true;
            description "bind ipv6 prefix";
        }
        uses portset-para;
    }
}

}

}

container network-ranges {
    description "This model supports a hierarchy
to achieve dynamic configuration. That is to
say we could configure the server at different
levels through this model. The top level is a
global level which is defined as the container
'network-ranges'. The following levels are
defined as sub-containers under it. The
'network-ranges' contains the parameters
(e.g. option-sets) that would be allocated to
all the clients served by this server.";
    leaf-list option-set {
        type uint8;
        description "a specific option set";
    }
    list network-range {
        key network-range-id;
        description "Under the 'network-ranges'
container, a 'network-range' list is
defined to configure the server at a

```



```
network level which is also considered
as the second level. Different network
are identified by the key 'network-range-id'.
This is because a server may have different
configuration parameters (e.g. option sets)
for different networks.";
leaf network-range-id {
    type uint8;
    mandatory true;
    description "equivalent to subnet id";
}
leaf network-description {
    type string;
    mandatory true;
    description "description of the subnet";
}
leaf network-prefix {
    type inet:ipv6-prefix;
    mandatory true;
    description "subnet prefix";
}
leaf inherit-option-set {
    type boolean;
    mandatory true;
    description "indicate whether to inherit
the configuration from higher level";
}
leaf-list option-set {
    type uint8;
    description "configured option set";
}
container address-pools {
    description "A container describes
the DHCPv6 server's address pools.";
    list address-pool {
        key pool-id;
        description "A DHCPv6 server can
be configured with several address
pools. This list defines such
address pools which are distinguish
by the key called 'pool-name'.";
        leaf pool-id {
            type uint8;
            mandatory true;
            description "pool id";
        }
        leaf pool-prefix {
            type inet:ipv6-prefix;
```



```
        mandatory true;
        description "pool prefix";
    }
    leaf start-address {
        type inet:ipv6-address-no-zone;
        mandatory true;
        description "start address";
    }
    leaf end-address {
        type inet:ipv6-address-no-zone;
        mandatory true;
        description "end address";
    }
    leaf renew-time {
        type yang:timeticks;
        mandatory true;
        description "renew time";
    }
    leaf rebind-time {
        type yang:timeticks;
        mandatory true;
        description "rebind time";
    }
    leaf preferred-lifetime {
        type yang:timeticks;
        mandatory true;
        description "preferred lifetime
for IA";
    }
    leaf valid-lifetime {
        type yang:timeticks;
        mandatory true;
        description "valid lifetime for IA";
    }
    leaf total-ipv6-count {
        type uint64;
        config "false";
        mandatory true;
        description "how many ipv6
addressses are in the pool";
    }
    leaf used-ipv6-count {
        type uint64;
        config "false";
        mandatory true;
        description "how many are
allocated";
    }
}
```



```
leaf utilization-ratio {
    type threshold;
    mandatory true;
    description "the utilization ratio";
}
leaf inherit-option-set {
    type boolean;
    mandatory true;
    description "indicate whether to
inherit the configuration from
higher level";
}
leaf-list option-set {
    type uint8;
    description "configured option
set";
}
container reserved-addresses {
    description "reserved addresses";
    list static-binding {
        key cli-id;
        description "static binding of
reserved addresses";
        leaf cli-id {
            type uint32;
            mandatory true;
            description "client id";
        }
        leaf duid {
            type duidtype;
            mandatory true;
            description "DHCP Unique
Identifier";
        }
        leaf-list reserv-addr {
            type inet:ipv6-address;
            description "reserved addr";
        }
    }
    leaf-list other-reserv-addr {
        type inet:ipv6-address;
        description "other reserved
addr";
    }
}
list binding-info {
    key cli-id;
```



```
    config "false";
    description "A list records a binding
    information for each DHCPv6 client
    that has already been allocated IPv6
    addresses.";
    leaf cli-id {
        type uint32;
        mandatory true;
        description "client id";
    }
    leaf duid {
        type duidtype;
        mandatory true;
        description "DHCP Unique Identifier";
    }
    list cli-ia {
        key ia-id;
        description "client IA";
        leaf ia-type {
            type string;
            mandatory true;
            description "IA type";
        }
        leaf ia-id {
            type uint32;
            mandatory true;
            description "IAID";
        }
        leaf-list cli-addr {
            type inet:ipv6-address;
            description "client addr";
        }
        leaf pool-id {
            type uint8;
            mandatory true;
            description "pool id";
        }
    }
}

container prefix-pools {
    description "If a server supports prefix
    delegation function, this container will
    be used to define the delegating router's
    prefix pools.";
    list prefix-pool {
        key pool-id;
        description "Similar to server's
```



```
address pools, a delegating router
can also be configured with multiple
prefix pools specified by a list
called 'prefix-pool'.";
leaf pool-id {
    type uint8;
    mandatory true;
    description "pool id";
}
leaf prefix {
    type inet:ipv6-prefix;
    mandatory true;
    description "ipv6 prefix";
}
leaf prefix-length {
    type uint8;
    mandatory true;
    description "prefix length";
}
leaf renew-time {
    type yang:timeticks;
    mandatory true;
    description "renew time";
}
leaf rebind-time {
    type yang:timeticks;
    mandatory true;
    description "rebind time";
}
leaf preferred-lifetime {
    type yang:timeticks;
    mandatory true;
    description "preferred lifetime for
    IA";
}
leaf valid-lifetime {
    type yang:timeticks;
    mandatory true;
    description "valid lifetime for IA";
}
leaf utilization-ratio {
    type threshold;
    mandatory true;
    description "utilization ratio";
}
leaf inherit-option-set {
    type boolean;
    mandatory true;
```



