SIDROPS Working Group
Internet-Draft
Intended status: Standards Track
Expires: 9 October 2026

Y. Liu
China Mobile
C. Lin
New H3C Technologies
H. Wang
Huawei Technologies
J. Roy
J. Haas
HPE
H. Liu
ZTE Corporation
D. Ma
ZDNS

7 April 2026

# YANG Data Model for RPKI to Router Protocol
draft-ietf-sidrops-rtr-yang-05

## Abstract

This document defines YANG data models for managing Resource Public Key Infrastructure (RPKI) to Router Protocol (RFC6810 and RFC8210).

## 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 https://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 9 October 2026.

## Copyright Notice

Table of Contents



## 1. Introduction

[RFC6810] and [RFC8210] describes a protocol to deliver Resource Public Key Infrastructure (RPKI) prefix origin data and router keys from a trusted cache server to a router, referred to as the RPKI to Router (RTR) protocol.

[I-D.ietf-sidrops-8210bis] describes version 2 of the RTR protocol, which adds a new Autonomous System Provider Authorization (ASPA)) PDU type.

This document defines YANG [RFC7950] data models for managing RTR protocol ([RFC6810], [RFC8210], and [I-D.ietf-sidrops-8210bis]).

### 1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. Model Overview

Two YANG data models are defined in this document.

The ietf-rpki-rtr.yang data model provides the methods for managing RTR protocol. It includes:

* Connectivity parameters, such as RPKI cache server IP address and destination port.

* Session parameters, such as purge time, refresh time, response time.

* Session status and statistics, such as session ID, serial number, number of received and transmitted messages.

The ietf-rpki-table.yang data model provides the methods for managing records of RTR protocol and the corresponding state hash which is a hash value used in the Canonical Cache Representation (CCR) content [I-D.ietf-sidrops-rpki-ccr]. It includes:

* Validated ROA Payload (VRP) records.

* Router key records.

* ASPA records.

* CCR state hash, which is optional and is used to verify the integrity and consistency of RPKI data originating from the RPKI cache.

## 3. RPKI to Router YANG Module

### 3.1. Tree View

The complete tree of the "ietf-rpki-rtr" YANG module is represented as following. See [RFC8340] for an explanation of the symbols used.

```
   module: ietf-rpki-rtr
     augment /rt:routing/rt:control-plane-protocols
               /rt:control-plane-protocol:
       +--rw rpki-rtr
          +--rw sessions
             +--rw session* [server-address]
                +--rw server-address           inet:ip-address
                +--rw server-port?             inet:port-number
                +--rw local-address?           union
                +--rw local-port?              inet:port-number
                +--rw enabled?                 boolean
                +--rw preference?              uint32
                +--rw description?             string
                +--ro session-state?           enumeration
                +--rw enable-authentication?   boolean
                +--rw authentication
                |  +--rw (option)?
                |     +--:(md5)
                |     |  +--rw md5-password?   ianach:crypt-hash
                |     +--:(ssh)
                |     |  +--rw client-identity
                |     |  |  +--rw username?  string
                |     |  |  +--rw public-key!
                |     |  |                   {userauth-publickey}?
                |     |  |  +--rw password!
                |     |  |                   {userauth-password}?
                |     |  |  +--rw hostbased!
                |     |  |                   {userauth-hostbased}?
                |     |  |  +--rw none?      empty {userauth-none}?
                |     |  |  +--rw certificate!
                |     |  |                   {sshcmn:ssh-x509-certs}?
                |     |  +--rw server-authentication
                |     |  |  +--rw ssh-host-keys!
                |     |  |  +--rw ca-certs!  {sshcmn:ssh-x509-certs}?
                |     |  |  +--rw ee-certs!  {sshcmn:ssh-x509-certs}?
                |     |  +--rw transport-params
                |     |  |       {ssh-client-transport-params-config}?
                |     |  +--rw keepalives!   {ssh-client-keepalives}?
                |     |     +--rw max-wait?       uint16
                |     |     +--rw max-attempts?   uint8
                |     +--:(tcp-ao-keychain)
                |        +--rw keychain-name?  key-chain:key-chain-ref
                +--rw vrp-limit
                |  +--rw max-number?              uint64
                |  +--rw threshold-percentage?    uint8
                |  +--rw over-threshold-action?   enumeration
                |  +--rw reconnect-interval?      uint32
                +--rw aspa-limit
```

```
|  +--rw max-number?              uint64
|  +--rw threshold-percentage?    uint8
|  +--rw over-threshold-action?   enumeration
|  +--rw reconnect-interval?      uint32
+--ro statistics
|  +--ro total-vrp-records? yang:zero-based-counter64
|  +--ro ipv4-vrp-records?  yang:zero-based-counter64
|  +--ro ipv6-vrp-records?  yang:zero-based-counter64
|  +--ro router-key-records?
|  |                        yang:zero-based-counter64
|  +--ro aspa-records? yang:zero-based-counter64
+--ro connection-data
|  +--ro flaps?                          uint32
|  +--ro last-session-up-down?           yang:timestamp
|  +--ro last-update-sync-timestamp?     yang:timestamp
|  +--ro last-full-sync-timestamp?       yang:timestamp
|  +--ro last-serial-query-timestamp?    yang:timestamp
|  +--ro last-reset-query-timestamp?     yang:timestamp
|  +--ro last-eod-received?              yang:timestamp
|  +--ro last-config-change-timestamp? yang:timestamp
|  +--ro last-error-timestamp?           yang:timestamp
|  +--ro last-connection-error-timestamp?
|  |                                     yang:timestamp
|  +--ro last-connection-timestamp?      yang:timestamp
|  +--ro error-reason?                   string
+--ro protocol-data
|  +--ro protocol-version?   uint32
|  +--ro refresh-time?       yang:timestamp
|  +--ro response-time?      yang:timestamp
|  +--ro purge-time?         yang:timestamp
|  +--ro hold-time?          yang:timestamp
|  +--ro record-lifetime?    yang:timestamp
|  +--ro retry-interval?     uint32
|  +--ro expire-interval?    uint32
|  +--ro session-id?         uint16
|  +--ro serial-full?        uint32
|  +--ro serial-incremental? uint32
|  +--ro in-total-messages?  yang:zero-based-counter64
|  +--ro out-total-messages? yang:zero-based-counter64
+--ro pdu-counters
|  +--ro serial-notify?    yang:zero-based-counter64
|  +--ro cache-response?   yang:zero-based-counter64
|  +--ro ipv4-prefix?      yang:zero-based-counter64
|  +--ro ipv6-prefix?      yang:zero-based-counter64
|  +--ro end-of-data?      yang:zero-based-counter64
|  +--ro cache-reset?      yang:zero-based-counter64
|  +--ro reset-query?      yang:zero-based-counter64
|  +--ro serial-query?     yang:zero-based-counter64
```

```
               +--ro error-pdu-counters
                  +--ro corrupt-data?      yang:zero-based-counter64
                  +--ro internal-error?    yang:zero-based-counter64
                  +--ro unsupported-protocol-version?
                  |                        yang:zero-based-counter64
                  +--ro unsupported-pdu-type?
                  |                        yang:zero-based-counter64
                  +--ro unexpected-protocol-version?
                  |                        yang:zero-based-counter64
                  +--ro no-data-available? yang:zero-based-counter64
                  +--ro invalid-request?   yang:zero-based-counter64
                  +--ro withdrawal-unknown-record?
                  |                        yang:zero-based-counter64
                  +--ro duplicate-announcement-received?
                                           yang:zero-based-counter64
```

