MBONED Working Group
Internet-Draft
Intended status: Standards Track
Expires: 19 October 2026

Y. Liu, Ed.
China Mobile
C. Lin, Ed.
New H3C Technologies
Z. Zhang
ZTE Corporation
X. Geng
Huawei Technologies
V. Kumar Nagaraj
Juniper Networks
17 April 2026

A YANG Data Model for Automatic Multicast Tunneling (AMT)
draft-ietf-mboned-amt-yang-09

Abstract

This document defines a YANG data model for the management of Automatic Multicast Tunneling (AMT) protocol operations.

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 19 October 2026.

Copyright Notice

Table of Contents

1. Introduction

[RFC7450] introduces the protocol definition of the Automatic Multicast Tunneling (AMT) for delivering multicast traffic from sources in a multicast-enabled network to receivers that lack multicast connectivity to the source network. AMT uses UDP encapsulation and unicast replication to provide this functionality.

[RFC8777] updates [RFC7450] by modifying the Relay Discovery process. It defines DNS Reverse IP AMT Discovery (DRIAD) mechanism for AMT Gateways to discover AMT Relays that are capable of forwarding multicast traffic from a known source IP address.

This document defines a YANG data model for managing AMT protocol.

RFC Ed.: Please replace all occurrences of 'XXXX' with the actual RFC number (and remove this note). Also, please update the revision date to match the publication date.

2. Terminology & Notation Conventions

2.1. Conventions

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.

2.2. Terminology

The terminology for describing YANG data models is found in [RFC6020] (which defines YANG version 1) and [RFC7950] (which defines YANG version 1.1), including:

* augment
* data model
* data node
* identity
* module

The following AMT terms are used in this document:

* Gateway: A functional entity that acts as an endpoint for AMT tunnels on the receiver's domain. Its primary role is to discover AMT Relays, establish AMT tunnels to them, receive multicast traffic over these tunnels, and then forward that traffic directly within its local domain.
* Relay: A functional entity located in the source's domain or a multicast-enabled part of the Internet. Its primary role is to listen for requests from AMT Gateways, replicate multicast traffic from standard multicast routing domains, and encapsulate/ forward this traffic through established AMT tunnels to requesting Gateways.

* Tunnel: A unidirectional, UDP-based encapsulation tunnel established between an AMT Gateway (the tunnel head) and an AMT Relay (the tunnel tail). It is used to transport multicast packets from the Relay to the Gateway over networks that do not support built-in IP multicast.

* Pseudo-Interface: A logical interface within the AMT Gateway or Relay that represents the endpoint of an AMT tunnel. Multicast routing protocols interact with this interface as if it were a physical interface receiving standard multicast traffic.

* Gateway Service: The functional component in the AMT protocol architecture. It interacts downstream with local multicast receivers and upstream with AMT Relays. It implements the complete service functionality of the AMT protocol, including processing multicast subscription requests, establishing tunnels, decapsulating data, and forwarding multicast traffic.

* Relay Service: The central component in the AMT protocol architecture. It establishes and manages tunnels between multicast sources and AMT gateways, enabling the conversion and forwarding of multicast traffic from multicast networks to unicast networks.

* Secret Key Rotation Interval: The maximum recommended validity period or rotation interval for the private secret (or key) used by an AMT Relay to compute Response Message Authentication Code (MAC) values, according to Section 5.3.6 of [RFC7450].

2.3. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

3. Data Model Overview

The AMT protocol mainly includes two components illustrated in Figure 1. The two components are Relay and Gateway entities, each with their internal modules (Discovery, Tunnel Management, and Multicast Forwarding).

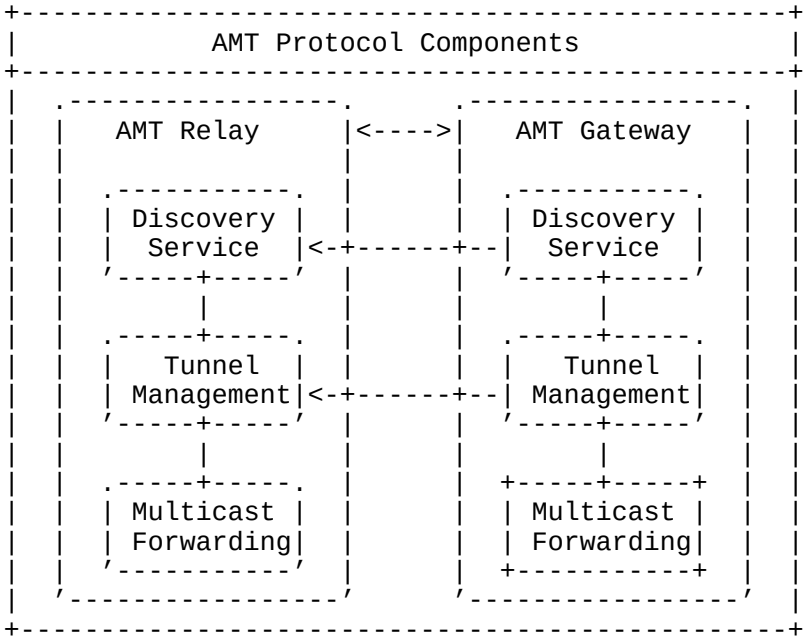

Figure 1: AMT Protocol Components

The AMT data model provides methods for managing AMT protocol, covering all its core functional components as illustrated in Figure 1. It includes:

* Parameters of AMT Relay service, such as Relay Discovery Address (Section 4.1.5 of [RFC7450]), Relay Address (Section 4.1.5 of [RFC7450]), the maximum number of tunnels, and secret key rotation interval.