```
        description "whether to inherit
        configuration from higher level";
    }
    leaf-list option-set {
        type uint8;
        description "configured option
        set";
    }
    container reserved-prefixes {
        description "reserved prefixes";
        list static-binding {
            key cli-id;
            description "static binding";
            leaf cli-id {
                type uint32;
                mandatory true;
                description "client id";
            }
            leaf duid {
                type duidtype;
                mandatory true;
                description "DHCP Unique
                Identifier";
            }
            leaf reserv-prefix-len {
                type uint8;
                mandatory true;
                description "reserved
                prefix length";
            }
            leaf reserv-prefix {
                type inet:ipv6-prefix;
                mandatory true;
                description
                "reserved prefix";
            }
        }
    }
    leaf exclude-prefix-len {
        type uint8;
        mandatory true;
        description "exclude prefix
        length";
    }
    leaf exclude-prefix {
        type inet:ipv6-prefix;
        mandatory true;
        description "exclude prefix";
    }
}
```



```
    list other-reserv-prefix {
      key reserv-id;
      description
        "other reserved prefix";
      leaf reserv-id {
        type uint8;
        mandatory true;
        description
          "reserved prefix id";
      }
      leaf prefix-len {
        type uint8;
        mandatory true;
        description "prefix length";
      }
      leaf prefix {
        type inet:ipv6-prefix;
        mandatory true;
        description
          "reserved prefix";
      }
    }
  }
}

list binding-info {
  key cli-id;
  config "false";
  description "A list records a
binding information for each
DHCPv6 client that has already
been allocated IPv6 addresses.";
  leaf cli-id {
    type uint32;
    mandatory true;
    description "client id";
  }
  leaf duid {
    type duidtype;
    mandatory true;
    description "DHCP Unique
Identifier";
  }
}

list cli-iapd {
  key iaaid;
  description "client IAPD";
  leaf iaaid {
    type uint32;
    mandatory true;
  }
}
```



```
        description "IAID";
    }
    leaf-list cli-prefix {
        type inet:ipv6-prefix;
        description
            "client ipv6 prefix";
    }
    leaf-list cli-prefix-len {
        type uint8;
        description
            "client prefix length";
    }
    leaf pool-id {
        type uint8;
        mandatory true;
        description "pool id";
    }
}

}

}
container hosts {
    description "hosts level";
    list host {
        key cli-id;
        description "specific host";
        leaf cli-id {
            type uint32;
            mandatory true;
            description "client id";
        }
        leaf duid {
            type duidtype;
            mandatory true;
            description "DHCP Unique
                Identifier";
        }
        leaf inherit-option-set {
            type boolean;
            mandatory true;
            description "whether to inherit
                configuration
                from higher level";
        }
        leaf-list option-set {
            type uint8;
            description "configured option set";
        }
        leaf nis-domain-name {
```



```

        type string;
        description "nis domain name";
    }
    leaf nis-plus-domain-name {
        type string;
        description "nisp domain name";
    }
}

}

}

container relay-opaque-params {
    description "This container contains some opaque values in Relay Agent options that need to be configured on the server side only for value match. Such Relay Agent options include Interface-Id option, Remote-Id option and Subscriber-Id option.";
    list relays {
        key relay-name;
        description "relay agents";
        leaf relay-name {
            type string;
            mandatory true;
            description "relay agent name";
        }
        list interface-info {
            key if-name;
            description "interface info";
            leaf if-name {
                type string;
                mandatory true;
                description "interface name";
            }
            leaf interface-id {
                type string;
                mandatory true;
                description "interface id";
            }
        }
        list subscribers {
            key subscriber;
            description "subscribers";
            leaf subscriber {
                type uint8;
                mandatory true;
                description "subscriber";
            }
        }
    }
}

```



```
        leaf subscriber-id {
            type string;
            mandatory true;
            description "subscriber id";
        }
    }
    list remote-host {
        key ent-num;
        description "remote host";
        leaf ent-num {
            type uint32;
            mandatory true;
            description "enterprise number";
        }
        leaf remote-id {
            type string;
            mandatory true;
            description "remote id";
        }
    }
}

container rsoo-enabled-options {
    description "rsoo enabled options";
    list rsoo-enabled-option{
        key option-code;
        description "rsoo enabled option";
        leaf option-code {
            type uint16;
            mandatory true;
            description "option code";
        }
        leaf description {
            type string;
            mandatory true;
            description "description of the option";
        }
    }
}

container packet-stats {
    config "false";
    description "A container presents
the packet statistics related to
the DHCPv6 server.";
    leaf solicit-count {
        type uint32;
        mandatory true;
        description "solicit counter";
    }
}
```



```
}
leaf request-count {
    type uint32;
    mandatory true;
    description "request counter";
}
leaf renew-count {
    type uint32;
    mandatory true;
    description "renew counter";
}
leaf rebind-count {
    type uint32;
    mandatory true;
    description "rebind counter";
}
leaf decline-count {
    type uint32;
    mandatory true;
    description "decline count";
}
leaf release-count {
    type uint32;
    mandatory true;
    description "release counter";
}
leaf info-req-count {
    type uint32;
    mandatory true;
    description "information request
counter";
}
leaf advertise-count {
    type uint32;
    mandatory true;
    description "advertise counter";
}
leaf confirm-count {
    type uint32;
    mandatory true;
    description "confirm counter";
}
leaf reply-count {
    type uint32;
    mandatory true;
    description "reply counter";
}
leaf reconfigure-count {
```