### 3.2. YANG Module

```
<CODE BEGINS> file "ietf-rpki-rtr@2026-03-31.yang"
module ietf-rpki-rtr {
  yang-version "1.1";
  namespace "urn:ietf:params:xml:ns:yang:ietf-rpki-rtr";
  prefix "rpki-rtr";
  import ietf-yang-types {
    prefix "yang";
    reference
      "RFC 9911: Common YANG Data Types";
  }
  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 9911: Common YANG Data Types";
  }
  import ietf-routing {
    prefix rt;
    reference
      "RFC 8349: A YANG Data Model for Routing Management
                 (NMDA Version)";
  }
  import iana-crypt-hash {
    prefix "ianach";
    reference
      "RFC 7317: A YANG Data Model for System Management";
  }
  import ietf-ssh-client {
    prefix "ssh";
    reference
      "RFC 9644: YANG Groupings for SSH Clients and SSH Servers";
```

```
  }
  import ietf-interfaces {
    prefix "if";
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }
  import ietf-key-chain {
    prefix key-chain;
    reference
      "RFC 8177: YANG Data Model for Key Chains";
  }

  organization
    "IETF Secure Inter-Domain Routing Operations (SIDROPS)
     Working Group";

  contact
    "WG Web:   <https://datatracker.ietf.org/wg/sidrops/>
     WG List:  SIDROPS <mailto:sidrops@ietf.org>

     Editor:   Yisong Liu
               <mailto:liuyisong@chinamobile.com>
     Editor:   Changwang Lin
               <mailto:linchangwang.04414@h3c.com>
     Editor:   Haibo Wang
               <mailto:rainsword.wang@huawei.com>
     Editor:   Jishnu Roy
               <mailto:jishnu.roy@hpe.com>
     Editor:   Jeffrey Haas
               <mailto:jeffrey.haas@hpe.com>
     Editor:   Hongwei Liu
               <mailto:liu.hongwei3@zte.com.cn>
     Editor:   Di Ma
               <mailto:madi@zdns.cn>";

  description
    "This module describes a YANG model for the Resource Public
     Key Infrastructure (RPKI) to Router (RTR) protocol management.

     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
     NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
     'MAY', and 'OPTIONAL' in this document are to be interpreted as
     described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
     they appear in all capitals, as shown here.

     Copyright (c) 2026 IETF Trust and the persons identified as
     authors of the code. All rights reserved.
```

```
  revision 2026-03-31 {
    description
      "Initial Version";
    reference
      "RFC XXXX: YANG Data Model for RPKI to Router Protocol";
  }
  typedef ipv4-pfx-len {
    type uint8 {
      range "0 .. 32";
    }
    description
      "IPv4 Prefix Length.";
  }
  typedef ipv6-pfx-len {
    type uint8 {
      range "0 .. 128";
    }
    description
      "IPv6 Prefix Length.";
  }
  typedef subject-key-id {
    type binary {
      length 20;
    }
    description
      "Subject Key Identifier.";
  }
  identity rpki-rtr {
    base rt:routing-protocol;
    description
      "RTR protocol.";
  }
  grouping records-limit {
    description
```

```
        "Limit of records that can be received from the RPKI
         cache server.";
      leaf max-number {
        type uint64;
        description
          "Configures the maximum number of records that can be
           received from the RPKI cache server.";
      }
      leaf threshold-percentage {
        type uint8 {
          range "0..100";
        }
        units "percent";
        description
          "Configures the threshold percentage for record maximum
           number.";
      }
      leaf over-threshold-action {
        type enumeration {
          enum alert-only {
            description
              "Generates alert messages.";
          }
          enum discard {
            description
              "Discards excess records.";
          }
          enum reconnect {
            description
              "Diconncets with the RPKI cache server,
               and tries to reconnect after reconnection
               timer expires.";
          }
          enum idle-forever {
            description
              "Diconncets with the RPKI cache server
               forever.";
          }
        }
        description
          "The action to taken when record number exceeds
           threshold.";
      }
      leaf reconnect-interval {
        type uint32 {
          range "1..30000";
        }
        units "minutes";
```

```
        description
          "Time interval for the reconnection timer.";
      }
    }
    augment "/rt:routing/rt:control-plane-protocols/"
          + "rt:control-plane-protocol" {
      when "derived-from-or-self(rt:type, 'rpki-rtr')" {
        description
          "This augmentation is valid for a routing protocol
           instance of RTR.";
      }
      description
        "RTR protocol augmentation of ietf-routing module
         control-plane-protocol.";
      container rpki-rtr {
        description
          "Configuration parameters for the RTR protocol.";
        container sessions {
          description
            "Parameters of RPKI sessions to cache servers.";
          list session {
            key "server-address";
            description
              "Each entry contains parameters for a RPKI session
               identified by the 'server-address' key.";
            leaf server-address {
              type inet:ip-address;
              mandatory true;
              description
                "The IP address of the RPKI cache server resembling
                 a session";
            }
            leaf server-port {
              type inet:port-number;
              description
                "The remote port for the connection
                 to the RPKI cache server";
            }
            leaf local-address {
              type union {
                type inet:ip-address;
                type if:interface-ref;
              }
              description
                "The local IP (either IPv4 or IPv6) address to use for
                 the connection to the RPKI cache server. This may be
                 expressed as either an IP address or reference to the
                 name of an interface.";
```

```
        }
        leaf local-port {
          type inet:port-number;
          description
            "The local port for the connection
             to the RPKI cache server";
        }
        leaf enabled {
          type boolean;
          default "true";
          description
            "Whether the RPKI cache server is enabled.";
        }
        leaf preference {
          type uint32;
          description
            "The router's preference to connect to that cache.
             The lower the value, the more preferred.";
        }
        leaf description {
          type string;
          description
            "Textual description of the RPKI cache server";
        }
        leaf session-state {
          type enumeration {
            enum idle {
              description
                "The session is down.";
            }
            enum connect {
              description