* Parameters of AMT Gateway service, such as Relay Discovery Address (Section 4.1.5 of [RFC7450]), Relay Address (Section 4.1.5 of [RFC7450]), Discovery Timeout (Section 5.2.2.4 of [RFC7450]), Request Timeout (Section 5.2.2.4 of [RFC7450]), and Maximum Retransmission Count (Section 5.2.2.4 of [RFC7450]).

* AMT tunnel information, such as endpoint IP address and UDP port number, local IP address and UDP port number.

* DNS Resource Record (RR) used by an AMT Relay service.

4. AMT YANG Module

4.1. Prefixes

Table 1 summarizes the prefixes used in this document.

Prefix	YANG module	Reference
inet	ietf-inet-types	[RFC9911]
rt-types	ietf-routing-types	[RFC8294]
rt	ietf-routing	[RFC8349]
yang	ietf-yang-types	[RFC9911]
if	ietf-interfaces	[RFC8343]

Table 1: Prefixes and Corresponding YANG Modules

4.2. Tree View

The full tree diagram of the "ietf-amt" YANG module is provided in Appendix A. The following subsections list the subtree structures.

4.2.1. Overall Structure

The overall tree structure of the AMT YANG module is shown in Figure 2.

The AMT YANG module augments the core routing YANG module "ietf-routing" specified in [RFC8349]. Specifically, the AMT YANG module augments "/rt:routing/rt:control-plane-protocols".

```
module: ietf-amt
  augment /rt:routing/rt:control-plane-protocols:
    +--rw amt!
       +--rw relay {amt-relay}?
       |  ...
       +--rw gateway {amt-gateway}?
          ...
```

Figure 2: Overall AMT Tree Structure

The ’amt’ container encapsulates all AMT functionality and serves as the primary entry point for its configuration and state. The ’amt’ container consists of two functional components: the ’relay’ and the ’gateway’. Support of an AMT Relay or an AMT Gateway is indicated by dedicated YANG features.

The ’relay’ container manages the AMT Relay function on the multicast source side. It provides the configuration and operational state for AMT Relay devices that receive multicast traffic, tunnel it to AMT Gateways over unicast, and act as tunnel termination points. This container is conditionally present only if the device implements the ’amt-relay’ feature, typically on service provider edge routers or data center gateways.

The ’gateway’ container manages the AMT Gateway function on the multicast receiver side. It provides the configuration and operational state for AMT Gateway devices that discover AMT Relays, establish tunnels to receive multicast traffic, and forward it to local receivers. This container is conditionally present only if the device implements the ’amt-gateway’ feature, typically on enterprise edge routers or Customer Premises Equipment (CPE).

4.2.2. Relay

The overall structure of ’relay’ is shown in Figure 3.

```
module: ietf-amt
  augment /rt:routing/rt:control-plane-protocols:
    +--rw amt!
       +--rw relay {amt-relay}?
       |  +--rw addresses
       |  |  +--rw address* [family]
       |  |     +--rw family              identityref
       |  |     +--rw anycast-prefix?     inet:ip-prefix
       |  |     +--rw local-address?      inet:ip-address
       |  +--rw tunnel-limit?             uint32
       |  +--rw secret-key-rotation-interval?      uint32
       |  +--rw relay-dns-resource-records
       |  |  +--rw record* [source-address]
       |  |     +--rw source-address         inet:ip-address
       |  |     +--rw precedence?            uint32
       |  |     +--rw d-bit?                 boolean
       |  |     +--rw relay-type?            enumeration
       |  |     +--rw discovery-address?     inet:ip-address
       |  |     +--rw domain-name?           inet:domain-name
       |  +--ro tunnels
       |  |  ...
       |  +--ro relay-message-statistics
       |     ...
       +--rw gateway {amt-gateway}?
          ...
```

Figure 3: AMT Relay Subtree Structure

The 'relay' data nodes are described as follows:

'addresses': Indicates the core address configurations for AMT Relay.

This data node includes 'family', 'anycast-prefix', and 'local-address'. The 'family' indicates the address family (IPv4 or IPv6). The 'anycast-prefix' indicates the address prefix used by the Gateway to discover the Relay. The 'local-address' indicates the local interface address the Relay actually listens on and sends AMT messages on, or the actual communication address after the tunnel is established.

'tunnel-limit': Indicates the maximum number of endpoint Gateways that a Relay can serve simultaneously.