```
        type uint32;
        mandatory true;
        description "reconfigure counter";
    }
    leaf relay-forward-count {
        type uint32;
        mandatory true;
        description "relay forward counter";
    }
    leaf relay-reply-count {
        type uint32;
        mandatory true;
        description "relay reply counter";
    }
}

container relay {
    description "dhcpv6 relay portion";
    container relay-attributes {
        description "A container describes
        some basic attributes of the relay
        agent including some relay agent
        specific options data that need to
        be configured previously. Such options
        include Remote-Id option and
        Subscriber-Id option.";
        leaf name {
            type string;
            mandatory true;
            description "relay agent name";
        }
        leaf enable {
            type boolean;
            mandatory true;
            description "whether the relay is enabled";
        }
        leaf description {
            type string;
            description "description of the relay agent";
        }
        leaf-list dest-addr {
            type inet:ipv6-address;
            description "Each DHCPv6 relay agent may
            be configured with a list of destination
            addresses. This node defines such a list
            of IPv6 addresses that may include
            unicast addresses, multicast addresses
```



```
        or other addresses.";
    }
    list subscribers {
        key subscriber;
        description "subscribers";
        leaf subscriber {
            type uint8;
            mandatory true;
            description "subscriber";
        }
        leaf subscriber-id {
            type string;
            mandatory true;
            description "subscriber id";
        }
    }
    list remote-host {
        key ent-num;
        description "remote host";
        leaf ent-num {
            type uint32;
            mandatory true;
            description "enterprise number";
        }
        leaf remote-id {
            type string;
            mandatory true;
            description "remote id";
        }
    }
    uses vendor-infor;
}

container relay-supplied-options-option {
    description "relay supplied options option";
    list rsoo-set {
        key rsoo-set-id;
        description "rsoo set";
        leaf rsoo-set-id {
            type uint8;
            mandatory true;
            description "rsoo set id";
        }
    }
    container erp-local-domain-name-option {
        description "erp local domain name option";
        leaf enable {
            type boolean;
            mandatory true;
            description "indicate whether
```



```
        this option is included in the
        rsoo set";
    }
    list erp-for-client {
        key cli-id;
        description "erp for client";
        leaf cli-id {
            type uint32;
            mandatory true;
            description "client id";
        }
        leaf duid {
            type duidtype;
            mandatory true;
            description "DHCP Unique
            Identifier";
        }
        leaf erp-name {
            type string;
            mandatory true;
            description "erp name";
        }
    }
}

list relay-if {
    key if-name;
    description "A relay agent may have several
    interfaces, we should provide a way to configure
    and manage parameters on the interface-level. A
    list that describes specific interfaces and
    their corresponding parameters is employed to
    fulfil the configuration. Here we use a string
    called 'if-name; as the key of list.";
    leaf if-name {
        type string;
        mandatory true;
        description "interface name";
    }
    leaf enable {
        type boolean;
        mandatory true;
        description
        "whether this interface is enabled";
    }
    leaf ipv6-address {
        type inet:ipv6-address;
```



```
        description
        "ipv6 address for this interface";
    }
    leaf interface-id {
        type string;
        description "interface id";
    }
    leaf-list rsoo-set {
        type uint8;
        description "configured rsoo set";
    }
    list pd-route {
        key pd-route-id;
        description "pd route";
        leaf pd-route-id {
            type uint8;
            mandatory true;
            description "pd route id";
        }
        leaf requesting-router-id {
            type uint32;
            mandatory true;
            description "requesting router id";
        }
        leaf delegating-router-id {
            type uint32;
            mandatory true;
            description "delegating router id";
        }
        leaf next-router {
            type inet:ipv6-address;
            mandatory true;
            description "next router";
        }
        leaf last-router {
            type inet:ipv6-address;
            mandatory true;
            description "previous router";
        }
    }
    list next-entity {
        key dest-addr;
        description "This node defines
        a list that is used to describe
        the next hop entity of this
        relay distinguished by their
        addresses.";
        leaf dest-addr {
```



```
    type inet:ipv6-address;
    mandatory true;
    description "destination addr";
}
leaf available {
    type boolean;
    mandatory true;
    description "whether the next entity
is available or not";
}
leaf multicast {
    type boolean;
    mandatory true;
    description "whether the address is
multicast or not";
}
leaf server {
    type boolean;
    mandatory true;
    description "whether the next entity
is a server";
}
container packet-stats {
    config "false";
    description "packet statistics";
    leaf cli-packet-rvd-count {
        type uint32;
        mandatory true;
        description "client received packet
counter";
    }
    leaf solicit-rvd-count {
        type uint32;
        mandatory true;
        description
"solicit received counter";
    }
    leaf request-rvd-count {
        type uint32;
        mandatory true;
        description
"request received counter";
    }
    leaf renew-rvd-count {
        type uint32;
        mandatory true;
        description
"renew received counter";
    }
}
```



```
}
leaf rebind-rvd-count {
    type uint32;
    mandatory true;
    description
        "rebind received counter";
}
leaf decline-rvd-count {
    type uint32;
    mandatory true;
    description
        "decline received counter";
}
leaf release-rvd-count {
    type uint32;
    mandatory true;
    description
        "release received counter";
}
leaf info-req-rvd-count {
    type uint32;
    mandatory true;
    description
        "information request counter";
}
leaf relay-for-rvd-count {
    type uint32;
    mandatory true;
    description
        "relay forward received counter";
}
leaf relay-rep-rvd-count {
    type uint32;
    mandatory true;
    description
        "relay reply received counter";
}
leaf packet-to-cli-count {
    type uint32;
    mandatory true;
    description
        "packet to client counter";
}
leaf adver-sent-count {
    type uint32;
    mandatory true;
    description
        "advertisement sent counter";
}
```



```
    }
    leaf confirm-sent-count {
        type uint32;
        mandatory true;
        description
            "confirm sent counter";
    }
    leaf reply-sent-count {
        type uint32;
        mandatory true;
        description
            "reply sent counter";
    }
    leaf reconfig-sent-count {
        type uint32;
        mandatory true;
        description
            "reconfigure sent counter";
    }
    leaf relay-for-sent-count {
        type uint32;
        mandatory true;
        description
            "relay forward sent counter";
    }
    leaf relay-rep-sent-count {
        type uint32;
        mandatory true;
        description
            "relay reply sent counter";
    }
}

container relay-stats {
    config "false";
    description "relay statistics";
    leaf cli-packet-rvd-count {
        type uint32;
        mandatory true;
        description "client packet received counter";
    }
    leaf relay-for-rvd-count {
        type uint32;
        mandatory true;
        description "relay forward received counter";
    }
    leaf relay-rep-rvd-count {
```