                "The session is waiting for the underlying
                 transport session to be established.";
            }
            enum establish {
              description
                "The session is up.";
            }
            enum ex-incr {
              description
                "Incremental update of records in progress.";
            }
            enum ex-full {
              description
                "Full update of records in progress.";
            }
          }
```

```
        config false;
        description
          "The session state.";
      }
      leaf enable-authentication {
        type boolean;
        default "false";
        description
          "Whether the session is secured.";
      }
      container authentication {
        when "../enable-authentication = 'true'";
        description
          "Container for describing how a particular session
           is to be secured.";
        choice option {
          description
            "Choice for session secruring methods.";
          case md5 {
            leaf md5-password {
              type ianach:crypt-hash;
              description
                "The password for md5 authentication.";
            }
            description
              "Uses TCP-MD5 to secure the session.";
          }
          case ssh {
            uses ssh:ssh-client-grouping {
              reference
                "RFC 9644: YANG Groupings for SSH Clients and
                           SSH Servers";
            }
            description
              "Uses SSH to secure the session.";
          }
          case tcp-ao-keychain {
            leaf keychain-name {
              type key-chain:key-chain-ref;
              description
                "Name of key chain.";
              reference
                "RFC 8177: YANG Data Model for Key Chains";
            }
            description
              "Uses key-chain to secure the session.";
          }
        }
```

```
        }
        container vrp-limit {
          description
            "Limit of Validated ROA Payload records that can be
             received from the RPKI cache server.";
          uses records-limit;
        }
        container aspa-limit {
          description
            "Limit of Autonomous System Provider Authorization
             (ASPA) records that can be received from the RPKI
             cache server.";
          uses records-limit;
        }
        container statistics {
          config false;
          description
            "Statistics of the RPKI cache server.";
          leaf total-vrp-records {
            type yang:zero-based-counter64;
            description
              "The total number of Validated ROA Payloads
               received from the RPKI cache server.";
          }
          leaf ipv4-vrp-records {
            type yang:zero-based-counter64;
            description
              "The number of Validated ROA Payloads for IPv4
               prefixes received from the RPKI cache server.";
          }
          leaf ipv6-vrp-records {
            type yang:zero-based-counter64;
            description
              "The number of Validated ROA Payloads for IPv6
               prefixes received from the RPKI cache server.";
          }
          leaf router-key-records {
            type yang:zero-based-counter64;
            description
              "The number of router keys received from the RPKI
               cache server.";
          }
          leaf aspa-records {
            type yang:zero-based-counter64;
            description
              "The number of ASPAs received from the RPKI
               cache server.";
          }
```

```
        }
        container connection-data {
          config false;
          description
            "State information relating to the connection
             with the RPKI cache server.";
          leaf flaps {
            type uint32;
            description
              "Count for number of flaps observed on the
               session.";
          }
          leaf last-session-up-down {
            type yang:timestamp;
            description
              "This timestamp indicates the time that the
               RPKI-RTR session last transitioned in or out
               of the UP state. The value is the timestamp in
               microseconds relative to the Unix Epoch (Jan 1,
               1970 00:00:00 UTC). The RPKI-RTR session uptime
               can be computed by clients as the difference
               between this value and the current time
               in UTC (assuming the session is in the UP
               state, per the session-state leaf).";
            reference
              "RFC 6810: The Resource Public Key Infrastructure
                         (RPKI) to Router Protocol";
          }
          leaf last-update-sync-timestamp {
            type yang:timestamp;
            description
              "Time of last serial sync with cache server.";
          }
          leaf last-full-sync-timestamp {
            type yang:timestamp;
            description
              "Time of last reset sync with cache server.";
          }
          leaf last-serial-query-timestamp {
            type yang:timestamp;
            description
              "Time of last serial query sent to cache server.";
          }
          leaf last-reset-query-timestamp {
            type yang:timestamp;
            description
              "Time of last reset query sent to cache server.";
          }
```

```
            leaf last-eod-received {
              type yang:timestamp;
              description
                "Time in microseconds at which last EOD was
                 received.";
            }
            leaf last-config-change-timestamp {
              type yang:timestamp;
              description
                "Time of last host, port, VRF or local interface
                 change.";
            }
            leaf last-error-timestamp {
              type yang:timestamp;
              description
                "Time of sending/receiving protocol error to/from
                 cache server.";
            }
            leaf last-connection-error-timestamp {
              type yang:timestamp;
              description
                "Time of last connection error to cache server.";
            }
            leaf last-connection-timestamp {
              type yang:timestamp;
              description
                "Time of last connection to cache server.";
            }
            leaf error-reason {
              type string;
              description
                "Reason for error in connection.";
            }
          }
          container protocol-data {
            config false;
            description
              "State parameters related to the RTR protocol";
            leaf protocol-version {
              type uint32;
              description
                "The version number of the RTR protocol.";
            }
            leaf refresh-time {