'secret-key-rotation-interval': Indicates the maximum recommended

validity period or rotation interval for the private secret (or key) used by an AMT Relay to compute Response MAC values. In addition, the private secret (or key) is known only to the AMT Relay, and the provisioning of the private secret (or key) is out of scope.

'relay-dns-resource-records': Indicates the DNS RR configuration for AMT Relay Discovery. Each DNS RR configuration ('record') includes the specific multicast source IP address to which this DNS RR applies ('source-address'), the priority value of this DNS RR ('precedence'), the discovery optional flag ('d-bit'), the type of AMT Relay Address ('relay-type'), the directly specified IP address of AMT Relay Discovery Address ('discovery-address'), and the wire-encoded domain name of AMT Relay ('domain-name').

'tunnels' (Figure 4): Indicates tunnel information from various AMT Gateways connected to this AMT Relay.

Each tunnel entry ('tunnel') includes the IP address and port number of the tunnel opposite end ('gateway') ('gateway-address' and 'gateway-port'), the local IP address and UDP port number used by the local ('relay') end for this tunnel ('local-address' and 'local-port'), the tunnel status ('state'), the multicast flow information ('multicast-flows') carried by the tunnel, the number of different multicast groups currently carried by this tunnel ('multicast-group-num'), the message counter carried by the tunnel ('request-message-count', 'membership-query-message-count', and 'membership-update-message-count'), and the time on the most recent occasion at which any one or more of the tunnel's counters suffered a discontinuity ('discontinuity-time').

Each multicast flow information ('flow') has multicast source address ('source-address') and multicast group address ('group-address').

Design note: The four data nodes ('gateway-address', 'gateway-port', 'local-address', and 'local-port') do not reuse the standard "udp-client" grouping defined in [I-D.ietf-netconf-udp-client-server] because AMT requires the Gateway to be a specific IP address (inet:ip-address), while the "udp-client" grouping allows the use of domain names (inet:host). Reuse could lead to configuration errors or runtime risks, so a custom structure must be defined to enforce this constraint.

```
module: ietf-amt
  augment /rt:routing/rt:control-plane-protocols:
    +--rw amt!
       +--rw relay {amt-relay}?
       |  ...
       |  +--ro tunnels
       |  |  +--ro tunnel* [gateway-address gateway-port]
       |  |     +--ro gateway-address     inet:ip-address
       |  |     +--ro gateway-port        inet:port-number
       |  |     +--ro local-address?      inet:ip-address
       |  |     +--ro local-port?         inet:port-number
       |  |     +--ro state?              identityref
       |  |     +--ro multicast-flows
       |  |     |  +--ro flow* [source-address
       |  |     |     |         group-address]
       |  |     |     +--ro source-address
       |  |     |     |         ip-multicast-source-address
       |  |     |     +--ro group-address
       |  |     |               rt-types:ip-multicast-group-address
       |  |     +--ro multicast-group-num?       yang:gauge32
       |  |     +--ro request-message-count?
       |  |     |             yang:zero-based-counter64
       |  |     +--ro membership-query-message-count?
       |  |     |             yang:zero-based-counter64
       |  |     +--ro membership-update-message-count?
       |  |     |             yang:zero-based-counter64
       |  |     +--ro discontinuity-time?        yang:date-and-time
       |  +--ro relay-message-statistics
       |     ...
       +--rw gateway {amt-gateway}?
          ...
```

Figure 4: AMT Relay Tunnel Subtree Structure

’relay-message-statistics’: Indicates various messages and error statistics handled by AMT Relay as shown in Figure 5.

```
module: ietf-amt
  augment /rt:routing/rt:control-plane-protocols:
    +--rw amt!
       +--rw relay {amt-relay}?
       |  ...
       |  +--ro tunnels
       |  |  ...
       |  +--ro relay-message-statistics
       |     +--ro received
       |     |  +--ro relay-discovery?      yang:zero-based-counter64
       |     |  +--ro request?              yang:zero-based-counter64
       |     |  +--ro membership-update?    yang:zero-based-counter64
       |     |  +--ro teardown?             yang:zero-based-counter64
       |     +--ro sent
       |     |  +--ro relay-advertisement? yang:zero-based-counter64
       |     |  +--ro membership-query?     yang:zero-based-counter64
       |     +--ro error
       |     |  +--ro incomplete-packet?    yang:zero-based-counter64
       |     |  +--ro invalid-mac?          yang:zero-based-counter64
       |     |  +--ro unexpected-type?      yang:zero-based-counter64
       |     |  +--ro invalid-relay-discovery-address?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro invalid-membership-request-address?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro invalid-membership-update-address?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro incomplete-relay-discovery-messages?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro incomplete-membership-request-messages?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro incomplete-membership-update-messages?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro no-active-gateway?    yang:zero-based-counter64
       |     |  +--ro invalid-inner-header-checksum?
       |     |  |                           yang:zero-based-counter64
       |     |  +--ro gateways-timed-out?   yang:zero-based-counter64
       |     +--ro discontinuity-time?      yang:date-and-time
       +--rw gateway {amt-gateway}?
          ...
```