```
        type uint32;
        mandatory true;
        description "relay reply received counter";
    }
    leaf packet-to-cli-count {
        type uint32;
        mandatory true;
        description "packet to client counter";
    }
    leaf relay-for-sent-count {
        type uint32;
        mandatory true;
        description "relay forward sent counter";
    }
    leaf relay-rep-sent-count {
        type uint32;
        mandatory true;
        description "relay reply sent counter";
    }
    leaf discarded-packet-count {
        type uint32;
        mandatory true;
        description "discarded packet counter";
    }
}

container client {
    description "dhcpv6 client portion";
    list client-if {
        key if-name;
        description "A client may have several
        interfaces, it is more reasonable to
        configure and manage parameters on
        the interface-level. The list defines
        specific client interfaces and their
        data. Different interfaces are distinguished
        by the key which is a configurable string
        value.";
        leaf if-name {
            type string;
            mandatory true;
            description "interface name";
        }
        leaf cli-id {
            type uint32;
            mandatory true;
            description "client id";
        }
    }
}
```



```
leaf duid {
    type duidtype;
    mandatory true;
    description "DHCP Unique
    Identifier";
}
leaf enable {
    type boolean;
    mandatory true;
    description "whether the interface is enabled";
}
leaf description {
    type string;
    description
    "description of the client interface";
}
leaf pd-function {
    type boolean;
    mandatory true;
    description "Whether the client
    can act as a requesting router
    to request prefixes using prefix
    delegation ([RFC3633]).";
}
leaf rapid-commit {
    type boolean;
    mandatory true;
    description "'1' indicates a client can
    initiate a Solicit-Reply message exchange
    by adding a Rapid Commit option in Solicit
    message. '0' means the client is not allowed
    to add a Rapid Commit option to request
    addresses in a two-message exchange
    pattern.";
}
container mo-tab {
    description "The management tab
    label indicates the operation
    mode of the DHCPv6 client. 'm'=1
    and 'o'=1 indicate the client
    will use DHCPv6 to obtain all
    the configuration data. 'm'=1
    and 'o'=0 are a meaningless
    combination. 'm'=0 and 'o'=1
    indicate the client will use
    stateless DHCPv6 to obtain
    configuration data apart from
    addresses/prefixes data.
```



```
'm'=0 and 'o'=0 represent the
client will not use DHCPv6
but use SLAAC to achieve
configuration.";
leaf m-tab {
    type boolean;
    mandatory true;
    description "m tab";
}
leaf o-tab {
    type boolean;
    mandatory true;
    description "o tab";
}
}
container oro-options {
    description "oro options";
    list oro-option {
        key option-code;
        description "oro option";
        leaf option-code {
            type uint16;
            mandatory true;
            description "option code";
        }
        leaf description {
            type string;
            mandatory true;
            description "description of oro
options";
        }
    }
}
}
container client-configured-options {
    description "client configured options";
    list new-or-standard-cli-option {
        key option-code;
        description "new or standard client option";
        leaf option-code {
            type uint16;
            mandatory true;
            description "option code";
        }
        leaf option-name {
            type string;
            mandatory true;
            description "option name";
        }
    }
}
```



```
    leaf option-description {
      type string;
      mandatory true;
      description "description of client
option";
    }
    leaf option-reference {
      type string;
      description "the reference of option";
    }
    leaf option-value {
      type string;
      mandatory true;
      description "the option value";
    }
  }
  container user-class-option {
    description "user class option";
    leaf enable {
      type boolean;
      mandatory true;
      description "indicate whether this
option will be configured at the
client";
    }
    list user-class {
      key user-class-id;
      description "user class";
      leaf user-class-id {
        type uint8;
        mandatory true;
        description "user class id";
      }
      leaf user-class-info {
        type string;
        mandatory true;
        description "user class info";
      }
    }
  }
}
container vendor-class-option {
  description "vendor class option";
  leaf enable {
    type boolean;
    mandatory true;
    description "indicate whether this
option will be configured at the
client";
```



```
    }
    leaf ent-num {
        type uint32;
        mandatory true;
        description "enterprise number";
    }
    leaf-list data {
        type string;
        description "option data";
    }
}
container client-fqdn-option {
    description "client fqdn option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be configured at the
client";
    }
    leaf fqdn {
        type string;
        mandatory true;
        description "fqdn";
    }
    leaf server-initiate-update {
        type boolean;
        mandatory true;
        description "whether server initiate";
    }
    leaf client-initiate-update {
        type boolean;
        mandatory true;
        description "whether client initiate";
    }
}
container client-architecture-type-option {
    description
"client architecture type option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
option will be configured at the
client";
    }
}
list architecture-types {
    key type-id;
```



```
        description "architecture types";
        leaf type-id {
            type uint16;
            mandatory true;
            description "type id";
        }
        leaf most-preferred {
            type boolean;
            mandatory true;
            description "most preferred flag";
        }
    }
}
container client-network-interface-option {
    description
    "client network interface option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be configured at the
        client";
    }
    leaf type {
        type uint8;
        mandatory true;
        description "type";
    }
    leaf major {
        type uint8;
        mandatory true;
        description "major";
    }
    leaf minor {
        type uint8;
        mandatory true;
        description "minor";
    }
}
container kerberos-principal-name-option {
    description
    "kerberos principal name option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
        option will be configured at the
        client";
    }
}
```