              type yang:timestamp;
              description
                "Configures the time a router waits in between
                 sending periodic serial queries to the RPKI
```

```
          cache server.";
      }
      leaf response-time {
        type yang:timestamp;
        description
          "Configures the time a router waits for a response
           after sending a serial or reset query to the RPKI
           cache server.";
      }
      leaf purge-time {
        type yang:timestamp;
        description
          "Configures the time a router waits to keep data
           from the RPKI cache server after the session
           drops.";
      }
      leaf hold-time {
        type yang:timestamp;
        description
          "Hold-time for this session.";
      }
      leaf record-lifetime {
        type yang:timestamp;
        description
          "Record-lifetime this session.";
      }
      leaf retry-interval {
        type uint32;
        description
          "Number of seconds between poll error and cache
           server poll";
      }
      leaf expire-interval {
        type uint32;
        description
          "Number of seconds to retain data synced from
           cache server";
      }
      leaf session-id {
        type uint16;
        config false;
        description
          "When a cache server is started, it generates a
           Session ID to identify the instance of the cache
           and to bind it to the sequence of Serial Numbers
           that cache instance will generate.";
        reference
          "RFC 6810: The Resource Public Key Infrastructure
```

```
                        (RPKI) to Router Protocol
                    RFC 8210: The Resource Public Key Infrastructure
                        (RPKI) to Router Protocol, Version 1";
                }
                leaf serial-full {
                  type uint32;
                  config false;
                  description
                    "A 32-bit strictly increasing unsigned integer which
                     wraps from 2^32-1 to 0. It denotes the logical
                     version of a cache. It resembles the latest full
                     query.";
                  reference
                    "RFC 6810: The Resource Public Key Infrastructure
                        (RPKI) to Router Protocol
                     RFC 8210: The Resource Public Key Infrastructure
                        (RPKI) to Router Protocol, Version 1";
                }
                leaf serial-incremental {
                  type uint32;
                  config false;
                  description
                    "A 32-bit strictly increasing unsigned integer which
                     wraps from 2^32-1 to 0. It denotes the logical
                     version of a cache. It resembles the latest
                     incremental query.";
                  reference
                    "RFC 6810: The Resource Public Key Infrastructure
                        (RPKI) to Router Protocol
                     RFC 8210: The Resource Public Key Infrastructure
                        (RPKI) to Router Protocol, Version 1";
                }
                leaf in-total-messages {
                  type yang:zero-based-counter64;
                  description
                    "The total number of messages received from the
                     RPKI cache server.";
                }
                leaf out-total-messages {
                  type yang:zero-based-counter64;
                  description
                    "The total number of messages transmitted to the
                     RPKI cache server.";
                }
              }
              container pdu-counters {
                config false;
                description
```

```
            "Counters of PDUs that are received from cache";
          leaf serial-notify {
            type yang:zero-based-counter64;
            description
              "Serial notify PDU count";
          }
          leaf cache-response {
            type yang:zero-based-counter64;
            description
              "Cache response PDU count";
          }
          leaf ipv4-prefix {
            type yang:zero-based-counter64;
            description
              "IPv4 prefix PDU count";
          }
          leaf ipv6-prefix {
            type yang:zero-based-counter64;
            description
              "Ipv6 prefix PDU count";
          }
          leaf end-of-data {
            type yang:zero-based-counter64;
            description
              "End of data PDU count";
          }
          leaf cache-reset {
            type yang:zero-based-counter64;
            description
              "Cache reset PDU count";
          }
          leaf reset-query {
            type yang:zero-based-counter64;
            description
              "Reset query PDU count";
          }
          leaf serial-query {
            type yang:zero-based-counter64;
            description
              "Serial query PDU count";
          }
        }
        container error-pdu-counters {
          config false;
          description
            "Counters of error PDUs that originate from router
             or cache server";
          leaf corrupt-data {
```

```
                type yang:zero-based-counter64;
                description
                  "Corrupt data PDU count";
              }
              leaf internal-error {
                type yang:zero-based-counter64;
                description
                  "Internal error PDU count";
              }
              leaf unsupported-protocol-version {
                type yang:zero-based-counter64;
                description
                  "Unsupported protocol version PDU count";
              }
              leaf unsupported-pdu-type {
                type yang:zero-based-counter64;
                description
                  "Unsupported PDU type count";
              }
              leaf unexpected-protocol-version {
                type yang:zero-based-counter64;
                description
                  "Unexpected protocol version PDU count";
              }
              leaf no-data-available {
                type yang:zero-based-counter64;
                description
                  "No data available PDU count";
              }
              leaf invalid-request {
                type yang:zero-based-counter64;
                description
                  "Invalid request PDU count";
              }
              leaf withdrawal-unknown-record {
                type yang:zero-based-counter64;
                description
                  "Withdrawal of unknown record PDU count";
              }
              leaf duplicate-announcement-received {
                type yang:zero-based-counter64;
                description
                  "Duplicate announcement received PDU count";
              }
            }
          }
        }
      }
```

```
      }
    }
    <CODE ENDS>
```