Figure 5: AMT Relay Statistics Subtree Structure

4.2.3. Gateway

The structure of ’gateway’ is shown in Figure 6.

```
module: ietf-amt
  augment /rt:routing/rt:control-plane-protocols:
    +--rw amt!
       +--rw relay {amt-relay}?
       |  ...
       +--rw gateway {amt-gateway}?
          +--rw pseudo-interfaces
          |  +--rw interface* [name]
          |     +--rw name                       if:interface-ref
          |     +--rw discovery-method?          identityref
          |     +--rw relay-discovery-address?  inet:ip-address
          |     +--rw relay-address?             inet:ip-address
          |     +--rw relay-port?                inet:port-number
          |     +--ro local-address?             inet:ip-address
          |     +--ro local-port?                inet:port-number
          |     +--rw upstream-interface?        if:interface-ref
          |     +--rw discovery-timeout?         uint32
          |     +--rw discovery-retrans-count?  uint32
          |     +--rw request-timeout?           uint32
          |     +--rw request-retrans-count?    uint32
          |     +--rw dest-unreach-retry-count? uint32
          |     +--ro tunnel-state?              identityref
          |     +--ro relay-discovery-message-count?
          |     |                        yang:zero-based-counter64
          |     +--ro relay-advertisement-message-count?
          |     |                        yang:zero-based-counter64
          |     +--ro request-message-count?
          |     |                        yang:zero-based-counter64
          |     +--ro membership-query-message-count?
          |     |                        yang:zero-based-counter64
          |     +--ro membership-update-message-count?
          |     |                        yang:zero-based-counter64
          |     +--ro discontinuity-time?     yang:date-and-time
          +--ro gateway-message-statistics
             +--ro discontinuity-time?        yang:date-and-time
             +--ro received
             |  +--ro relay-advertisement? yang:zero-based-counter64
             |  +--ro membership-query? yang:zero-based-counter64
             +--ro sent
                +--ro relay-discovery? yang:zero-based-counter64
                +--ro request?           yang:zero-based-counter64
                +--ro membership-update? yang:zero-based-counter64
                +--ro teardown?          yang:zero-based-counter64
```

Figure 6: AMT Gateway Subtree Structure

The ’gateway’ data nodes are described as follows:

'pseudo-interfaces': Indicates the configuration and operational state of pseudo interfaces used to establish AMT tunnels between Gateways and Relays.

'gateway-message-statistics': Indicates the message statistics of the AMT Gateway. It has the time on the most recent occasion at which any one or more of the AMT Gateway message counters suffered a discontinuity ('discontinuity-time'), the received message statistics of AMT Gateway ('received'), and the sent message statistics of AMT Gateway ('sent').

'received' container includes the number of AMT Relay advertisement messages received ('relay-advertisement') and the number of AMT membership query messages received ('membership-query').

'sent' container includes the number of AMT Relay Discovery messages sent ('relay- discovery'), the number of AMT membership request messages sent ('request'), the number of AMT membership update messages sent ('membership-update'), and the number of AMT teardown messages sent ('teardown').

4.3. YANG Module

This document imports modules defined in [RFC9911], [RFC8294], [RFC8343], and [RFC8349].