```
    }
    leaf principal-name {
        type string;
        mandatory true;
        description "principal name";
    }
}
container client-link-layer-addr-option {
    description
        "client link layer address option";
    leaf enable {
        type boolean;
        mandatory true;
        description "indicate whether this
            option will be configured at the
            client";
    }
    leaf link-layer-type {
        type uint16;
        mandatory true;
        description "link layer type";
    }
    leaf link-layer-addr {
        type string;
        mandatory true;
        description "link layer address";
    }
}
}
container identity-associations {
    config "false";
    description "IA is a construct through
        which a server and a client can identify,
        group, and manage a set of related IPv6
        addresses. The key of the list is a
        4-byte number IAID defined in [RFC3315].";
    list identity-association {
        key iaid;
        description "IA";
        leaf iaid {
            type uint32;
            mandatory true;
            description "IAID";
        }
        leaf ia-type {
            type string;
            mandatory true;
            description "IA type";
        }
    }
}
```



```
    }
    leaf-list ipv6-addr {
      type inet:ipv6-address;
      description "ipv6 address";
    }
    leaf-list ipv6-prefix {
      type inet:ipv6-prefix;
      description "ipv6 prefix";
    }
    leaf-list prefix-length {
      type uint8;
      description "ipv6 prefix length";
    }
    leaf t1-time {
      type yang:timeticks;
      mandatory true;
      description "t1 time";
    }
    leaf t2-time {
      type yang:timeticks;
      mandatory true;
      description "t2 time";
    }
    leaf preferred-lifetime {
      type yang:timeticks;
      mandatory true;
      description "preferred lifetime";
    }
    leaf valid-lifetime {
      type yang:timeticks;
      mandatory true;
      description "valid lifetime";
    }
  }
}

container if-other-paras {
  config "false";
  description "A client can obtain
  extra configuration data other than
  address and prefix information through
  DHCPv6. This container describes such
  data the client was configured. The
  potential configuration data may
  include DNS server addresses, SIP
  server domain names, etc.";
  leaf-list uni-dhcpv6-serv-addr {
    type inet:ipv6-address;
    description "unicast server address";
```



```
}
container dns-paras {
  description "dns parameters";
  leaf domain-search-list {
    type string;
    mandatory true;
    description "domain search list";
  }
  list dns-servers {
    key dns-serv-id;
    description "dns servers";
    leaf dns-serv-id {
      type uint8;
      mandatory true;
      description "dns server id";
    }
    leaf dns-serv-addr {
      type inet:ipv6-address;
      mandatory true;
      description "dns server address";
    }
  }
}

container sip-paras {
  description "sip parameters";
  list sip-servers {
    key sip-serv-id;
    description "sip server info";
    leaf sip-serv-id {
      type uint8;
      mandatory true;
      description "sip server id";
    }
    leaf sip-serv-addr {
      type inet:ipv6-address;
      mandatory true;
      description "sip server address";
    }
    leaf sip-serv-domain-name {
      type string;
      mandatory true;
      description
        "sip server domain name";
    }
  }
}

container nis-paras {
  description "nis parameters";
```



```
    leaf nis-domain-name {
      type string;
      mandatory true;
      description "nis domain name";
    }
    list nis-server {
      key nis-serv-id;
      description "nis server";
      leaf nis-serv-id {
        type uint8;
        mandatory true;
        description "nis server id";
      }
      leaf nis-serv-addr {
        type inet:ipv6-address;
        mandatory true;
        description "nis server address";
      }
    }
  }
  container nis-plus-paras {
    description "nisp parameters";
    leaf nis-plus-domain-name {
      type string;
      mandatory true;
      description "nisp domain name";
    }
    list nis-plus-server {
      key nis-plus-serv-id;
      description "nisp server";
      leaf nis-plus-serv-id {
        type uint8;
        mandatory true;
        description "nisp server id";
      }
      leaf nis-plus-serv-addr {
        type inet:ipv6-address;
        mandatory true;
        description "nisp server address";
      }
    }
  }
  leaf info-refresh-time {
    type yang:timeticks;
    description "info refresh time";
  }
  container time-zone-paras {
    description "time zone parameters";
```



```
    leaf tz-posix {
      type string;
      mandatory true;
      description "tz posix";
    }
    leaf tz-database {
      type string;
      mandatory true;
      description "tz database";
    }
  }
  leaf cli-fqdn {
    type string;
    description "client fqdn";
  }
  container ntp-paras {
    description "ntp parameters";
    list ntp-server {
      key ntp-serv-id;
      description "ntp server";
      leaf ntp-serv-id {
        type uint8;
        mandatory true;
        description "ntp server id";
      }
      leaf ntp-serv-addr {
        type inet:ipv6-address;
        mandatory true;
        description "ntp server address";
      }
      leaf ntp-serv-mul-addr {
        type inet:ipv6-address;
        mandatory true;
        description "ntp server multicast
          address";
      }
      leaf ntp-serv-fqdn {
        type string;
        mandatory true;
        description "ntp server fqdn";
      }
    }
  }
  container sntp-paras {
    description "sntp parameters";
    list sntp-server {
      key sntp-serv-id;
      description "sntp server";
    }
  }
```



```
    leaf sntp-serv-id {
      type uint8;
      mandatory true;
      description "sntp server id";
    }
    leaf sntp-serv-addr {
      type inet:ipv6-address;
      mandatory true;
      description "sntp server address";
    }
  }
}
container network-boot-paras {
  description "network boot parameters";
  list boot-file {
    key boot-file-id;
    description "boot file";
    leaf boot-file-id {
      type uint8;
      mandatory true;
      description "boot file id";
    }
  }
  leaf-list suitable-arch-type {
    type uint16;
    description "architecture type";
  }
  leaf-list suitable-net-if {
    type uint32;
    description "network interface";
  }
  leaf boot-file-url {
    type string;
    mandatory true;
    description "boot file url";
  }
  list boot-file-paras {
    key para-id;
    description "boot file parameters";
    leaf para-id {
      type uint8;
      mandatory true;
      description "parameter id";
    }
    leaf parameter {
      type string;
      mandatory true;
      description "parameter value";
    }
  }
}
```