# 4. RPKI Table YANG Module

## 4.1. Tree View

The complete tree of the "ietf-rpki-table" YANG module is represented as following. See [RFC8340] for an explanation of the symbols used.

```
   module: ietf-rpki-table
     augment /rt:routing:
       +--ro vrp-tables
       |  +--ro vrp-table* [name]
       |     +--ro name                       string
       |     +--ro ccr-roapayloadstate-hash? ccr-hash
       |     +--ro ipv4
       |     |  +--ro vrps
       |     |  |  +--ro vrp* [prefix max-len asn source]
       |     |  |     +--ro prefix       inet:ipv4-prefix
       |     |  |     +--ro max-len      ipv4-pfx-len
       |     |  |     +--ro asn          inet:as-number
       |     |  |     +--ro source       union
       |     |  +--ro total-records?     yang:gauge32
       |     |  +--ro records-added?     yang:counter64
       |     |  +--ro records-deleted?   yang:counter64
       |     +--ro ipv6
       |        +--ro vrps
       |        |  +--ro vrp* [prefix max-len asn source]
       |        |     +--ro prefix       inet:ipv6-prefix
       |        |     +--ro max-len      ipv6-pfx-len
       |        |     +--ro asn          inet:as-number
       |        |     +--ro source       union
       |        +--ro total-records?     yang:gauge32
       |        +--ro records-added?     yang:counter64
       |        +--ro records-deleted?   yang:counter64
       +--ro router-key-tables
       |  +--ro router-key-table* [name]
       |     +--ro name                       string
       |     +--ro ccr-routerkeystate-hash? ccr-hash
       |     +--ro router-keys
       |        +--ro router-key* [ski asn key server-address]
       |           +--ro ski            subject-key-id
       |           +--ro asn            inet:as-number
       |           +--ro key            string
       |           +--ro server-address inet:ip-address
       +--ro aspa-tables
          +--ro aspa-table* [name]
             +--ro name                        string
             +--ro ccr-aspapayloadstate-hash? ccr-hash
             +--ro aspas* [customer-asn]
                +--ro customer-asn        inet:as-number
                +--ro server-address?     inet:ip-address
                +--ro provider-asns* [provider-asn]
                   +--ro provider-asn     inet:as-number
```