<CODE BEGINS> file "ietf-amt@2026-03-10.yang"

```
module ietf-amt {
  yang-version "1.1";
  namespace "urn:ietf:params:xml:ns:yang:ietf-amt";
  prefix amt;

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 9911: Common YANG Data Types, Section 4";
  }

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 9911: Common YANG Data Types, Section 3";
  }

  import ietf-routing-types {
    prefix rt-types;
    reference
```

```
        "RFC 8294: Common YANG Data Types for the Routing Area";
    }

    import ietf-interfaces {
      prefix if;
      reference
        "RFC 8343: A YANG Data Model for Interface Management";
    }

    import ietf-routing {
      prefix rt;
      reference
        "RFC 8349: A YANG Data Model for Routing Management
                   (NMDA Version)";
    }

    organization
      "IETF Multicast Backbone Deployment (MBONED) Working Group";

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

       Editor:   Yisong Liu
                 <mailto:liuyisong@chinamobile.com>
       Editor:   Changwang Lin
                 <mailto:linchangwang.04414@h3c.com>
       Editor:   Zheng(Sandy) Zhang
                 <mailto:zhang.zheng@zte.com.cn>
       Editor:   Xuesong Geng
                 <mailto:gengxuesong@huawei.com>
       Editor:   Vinod Kumar Nagaraj
                 <mailto:vinkumar@juniper.net>";

    description
      "This module describes a YANG data model for managing the
       Automatic Multicast Tunneling (AMT) protocol.

       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 2026-03-10 {
        description
          "Initial Version";
        reference
          "RFC XXXX: A YANG Data Model for Automatic Multicast
                     Tunneling (AMT)";
      }

      feature amt-gateway {
        description
          "Indicates support of AMT Gateway functionality.";
        reference
          "RFC 7450: Automatic Multicast Tunneling, Section 4.1.2";
      }

      feature amt-relay {
        description
          "Indicates support of AMT Relay functionality.";
        reference
          "RFC 7450: Automatic Multicast Tunneling, Section 4.1.3";
      }

      typedef ip-multicast-source-address {
        type union {
          type rt-types:ipv4-multicast-source-address;
          type rt-types:ipv6-multicast-source-address;
        }
        description
          "This type represents a version-neutral IP multicast source
           address. The format of the textual representation implies
           the IP address family.";
      }

      identity tunnel-state-base {
        description
```

```
      "Base identity for AMT tunnel states.";
  }

  identity up {
    base tunnel-state-base;
    description
      "The AMT tunnel has been successfully established.";
  }

  identity establishing {
    base tunnel-state-base;
    description
      "The AMT tunnel is being established.";
  }

  identity initial {
    base tunnel-state-base;
    description
      "Initial AMT tunnel state.";
  }

  identity discovering {
    base tunnel-state-base;
    description
      "The Relay Discovery message has been sent
       and is waiting for the Advertisement message.";
  }

  identity requesting {
    base tunnel-state-base;
    description
      "The Request message has been sent, waiting for the Query
       message.";
  }

  identity discovery-method-base {
    description
      "Base identity for all methods used to discover an
       AMT Relay Address.

       New discovery methods should be defined by creating
       new identities derived from this base identity.";
  }

  identity by-amt-solicit {
    base discovery-method-base;
    description
      "Find the Relay Address by sending an AMT Discovery message.
```

```
          This method involves sending an AMT Discovery message to
          discover available Relays in the network.";
        reference
          "RFC 7450: Automatic Multicast Tunneling, Section 5.1.1";
      }

      identity by-dns-reverse-ip {
        base discovery-method-base;
        description
          "Find the Relay Address by DNS reverse IP AMT Discovery.

          This method uses DNS reverse IP lookup to discover AMT
          Relays based on the client's IP address.";
        reference
          "RFC 8777: DNS Reverse IP Automatic Multicast Tunneling (AMT)
                     Discovery";
      }

      augment "/rt:routing/rt:control-plane-protocols" {
        description
          "AMT augmentation to the routing instance model.";
        container amt {
          presence "Enables AMT";
          description
            "Management parameters for the AMT protocol.";
          container relay {
            if-feature "amt-relay";
            description
              "Parameters of the AMT Relay service.";
            container addresses {
              description
                "Parameters of AMT Relay Addresses.";
              list address {
                key "family";
                description
                  "Each entry contains parameters for an AMT Relay
                   Address identified by the 'family' key. Under
                   normal operation, these addresses SHOULD belong
                   to the same address family indicated by 'family'.
                   Any mismatch is an indication of abnormal
                   configuration and is therefore allowed to be
                   reported.

                   The 'anycast-prefix' serve as the discovery entry
                   for AMT Relays, while unicast IP addresses
                   'local-address' are the actual communication entities
                   of AMT Relays. The AMT Gateway first locates the AMT
                   Relay via the 'anycast-prefix' and then uses its
```

```
               'local-address' to complete all subsequent AMT
               interactions.";
            leaf family {
              type identityref {
                base rt:address-family;
              }
              description
                "Indicates the address family for the entry.";
            }
            leaf anycast-prefix {
              type inet:ip-prefix;
              description
                "An anycast IP prefix of the AMT Relay Discovery
                 Address which is used when sending discovery
                 messages to a Relay.

                 If 'family' is IPv4, it SHOULD be an IPv4 prefix;
                 If 'family' is IPv6, it SHOULD be an IPv6 prefix.

                 Any mismatch is an indication of abnormal
                 configuration and is therefore allowed to be
                 reported.";
            }
            leaf local-address {
              type inet:ip-address;
              description
                "A unicast IP address of the AMT Relay Address
                 that is statically configured on the Relay device.
                 This address is used by the AMT Gateway to establish
                 tunnels.

                 If 'family' is IPv4, it SHOULD be an IPv4 address;
                 If 'family' is IPv6, it SHOULD be an IPv6 address.

                 Any mismatch is an indication of abnormal
                 configuration and is therefore allowed to be
                 reported.";
            }
          }
        }
        leaf tunnel-limit {
          type uint32;
          description
            "The maximum number of endpoint Gateways that a Relay can
             serve simultaneously.";
        }
        leaf secret-key-rotation-interval {
          type uint32;
```

```
        units "minutes";
        description
          "Specifies the interval period for computing a new
           private secret. This maximum RECOMMENDED interval
           is 120 minutes.";
        reference
            "RFC 7450: Automatic Multicast Tunneling,
                       Section 5.3.6";
      }
      container relay-dns-resource-records {
        description
          "The DNS Resource Records (RRs) of the AMT Relay.";
        list record {
          key "source-address";
          description
            "Specifies an RR entry.";
          leaf source-address {
            type inet:ip-address;
            description
              "The unicast IP address of multicast sender.";
          }
          leaf precedence {
            type uint32;
            description
              "The precedence value of this record, used
               for Relay selection priority.

               Lower values indicate higher priority.
               Relays listed in AMT Relay records with
               a lower value for precedence are to be
               attempted first.";
            reference
              "RFC 8777: DNS Reverse IP Automatic Multicast
                         Tunneling (AMT) Discovery,
                         Section 4.2.1";
          }
          leaf d-bit {
            type boolean;
            default false;
            description
              "If the D-bit is set to true, the Gateway MAY
               send an AMT Request message directly to the
               discovered Relay Address without first
               sending an AMT Discovery message.

               If the D-bit is set to false, the Gateway MUST
               receive an AMT Relay advertisement message
               for an address before sending an AMT
```

```
                Request message to that address.";
              reference
                "RFC 8777: DNS Reverse IP Automatic Multicast