```
    }  
  }  
}  
container kerberos-paras {  
  description "kerberos parameters";  
  leaf default-realm-name {  
    type string;  
    mandatory true;  
    description "default realm name";  
  }  
  list kdc-info {  
    key kdc-id;  
    description "kdc info";  
    leaf kdc-id {  
      type uint8;  
      mandatory true;  
      description "kdc id";  
    }  
    leaf priority {  
      type uint16;  
      mandatory true;  
      description "priority";  
    }  
    leaf weight {  
      type uint16;  
      mandatory true;  
      description "weight";  
    }  
    leaf transport-type {  
      type uint8;  
      mandatory true;  
      description "transport type";  
    }  
    leaf port-number {  
      type uint16;  
      mandatory true;  
      description "port number";  
    }  
    leaf kdc-ipv6-addr {  
      type inet:ipv6-address;  
      mandatory true;  
      description "kdc ipv6 address";  
    }  
    leaf realm-name {  
      type string;  
      mandatory true;  
      description "realm name";  
    }  
  }  
}
```



```
    }  
  }  
  container addr-selection-params {  
    description "address selection parameters";  
    leaf automatic-row-add {  
      type boolean;  
      mandatory true;  
      description "row add";  
    }  
    leaf prefer-temporary-addr {  
      type boolean;  
      mandatory true;  
      description "prefer temporary";  
    }  
    list policy-table {  
      key policy-id;  
      description "policy table";  
      leaf policy-id {  
        type uint8;  
        mandatory true;  
        description "policy id ";  
      }  
      leaf label {  
        type uint8;  
        mandatory true;  
        description "label";  
      }  
      leaf precedence {  
        type uint8;  
        mandatory true;  
        description "precedence";  
      }  
      leaf prefix-len {  
        type uint8;  
        mandatory true;  
        description "prefix length";  
      }  
      leaf prefix {  
        type inet:ipv6-prefix;  
        mandatory true;  
        description "prefix";  
      }  
    }  
  }  
  leaf sol-max-rt {  
    type yang:timeticks;  
    mandatory true;  
    description "sol max rt";  
  }
```



```
}
leaf inf-max-rt {
    type yang:timeticks;
    mandatory true;
    description "inf max rt";
}
container pcg-server-paras {
    description "pcg server parameters";
    list pcg-server {
        key pcg-serv-id;
        description "pcg server";
        leaf pcg-serv-id {
            type uint8;
            mandatory true;
            description "pcg server id";
        }
        leaf pcg-serv-addr {
            type inet:ipv6-address;
            mandatory true;
            description "pcg server address";
        }
    }
}
}
container s46-rule-paras {
    description "s46 rule parameters";
    list s46-rule {
        key rule-id;
        description "s46 rule";
        leaf rule-id {
            type uint8;
            mandatory true;
            description "rule id";
        }
        leaf rule-type {
            type enumeration {
                enum "BMR">{
                    description "BMR";
                }
                enum "FMR">{
                    description "FMR";
                }
            }
            mandatory true;
            description "rule type";
        }
        leaf ea-len {
            type uint8;
            mandatory true;
```



```
        description "EA bits length";
    }
    leaf prefix4-len {
        type uint8;
        mandatory true;
        description "ipv4 prefix length";
    }
    leaf ipv4-prefix {
        type inet:ipv4-prefix;
        mandatory true;
        description "ipv4 prefix";
    }
    leaf prefix6-len {
        type uint8;
        mandatory true;
        description "ipv6 prefix length";
    }
    leaf ipv6-prefix {
        type inet:ipv6-prefix;
        mandatory true;
        description "ipv6 prefix";
    }
    }
    uses portset-para;
}

}
container s46-br-paras {
    description "s46 br parameters";
    list br {
        key br-id;
        description "br";
        leaf br-id {
            type uint8;
            mandatory true;
            description "br id";
        }
        leaf br-ipv6-addr {
            type inet:ipv6-address;
            mandatory true;
            description "br ipv6 address";
        }
    }
}

}
container s46-dmr-paras {
    description "s46 dmr parameters";
    list dmr {
        key dmr-id;
        description "dmr";
        leaf dmr-id {
```



```
        type uint8;
        mandatory true;
        description "dmr id";
    }
    leaf dmr-prefix-len {
        type uint8;
        mandatory true;
        description "dmr prefix length";
    }
    leaf dmr-ipv6-prefix {
        type inet:ipv6-prefix;
        mandatory true;
        description "dmr prefix";
    }
}

container s46-v4-v6-binding-paras {
    description "s46 v4 v6 binding parameters";
    leaf ipv4-addr {
        type inet:ipv4-address;
        mandatory true;
        description "ipv4 address";
    }
    leaf bind-prefix6-len {
        type uint8;
        mandatory true;
        description "bind ipv6 prefix";
    }
    uses portset-para;
    leaf erp-local-domain-name {
        type string;
        mandatory true;
        description "erp local domain name";
    }
}

container supported-options {
    description "supported options";
    list supported-option {
        key option-code;
        description "supported option";
        leaf option-code {
            type uint16;
            mandatory true;
            description "option code";
        }
        leaf description {
            type string;
            mandatory true;
        }
    }
}
```