## 4.2. YANG Module

```
   <CODE BEGINS> file "ietf-rpki-table@2026-03-31.yang"
   module ietf-rpki-table {
     yang-version "1.1";
     namespace "urn:ietf:params:xml:ns:yang:ietf-rpki-table";
     prefix "rpki-table";
     import ietf-yang-types {
       prefix "yang";
       reference
         "RFC 9911: Common YANG Data Types";
     }
     import ietf-inet-types {
       prefix "inet";
       reference
         "RFC 9911: Common YANG Data Types";
     }
     import ietf-routing {
       prefix rt;
       reference
         "RFC 8349: A YANG Data Model for Routing Management
                    (NMDA Version)";
     }

     organization
       "IETF Secure Inter-Domain Routing Operations (SIDROPS)
        Working Group";

     contact
       "WG Web:   <https://datatracker.ietf.org/wg/sidrops/>
        WG List:  SIDROPS <mailto:sidrops@ietf.org>

        Editor:   Yisong Liu
                  <mailto:liuyisong@chinamobile.com>
        Editor:   Changwang Lin
                  <mailto:linchangwang.04414@h3c.com>
        Editor:   Haibo Wang
                  <mailto:rainsword.wang@huawei.com>
        Editor:   Jishnu Roy
                  <mailto:jishnu.roy@hpe.com>
        Editor:   Jeffrey Haas
                  <mailto:jeffrey.haas@hpe.com>
        Editor:   Hongwei Liu
                  <mailto:liu.hongwei3@zte.com.cn>
        Editor:   Di Ma
                  <mailto:madi@zdns.cn>";

     description
       "This module describes a YANG model for the Resource Public
        Key Infrastructure (RPKI) to Router (RTR) protocol data
```

```
        management.

        The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
        NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
        'MAY', and 'OPTIONAL' in this document are to be interpreted as
        described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
        they appear in all capitals, as shown here.

        Copyright (c) 2026 IETF Trust and the persons identified as
        authors of the code. All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        All revisions of IETF and IANA published modules can be found
        at the YANG Parameters registry group
        (https://www.iana.org/assignments/yang-parameters).

        This version of this YANG module is part of RFC XXXX; see the
        RFC itself for full legal notices.";

      revision 2026-03-31 {
        description
          "Initial Version";
        reference
          "RFC XXXX: YANG Data Model for RPKI to Router Protocol";
      }
      typedef ipv4-pfx-len {
        type uint8 {
          range "0 .. 32";
        }
        description
          "IPv4 Prefix Length.";
      }
      typedef ipv6-pfx-len {
        type uint8 {
          range "0 .. 128";
        }
        description
          "IPv6 Prefix Length.";
      }
      typedef subject-key-id {
        type binary {
          length 20;
```

```
      }
      description
        "Subject Key Identifier.";
    }
    typedef ccr-hash {
      type binary {
        length 32;
      }
      description
        "Canonical Cache Representation (CCR) state hash.

         This represents a 256-bit SHA-256 hash of the canonical
         representation of RPKI data as defined in RFC YYYY.

         The hash provides a compact, verifiable representation
         of the complete state of RPKI validation data, enabling
         efficient synchronization and consistency verification
         between caches.";
      reference
        "RFC YYYY: A Profile for Resource Public Key Infrastructure
                   (RPKI) Canonical Cache Representation (CCR)";
    }
    grouping aspa-overall-records {
      description
        "Autonomous System Provider Authorization (ASPA) records
         received from all RPKI cache servers.";
      list aspas {
        key "customer-asn";
        description
          "An entry of ASPA.";
        leaf customer-asn {
          type inet:as-number;
          description
            "The AS number of a customer.";
        }
        leaf server-address {
          type inet:ip-address;
          description
            "IP address of the RPKI cache server.";
        }
        list provider-asns {
          key "provider-asn";
          description
            "Providers of the customer.";
          leaf provider-asn {
            type inet:as-number;
            description
              "The AS number of a provider.";
```

```
          }
        }
      }
    }

    augment "/rt:routing" {
      description
        "RPKI tables augmentation of ietf-routing module.";
      container vrp-tables {
        config false;
        description
          "List of tables containing Validated ROA Payloads
           received from all RPKI cache servers.";
        list vrp-table {
          key "name";
          description
            "Table of Validated ROA Payloads received from
             all RPKI cache servers.";
          leaf name {
            type string;
            description
              "Name of the Validated ROA Payload table.";
          }
          leaf ccr-roapayloadstate-hash {
            type ccr-hash;
            description
              "CCR state hash for the ROA payload.

               This hash represents the canonical state of this
               specific ROA entry as defined in the CCR specification.
               It is optional and can be used to verify the integrity
               and consistency of ROA data originating from the RPKI
               cache.";
            reference
              "RFC YYYY: A Profile for Resource Public Key
                         Infrastructure (RPKI) Canonical Cache
                         Representation (CCR)";
          }
          container ipv4 {
            config false;
            description
              "Container for IPv4 Validated ROA Payloads table.";
            container vrps {
              config false;
              description
                "Validated ROA Payloads received from the RPKI
                 cache server.";
              list vrp {
```

```
            key "prefix max-len asn source";
            description
              "An entry of Validated ROA Payload.";
            leaf prefix {
              type inet:ipv4-prefix;
              description
                "The IPv4 prefix of the Validated ROA
                 Payload.";
            }
            leaf max-len {
              type ipv4-pfx-len;
              description
                "Denotes the longest prefix allowed. This
                 MUST NOT be less than the prefix length.";
            }
            leaf asn {
              type inet:as-number;
              description
                "The origin AS number of the Validated ROA
                 Payload.";
            }
            leaf source {
              type union {
                type string;
                type inet:ip-address;
              }
              description
                "String representing the source of the records
                 in this record-set.";
            }
          }
        }
        leaf total-records {
          type yang:gauge32;
          description
            "Number of prefix policy records.";
        }
        leaf records-added {
          type yang:counter64;
          description
            "Number of prefix policy records cumulatively added.";
        }
        leaf records-deleted {
          type yang:counter64;
          description
            "Number of prefix policy records cumulatively
             deleted.";
        }
```

```
          }
          container ipv6 {
            config false;
            description
              "Container for IPv6 Validated ROA Payloads table.";
            container vrps {
              config false;
              description
                "Validated ROA Payloads received from the RPKI cache
                 server.";
              list vrp {
                key "prefix max-len asn source";
                description
                  "An entry of Validated ROA Payload.";
                leaf prefix {
                  type inet:ipv6-prefix;
                  description
                    "The IPv6 prefix of the Validated ROA Payload.";
                }
                leaf max-len {
                  type ipv6-pfx-len;
                  description
                    "Denotes the longest prefix allowed. This
                     MUST NOT be less than the prefix length.";
                }
                leaf asn {
                  type inet:as-number;
                  description
                    "The origin AS number of the Validated ROA
                     Payload.";
                }
                leaf source {
                  type union {
                    type string;
                    type inet:ip-address;
                  }
                  description
                    "Representing the source of the records in this
                     record-set. Either a server IP or a source file
                     of static records.";
                }
              }
            }
            leaf total-records {
              type yang:gauge32;
              description
                "Number of prefix policy records.";
            }
```

```
            leaf records-added {
              type yang:counter64;
              description
                "Number of prefix policy records cumulatively added.";
            }
            leaf records-deleted {
              type yang:counter64;
              description
                "Number of prefix policy records cumulatively
                 deleted.";
            }
          }
        }
      }
      container router-key-tables {
        config false;
        description
          "List of router key table received from all RPKI cache
           servers.";
        list router-key-table {
          key "name";
          description
            "Table of router keys received from all RPKI cache
             servers.";
          leaf name {
            type string;
            description
              "Name of the router key table.";
          }
          leaf ccr-routerkeystate-hash {
            type ccr-hash;
            description
              "CCR state hash for the Router Key payload.

               This hash represents the canonical state of this
               specific Router Key entry as defined in the CCR
               specification. It is optional and can effectively
               verify the consistency of BGPsec router key data
               derived from the RPKI cache.";
            reference
              "RFC YYYY: A Profile for Resource Public Key
                         Infrastructure (RPKI) Canonical Cache
                         Representation (CCR)";
          }
          container router-keys {
            config false;
            description
              "Router keys received from the RPKI cache server.";
```

```
            list router-key {
              key "ski asn key server-address";
              description
                "An entry of router key.";
              leaf ski {
                type subject-key-id;
                description
                  "A router key's Subject Key Identifier (SKI).";
                reference
                  "RFC 6487: A Profile for X.509 PKIX Resource
                              Certificates";
              }
              leaf asn {
                type inet:as-number;
                description
                  "The AS number of the router which the key
                   belongs to.";
              }
              leaf key {
                type string;
                description
                  "A router key's subjectPublicKeyInfo value.";
                reference
                  "RFC 8608: BGPsec Algorithms, Key Formats, and
                              Signature Formats";
              }
              leaf server-address {
                type inet:ip-address;
                description
                  "IP address of the RPKI cache server.";
              }
            }
          }
        }
      }
      container aspa-tables {
        config false;
        description
          "List of tables of ASPAs received from all RPKI cache
           servers.";
        list aspa-table {
          key "name";
          description
            "Table of ASPAs received from all RPKI cache servers.";
          leaf name {
            type string;
            description
              "Name of the ASPA table.";
```

```
          }
          leaf ccr-aspapayloadstate-hash {
            type ccr-hash;
            description
              "CCR state hash for the ASPA payload.

               This hash represents the canonical state of this
               specific ASPA entry as defined in the CCR
               specification. It is optional and facilitates
               consistency checking for AS path validation data
               originating from the RPKI cache.";
            reference
              "RFC YYYY: A Profile for Resource Public Key
                         Infrastructure (RPKI) Canonical Cache
                         Representation (CCR)";
          }
          uses aspa-overall-records;
        }
      }
    }
  }
  <CODE ENDS>
```