                           Tunneling (AMT) Discovery,
                           Section 4.2.2";
            }
            leaf relay-type {
              type enumeration {
                enum empty {
                  value 0;
                  description
                    "The relay field is empty.";
                }
                enum ipv4-address {
                  value 1;
                  description
                    "The relay field contains a 4-octet IPv4
                     address.";
                }
                enum ipv6-address {
                  value 2;
                  description
                    "The relay field contains a 16-octet IPv6
                     address.";
                }
                enum domain-name {
                  value 3;
                  description
                    "The relay field contains a wire-encoded
                     domain name.";
                }
              }
              description
                "Indicates the type of Relay in the AMT Relay RR.

                 Value 0 indicates that no AMT Relay should be
                 used for multicast traffic from this source.

                 Values 1 and 2 indicate that the IP address is
                 used to describe the AMT Relay.

                 Value 3 indicates that the domain name is
                 used to describe the AMT Relay.";
              reference
                "RFC 8777: DNS Reverse IP Automatic Multicast
                           Tunneling (AMT) Discovery,
                           Section 4.2.3";
            }
```

```
        leaf discovery-address {
          type inet:ip-address;
          must "(../relay-type = 'ipv4-address' or "
             + "../relay-type = 'ipv6-address')" {
            error-message
              "The 'discovery-address' can only be configured
               when the 'relay-type' is ipv4-address or
               ipv6-address.";
          }
          description
            "The IP address of AMT Relay Discovery Address.

             When the 'relay-type' value is 1 or 2, this
             data node is used to indicate the AMT Relay of
             the AMT Relay RR.";
        }
        leaf domain-name {
          type inet:domain-name;
          must "../relay-type = 'domain-name'" {
            error-message
              "The 'domain-name' can only be configured when
               the 'relay-type' is domain-name.";
          }
          description
            "The wire-encoded domain name of the AMT Relay.

             When the 'relay-type' value is 3, this data node
             is used to indicate the AMT Relay of the AMT
             Relay RR.";
        }
      }
    }
    container tunnels {
      config false;
      description
        "AMT tunnel session information, which contains
         session parameters, state, and statistics for
         all AMT tunnels established between Gateways
         and this Relay.";
      list tunnel {
        key "gateway-address gateway-port";
        description
          "Records a tunnel entry.";
        leaf gateway-address {
          type inet:ip-address;
          description
            "The IP address of an AMT Gateway.";
        }
```

```
            leaf gateway-port {
              type inet:port-number;
              description
                "The UDP port number of an AMT Gateway.";
            }
            leaf local-address {
              type inet:ip-address;
              description
                "The local IP address of the AMT Relay.";
            }
            leaf local-port {
              type inet:port-number;
              description
                "The local UDP port number of the AMT Relay.";
            }
            leaf state {
              type identityref {
                base tunnel-state-base;
              }
              description
                "The state of AMT tunnel.";
            }
            container multicast-flows {
              config false;
              description
                "The multicast flow information in the AMT tunnel.

                 Contains operational data for all multicast
                 flows being forwarded through AMT tunnels between
                 this Relay and connected Gateways.";
              list flow {
                key "source-address group-address";
                description
                  "Records the characteristics of a multicast flow.";
                leaf source-address {
                  type ip-multicast-source-address;
                  description
                    "The source IP address of a multicast flow.

                     It MUST belong to the same address family as
                     group-address.";
                }
                leaf group-address {
                  type rt-types:ip-multicast-group-address;
                  description
                    "The group IP address of a multicast flow.

                     It MUST belong to the same address family as
```

```
                    source-address.";
                }
              }
            }
            leaf multicast-group-num {
              type yang:gauge32;
              description
                "Number of multicast groups.";
            }
            leaf request-message-count {
              type yang:zero-based-counter64;
              description
                "Number of AMT Request messages received
                 in the tunnel.";
            }
            leaf membership-query-message-count {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership Query messages sent
                 in the tunnel.";
            }
            leaf membership-update-message-count {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership Update messages received
                 in the tunnel.";
            }
            leaf discontinuity-time {
              type yang:date-and-time;
              description
                "The time on the most recent occasion at which any
                 one or more of this AMT tunnel's counters suffered
                 a discontinuity.

                 If no such discontinuities have occurred since the
                 last re-initialization of the AMT tunnel, then this
                 node contains the time when the AMT tunnel was last
                 initialized or the tunnel was established.";
            }
          }
        }
        container relay-message-statistics {
          config false;
          description
            "Message statistics of an AMT Relay.";
          container received {
            description
              "Received message statistics of AMT Relay.";
```

```
        leaf relay-discovery {
          type yang:zero-based-counter64;
          description
            "Number of AMT Relay Discovery messages
             received.";
        }
        leaf request {
          type yang:zero-based-counter64;
          description
            "Number of AMT membership Request messages
             received.";
        }
        leaf membership-update {
          type yang:zero-based-counter64;
          description
            "Number of AMT membership Update messages
             received.";
        }
        leaf teardown {
          type yang:zero-based-counter64;
          description
            "Number of AMT Teardown messages received.";
        }
      }
      container sent {
        description
          "Sent message statistics of AMT Relay.";
        leaf relay-advertisement {
          type yang:zero-based-counter64;
          description
            "Number of AMT Relay advertisement messages sent.";
        }
        leaf membership-query {
          type yang:zero-based-counter64;
          description
            "Number of AMT membership Query messages sent.";
        }
      }
      container error {
        description
          "Error message statistics of AMT Relay.";
        leaf incomplete-packet {
          type yang:zero-based-counter64;
          description
            "Number of messages received with length errors
             so severe that further classification could not
             occur.";
        }
```

```
            leaf invalid-mac {
              type yang:zero-based-counter64;
              description
                "Number of messages received with an invalid
                 Message Authentication Code (MAC).";
            }
            leaf unexpected-type {
              type yang:zero-based-counter64;
              description
                "Number of messages received with an unknown
                 message type specified.";
            }
            leaf invalid-relay-discovery-address {
              type yang:zero-based-counter64;
              description
                "Number of AMT Relay Discovery messages
                 received with an address other than the
                 configured anycast address.";
            }
            leaf invalid-membership-request-address {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership request messages
                 received with an address other than the
                 configured AMT local address.";
            }
            leaf invalid-membership-update-address {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership update messages
                 received with an address other than the
                 configured AMT local address.";
            }