```
        description
        "description of supported
        option";
    }
}
}
container packet-stats {
    config "false";
    description "A container records
    all the packet status information
    of a specific interface.";
    leaf solicit-count {
        type uint32;
        mandatory true;
        description "solicit counter";
    }
    leaf request-count {
        type uint32;
        mandatory true;
        description "request counter";
    }
    leaf renew-count {
        type uint32;
        mandatory true;
        description "renew counter";
    }
    leaf rebind-count {
        type uint32;
        mandatory true;
        description "rebind counter";
    }
    leaf decline-count {
        type uint32;
        mandatory true;
        description "decline counter";
    }
    leaf release-count {
        type uint32;
        mandatory true;
        description "release counter";
    }
    leaf info-req-count {
        type uint32;
        mandatory true;
        description "information request counter";
    }
    leaf advertise-count {
```



```
        type uint32;
        mandatory true;
        description "advertise counter";
    }
    leaf confirm-count {
        type uint32;
        mandatory true;
        description "confirm counter";
    }
    leaf reply-count {
        type uint32;
        mandatory true;
        description "reply counter";
    }
    leaf reconfigure-count {
        type uint32;
        mandatory true;
        description "reconfigure counter";
    }
}

}

}

/*
 * Notifications
 */

notification notifications {
    description "dhcpv6 notification module";
    container dhcpv6-server-event {
        description "dhcpv6 server event";
        container pool-running-out {
            description "raised when the
            address/prefix pool is going to
            run out. A threshold for utilization
            ratio of the pool has been defined in
            the server feature so that it will
            notify the administrator when the
            utilization ratio reaches the threshold,
            and such threshold is a settable
            parameter";
            leaf utilization-ratio {
                type uint16;
                mandatory true;
                description "utilization ratio";
            }
        }
        leaf duid {
            type duidtype;
        }
    }
}
```



```
        mandatory true;
        description "DHCP Unique
        Identifier";
    }
    leaf serv-name {
        type string;
        description "server name";
    }
    leaf pool-name {
        type string;
        mandatory true;
        description "pool name";
    }
}
container invalid-client-detected {
    description "raised when the server
    has found a client which can be
    regarded as a potential attacker. Some
    description could also be included.";
    leaf duid {
        type duidtype;
        mandatory true;
        description "DHCP Unique
        Identifier";
    }
    leaf description {
        type string;
        description "description of the event";
    }
}
container dhcpv6-relay-event {
    description "dhcpv6 relay event";
    container topo-changed {
        description "raised when the topology
        of the relay agent is changed.";
        leaf relay-if-name {
            type string;
            mandatory true;
            description "relay interface name";
        }
        leaf first-hop {
            type boolean;
            mandatory true;
            description "first hop";
        }
        leaf last-entity-addr {
            type inet:ipv6-address;
```



```
        mandatory true;
        description "last entity address";
    }
}
}
container dhcpv6-client-event {
    description "dhcpv6 client event";
    container ia-lease-event {
        description "raised when the
client was allocated a new IA from
the server or it renew/rebind/release
its current IA";
        leaf event-type {
            type enumeration{
                enum "allocation" {
                    description "allocate";
                }
                enum "rebind" {
                    description "rebind";
                }
                enum "renew" {
                    description "renew";
                }
                enum "release" {
                    description "release";
                }
            }
            mandatory true;
            description "event type";
        }
        leaf duid {
            type duidtype;
            mandatory true;
            description "DHCP Unique
Identifier";
        }
        leaf iaid {
            type uint32;
            mandatory true;
            description "IAID";
        }
        leaf serv-name {
            type string;
            description "server name";
        }
        leaf description {
            type string;
            description "description of event";
        }
    }
}
```



```
    }
  }
  container invalid-ia-detected {
    description "raised when the identity
    association of the client can be proved
    to be invalid. Possible condition includes
    duplicated address, illegal address, etc.";
    leaf duid {
      type duidtype;
      mandatory true;
      description "DHCP Unique
      Identifier";
    }
    leaf cli-duid {
      type uint32;
      mandatory true;
      description "duid of client";
    }
    leaf iauid {
      type uint32;
      mandatory true;
      description "IAID";
    }
    leaf serv-name {
      type string;
      description "server name";
    }
    leaf description {
      type string;
      description "description of the event";
    }
  }
  container retransmission-failed {
    description "raised when the retransmission
    mechanism defined in [RFC3315] is failed.";
    leaf duid{
      type duidtype;
      description "DUID";
    }
    leaf description {
      type enumeration {
        enum "MRC failed" {
          description "MRC failed";
        }
        enum "MRD failed" {
          description "MRD failed";
        }
      }
    }
  }
```