## 5. Implementation Status

Note to the RFC Editor - remove this section before publication, as well as remove the reference to [RFC7942].

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

### 5.1. Juniper Networks (HPE)

* Organization: Juniper Networks (HPE).

* Implementation: The following leaves/parameters in description are implemented.

* Description: YANG model leaves that are supported:

  - Ietf-rpki-rtr.yang

```
Server-address
Session-state
Total-vrp-records
Ipv4-vrp-records
Ipv6-vrp-records
Refresh-time
Hold-time
Serial-incremental
Serial-full
Last-session-up-down
Las-update-sync-timestamp
Last-eod-received
Last-serial-query-timestamp
Last-reset-query-timestamp
Flaps
```

  - Ietf-rpki-table.yang

```
Ipv4 | ipv6:
  Total-records
  Records-added
  Records-deleted.
```

* Maturity Level: Ready-for-deployment

* Coverage:

* Version: Draft-03

* Licensing: N/A

* Implementation experience: Nothing specific.

* Contact: jishnu.roy@hpe.com

* Last updated: March 30, 2026

### 5.2. New H3C Technologies

* Organization: New H3C Technologies.

* Implementation: The following leaves/parameters in description are implemented.

* Description: "ietf-rpki-rtr" and "ietf-rpki-table" YANG modules have been implemented in New H3C Products.

* Maturity Level: Ready-for-deployment

* Coverage: All data nodes of "ietf-rpki-rtr" and "ietf-rpki-table" YANG modules.

* Version: Draft-03

* Licensing: N/A

* Implementation experience: Nothing specific.

* Contact: li_meng_limeng@h3c.com

* Last updated: March 30, 2026

## 6. Security Considerations

This section is modeled after the template described in Section 3.7.1 of [RFC9907].