            leaf incomplete-relay-discovery-messages {
              type yang:zero-based-counter64;
              description
                "Number of AMT Relay Discovery messages
                 received that are not fully formed.";
            }
            leaf incomplete-membership-request-messages {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership request messages
                 received that are not fully formed.";
            }
            leaf incomplete-membership-update-messages {
              type yang:zero-based-counter64;
              description
```

```
                "Number of AMT membership update messages
                 received that are not fully formed.";
            }
            leaf no-active-gateway {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership update messages
                 received for a tunnel that does not exist
                 for the Gateway that sent the message.";
            }
            leaf invalid-inner-header-checksum {
              type yang:zero-based-counter64;
              description
                "Number of AMT membership update messages
                 received with an invalid IP checksum.";
            }
            leaf gateways-timed-out {
              type yang:zero-based-counter64;
              description
                "Number of Gateways that timed out because
                 of inactivity.";
            }
          }
          leaf discontinuity-time {
            type yang:date-and-time;
            description
              "The time on the most recent occasion at which any
               one or more of this AMT tunnel's message counters
               suffered a discontinuity.

               If no such discontinuities have occurred since the
               last re-initialization of the AMT tunnel, then this
               node contains the time when the AMT tunnel was last
               initialized or the tunnel was established.";
          }
        }
      } // relay
      container gateway {
        if-feature "amt-gateway";
        description
          "Parameters of AMT Gateway service.";
        container pseudo-interfaces {
          description
            "Parameters of AMT pseudo-interface.";
          list interface {
            key "name";
            description
              "An entry of AMT pseudo-interface.";
```

```
            leaf name {
              type if:interface-ref;
              description
                "Indicates the name of a pseudo interface.";
            }
            leaf discovery-method {
              type identityref {
                base discovery-method-base;
              }
              description
                "The method used to discover the relay address.";
            }
            leaf relay-discovery-address {
              type inet:ip-address;
              description
                "Specifies the AMT Relay Discovery Address.";
            }
            leaf relay-address {
              type inet:ip-address;
              description
                "Specifies the IP address of the AMT Relay.";
            }
            leaf relay-port {
              type inet:port-number;
              description
                "The UDP port number of the AMT Relay.";
            }
            leaf local-address {
              type inet:ip-address;
              config false;
              description
                "The local IP address of this AMT tunnel.";
            }
            leaf local-port {
              type inet:port-number;
              config false;
              description
                "The local UDP port number of this AMT tunnel.";
            }
            leaf upstream-interface {
              type if:interface-ref;
              description
                "Indicates the upstream interface to reach the AMT
                 Relay.";
            }
            leaf discovery-timeout {
              type uint32;
              units "seconds";
```

```
        description
          "Initial time to wait for a response to
           a Relay Discovery message.";
      }
      leaf discovery-retrans-count {
        type uint32;
        description
          "Maximum number of Relay Discovery retransmissions
           to allow before terminating Relay Discovery
           and reporting an error.";
      }
      leaf request-timeout {
        type uint32;
        units "seconds";
        description
          "Initial time to wait for a response
           to a Request message";
      }
      leaf request-retrans-count {
        type uint32;
        description
          "Maximum number of Request retransmissions
           to allow before abandoning a Relay and restarting
           Relay Discovery or reporting an error.";
      }
      leaf dest-unreach-retry-count {
        type uint32;
        description
          "The maximum number of times a Gateway should
           attempt to send the same Request or Membership
           Update message after receiving an ICMP Destination
           Unreachable message.";
      }
      leaf tunnel-state {
        type identityref {
          base tunnel-state-base;
        }
        config false;
        description
          "The tunnel's state.";
      }
      leaf relay-discovery-message-count {
        type yang:zero-based-counter64;
        config false;
        description
          "Number of AMT Relay Discovery messages sent
           on the interface.";
      }
```

```
            leaf relay-advertisement-message-count {
              type yang:zero-based-counter64;
              config false;
              description
                "Number of AMT Relay advertisement messages received
                 on the interface.";
            }
            leaf request-message-count {
              type yang:zero-based-counter64;
              config false;
              description
                "Number of AMT membership request messages sent
                 on the interface.";
            }
            leaf membership-query-message-count {
              type yang:zero-based-counter64;
              config false;
              description
                "Number of AMT membership query messages received
                 on the interface.";
            }
            leaf membership-update-message-count {
              type yang:zero-based-counter64;
              config false;
              description
                "Number of AMT membership update messages sent
                 on the interface.";
            }
            leaf discontinuity-time {
              type yang:date-and-time;
              config false;
              description
                "The time on the most recent occasion at which any
                 one or more of this interface's counters suffered
                 a discontinuity.

                 If no such discontinuities have occurred since the
                 last re-initialization of the AMT tunnel based on
                 this interface, then this node contains the time
                 when the AMT tunnel was last initialized or the
                 tunnel was established.";
            }
          }
        }
        container gateway-message-statistics {
          config false;
          description
            "Message statistics of the AMT Gateway.";
```

```
            leaf discontinuity-time {
              type yang:date-and-time;
              description
                "The time on the most recent occasion at which the AMT
                 Gateway message counters suffered a discontinuity.

                 If no such discontinuities have occurred since the
                 last re-initialization of the Gateway, then this
                 data node contains the time when the Gateway was last
                 initialized.";
            }
            container received {
              description
                "Received message statistics of the AMT Gateway.";
              leaf relay-advertisement {
                type yang:zero-based-counter64;
                description
                  "Number of AMT Relay advertisement messages
                   received.";
              }
              leaf membership-query {
                type yang:zero-based-counter64;
                description
                  "Number of AMT membership query messages
                   received.";
              }
            }
            container sent {
              description
                "Sent message statistics of the AMT Gateway.";
              leaf relay-discovery {
                type yang:zero-based-counter64;
                description
                  "Number of AMT Relay Discovery messages sent.";
              }
              leaf request {
                type yang:zero-based-counter64;
                description
                  "Number of AMT membership request messages sent.";
              }
              leaf membership-update {
                type yang:zero-based-counter64;
                description
                  "Number of AMT membership update messages sent.";
              }
              leaf teardown {
                type yang:zero-based-counter64;
                description
```

```
                    "Number of AMT teardown messages sent.";
                }
              }
            }
          } // gateway
        } // amt
      } // augment
    }
    <CODE ENDS>
```