```

        mandatory true;
        description "description of failure";
    }
}
container failed-status-turn-up {
    description "raised when the client receives
a message includes an unsuccessful Status Code
option.";
    leaf duid {
        type duiddtype;
        mandatory true;
        description "DHCP Unique
Identifier";
    }
    leaf status-code {
        type enumeration {
            enum "1" {
                description "UnspecFail";
            }
            enum "2" {
                description "NoAddrAvail";
            }
            enum "3" {
                description "NoBinding";
            }
            enum "4" {
                description "NotOnLink";
            }
            enum "5" {
                description "UseMulticast";
            }
        }
        mandatory true;
        description "employed status code";
    }
}
}
}
}
<CODE ENDS>
```

4. Security Considerations (TBD)

TBD

5. IANA Considerations (TBD)

TBD

6. Acknowledgements

The authors would like to thank Qi Sun, Lishan Li, Sladjana Zoric, Tomek Mrugalski, Marcin Siodelski, Bernie Volz and Bing Liu for their valuable comments and contributions to this work.

7. References

7.1. 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>>.
- [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>>.
- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", [RFC 3633](#), DOI 10.17487/RFC3633, December 2003, <<http://www.rfc-editor.org/info/rfc3633>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#), DOI 10.17487/RFC6020, October 2010, <<http://www.rfc-editor.org/info/rfc6020>>.
- [RFC6087] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", [RFC 6087](#), DOI 10.17487/RFC6087, January 2011, <<http://www.rfc-editor.org/info/rfc6087>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", [RFC 6991](#), DOI 10.17487/RFC6991, July 2013, <<http://www.rfc-editor.org/info/rfc6991>>.

7.2. Informative References

- [RFC3319] Schulzrinne, H. and B. Volz, "Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers", [RFC 3319](#), DOI 10.17487/RFC3319, July 2003, <<http://www.rfc-editor.org/info/rfc3319>>.

- [RFC3646] Droms, R., Ed., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", [RFC 3646](#), DOI 10.17487/RFC3646, December 2003, <<http://www.rfc-editor.org/info/rfc3646>>.
- [RFC3898] Kalusivalingam, V., "Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", [RFC 3898](#), DOI 10.17487/RFC3898, October 2004, <<http://www.rfc-editor.org/info/rfc3898>>.
- [RFC4075] Kalusivalingam, V., "Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6", [RFC 4075](#), DOI 10.17487/RFC4075, May 2005, <<http://www.rfc-editor.org/info/rfc4075>>.
- [RFC4242] Venaas, S., Chown, T., and B. Volz, "Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", [RFC 4242](#), DOI 10.17487/RFC4242, November 2005, <<http://www.rfc-editor.org/info/rfc4242>>.
- [RFC4704] Volz, B., "The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option", [RFC 4704](#), DOI 10.17487/RFC4704, October 2006, <<http://www.rfc-editor.org/info/rfc4704>>.
- [RFC4833] Lear, E. and P. Eggert, "Timezone Options for DHCP", [RFC 4833](#), DOI 10.17487/RFC4833, April 2007, <<http://www.rfc-editor.org/info/rfc4833>>.
- [RFC5908] Gayraud, R. and B. Lourdelet, "Network Time Protocol (NTP) Server Option for DHCPv6", [RFC 5908](#), DOI 10.17487/RFC5908, June 2010, <<http://www.rfc-editor.org/info/rfc5908>>.
- [RFC5970] Huth, T., Freimann, J., Zimmer, V., and D. Thaler, "DHCPv6 Options for Network Boot", [RFC 5970](#), DOI 10.17487/RFC5970, September 2010, <<http://www.rfc-editor.org/info/rfc5970>>.
- [RFC6334] Hankins, D. and T. Mrugalski, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Option for Dual-Stack Lite", [RFC 6334](#), DOI 10.17487/RFC6334, August 2011, <<http://www.rfc-editor.org/info/rfc6334>>.
- [RFC6422] Lemon, T. and Q. Wu, "Relay-Supplied DHCP Options", [RFC 6422](#), DOI 10.17487/RFC6422, December 2011, <<http://www.rfc-editor.org/info/rfc6422>>.

- [RFC6440] Zorn, G., Wu, Q., and Y. Wang, "The EAP Re-authentication Protocol (ERP) Local Domain Name DHCPv6 Option", [RFC 6440](#), DOI 10.17487/RFC6440, December 2011, <<http://www.rfc-editor.org/info/rfc6440>>.
- [RFC6784] Sakane, S. and M. Ishiyama, "Kerberos Options for DHCPv6", [RFC 6784](#), DOI 10.17487/RFC6784, November 2012, <<http://www.rfc-editor.org/info/rfc6784>>.
- [RFC6939] Halwasia, G., Bhandari, S., and W. Dec, "Client Link-Layer Address Option in DHCPv6", [RFC 6939](#), DOI 10.17487/RFC6939, May 2013, <<http://www.rfc-editor.org/info/rfc6939>>.
- [RFC7078] Matsumoto, A., Fujisaki, T., and T. Chown, "Distributing Address Selection Policy Using DHCPv6", [RFC 7078](#), DOI 10.17487/RFC7078, January 2014, <<http://www.rfc-editor.org/info/rfc7078>>.
- [RFC7083] Droms, R., "Modification to Default Values of SOL_MAX_RT and INF_MAX_RT", [RFC 7083](#), DOI 10.17487/RFC7083, November 2013, <<http://www.rfc-editor.org/info/rfc7083>>.
- [RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", [BCP 187](#), [RFC 7227](#), DOI 10.17487/RFC7227, May 2014, <<http://www.rfc-editor.org/info/rfc7227>>.
- [RFC7291] Boucadair, M., Penno, R., and D. Wing, "DHCP Options for the Port Control Protocol (PCP)", [RFC 7291](#), DOI 10.17487/RFC7291, July 2014, <<http://www.rfc-editor.org/info/rfc7291>>.
- [RFC7598] Mrugalski, T., Troan, O., Farrer, I., Perreault, S., Dec, W., Bao, C., Yeh, L., and X. Deng, "DHCPv6 Options for Configuration of Software Address and Port-Mapped Clients", [RFC 7598](#), DOI 10.17487/RFC7598, July 2015, <<http://www.rfc-editor.org/info/rfc7598>>.

Authors' Addresses

Yong Cui
Tsinghua University
Beijing 100084
P.R.China

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

Hao Wang
Tsinghua University
Beijing 100084
P.R.China

Phone: +86-10-6278-5822
Email: wangh13@mails.tsinghua.edu.cn

Linhui Sun
Tsinghua University
Beijing 100084
P.R.China

Phone: +86-10-6278-5822
Email: lh.sunlinh@gmail.com

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

Email: Ted.Lemon@nominum.com

Ian Farrer
Deutsche Telekom AG
CTO-ATI, Landgrabenweg 151
Bonn, NRW 53227
Germany

Email: ian.farrer@telekom.de

Sladjana Zoric
Deutsche Telekom AG
CTO-IPT, Landgrabenweg 151
Bonn, NRW 53227
Germany

Email: sladjana.zoric@telekom.de