The "ietf-rpki-rtr" YANG module and "ietf-rpki-table" YANG module define data models that are designed to be accessed via YANG-based management protocols, such as Network Configuration Protocol (NETCONF) [RFC6241] and RESTCONF [RFC8040]. These YANG-based management protocols (1) have to use a secure transport layer (e.g., Secure Shell (SSH) [RFC4252], TLS [RFC8446], and QUIC [RFC9000]) and (2) have to use mutual authentication.

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in these YANG modules that are writable/creatable/deletable (i.e., config true, which is the default). All writable data nodes are likely to be sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) and delete operations to these data nodes without proper

protection or authentication can have a negative effect on network operations. The following subtrees and data nodes have particular sensitivities/vulnerabilities:

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:server-address

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:server-port

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:local-address

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:local-port

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:enabled

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:preference

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:description

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:enable-authentication

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:authentication

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:vrp-limit

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:aspa-limit

Some of the readable data nodes in these YANG modules may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. Specifically, the following subtrees and data nodes have particular sensitivities/ vulnerabilities:

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:session-state

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:statistics

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:connection-data

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:protocol-data

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:pdu-counters

rpki-rtr:rpki-rtr/rpki-rtr:sessions/rpki-rtr:session/rpki-rtr:error-pdu-counters

rt:routing/rpki-table:vrp-tables/rpki-table:vrp-table

rt:routing/rpki-table:router-key-tables/rpki-table:router-key-table

rt:routing/rpki-table:aspa-tables/rpki-table:aspa-table

There are no particularly sensitive RPC or action operations.

## 7. IANA Considerations

### 7.1. RPKI to Router YANG Module Registry

IANA is requested to register the following URI in the "ns" registry within the "IETF XML Registry" group ([RFC3688]):

URI: urn:ietf:params:xml:ns:yang:ietf-rpki-rtr
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace

IANA is requested to register the following YANG modules in the "YANG Module Names" registry ([RFC6020]) within the "YANG Parameters" registry group.

Name: ietf-rpki-rtr
Maintained by IANA? N
Namespace: urn:ietf:params:xml:ns:yang:ietf-rpki-rtr
Prefix: rpki-rtr
Reference: RFC XXXX

### 7.2. RPKI Table YANG Module Registry

IANA is requested to register the following URI in the "ns" registry within the "IETF XML Registry" group ([RFC3688]):

```
URI: urn:ietf:params:xml:ns:yang:ietf-rpki-table
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace
```

IANA is requested to register the following YANG module in the "YANG Module Names" registry ([RFC6020]) within the "YANG Parameters" registry group.

```
Name: ietf-rpki-table
Maintained by IANA? N
Namespace: urn:ietf:params:xml:ns:yang:ietf-rpki-table
Prefix: rpki-table
Reference: RFC XXXX
```

## 8. Acknowledgments

The authors would like to thank Job Snijders, Santosh Kolenchery, Ebben Xavier Aries, Tapasee Ratna Goutam and Haiyang Zhang for their review and discussion of this document.

## 9. References

### 9.1. Normative References

[I-D.ietf-sidrops-8210bis]
Bush, R., Austein, R., and T. Harrison, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 2", Work in Progress, Internet-Draft, draft-ietf-sidrops-8210bis-25, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-8210bis-25>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6810] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol", RFC 6810, DOI 10.17487/RFC6810, January 2013, <https://www.rfc-editor.org/info/rfc6810>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8210] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1", RFC 8210, DOI 10.17487/RFC8210, September 2017, <https://www.rfc-editor.org/info/rfc8210>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

## 9.2. Informative References

[I-D.ietf-sidrops-rpki-ccr] Snijders, J., Bakker, B., Bruijnzeels, T., and T. Buehler, "A Profile for Resource Public Key Infrastructure (RPKI) Canonical Cache Representation (CCR)", Work in Progress, Internet-Draft, draft-ietf-sidrops-rpki-ccr-02, 4 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-rpki-ccr-02>.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/info/rfc4252>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9907] Bierman, A., Boucadair, M., Ed., and Q. Wu, "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 9907, DOI 10.17487/RFC9907, March 2026, <https://www.rfc-editor.org/info/rfc9907>.

## Contributors

Mengxiao Chen
H3C
China
Email: chen.mengxiao@h3c.com

Santosh Kolenchery
HPE
Email: santosh.kolenchery@hpe.com

Haiyang Zhang
H3C
China
Email: zhang.haiyanga@h3c.com

## Authors' Addresses

Yisong Liu
China Mobile
32 Xuanwumen West Street
Beijing
Xicheng District, 100053
China
Email: liuyisong@chinamobile.com

Changwang Lin
New H3C Technologies
8 Yongjia North Road
Beijing
Haidian District, 100094
China
Email: linchangwang.04414@h3c.com

Haibo Wang
Huawei Technologies
China
Email: rainsword.wang@huawei.com

Jishnu Roy
HPE
1133 Innovation Way
Sunnyvale, CA 94089
United States of America
Email: jishnu.roy@hpe.com

Jeffrey Haas
HPE
1133 Innovation Way
Sunnyvale, CA 94089
United States of America
Email: jeffrey.haas@hpe.com

Hongwei Liu
ZTE Corporation
China
Email: liu.hongwei3@zte.com.cn

Di Ma
ZDNS
Floor 21, Block B, Greenland Center
Chaoyang Beijing, 100102
China
Email: madi@zdns.cn