5. Operational Considerations

This document specifies a YANG data model for AMT that configures and monitors address parameters for both Relay and Gateway functions. Operators have to monitor for address family mismatches between associated address parameters to ensure correct protocol operation, tunnel establishment, and forwarding behavior.

The following address pairs and combinations are critical and have to be validated for address family consistency:

* On the AMT Relay:

  Within the ’relay/addresses/address’ list entry indexed by a given address family (’family’):

  - The ’anycast-prefix’ (discovery anycast prefix)

  - The ’local-address’ (unicast IP address)

  These IP addresses have to belong to the same address family indicated by the ’family’ leaf (either both IPv4 or both IPv6). A mismatch (e.g., IPv4 ’anycast-prefix’ paired with IPv6 ’local-address’ under the same IPv4 ’family’ entry) indicates a configuration anomaly that can prevent Relay Discovery, Advertisement responses, and tunnel setup.

* On the AMT Gateway:

  Within each ’gateway/pseudo-interfaces/interface’ entry:

  - The ’relay-discovery-address’

  - The ’relay-address’

  - The ’local-address’ (operational state)

These IP addresses must all belong to the same address family. A mismatch can lead to failure in Relay Discovery, tunnel establishment, or traffic decapsulation.

It is RECOMMENDED that operators implement automated configuration validation tools to detect such address family mismatches. When combined with the required monitoring, this provides a robust defense against misconfiguration.

Upon detecting an address family mismatch, the device MUST log an appropriate error or alarm and prevent the inconsistent configuration from being applied. Corrective actions include reconfiguring the affected addresses to match the intended address family and verifying routing reachability for the configured addresses.

This section focuses on fault management for address family mismatches, the core operational risk addressed here. While broader network management includes performance, security, and other aspects, this document does not define new requirements in those areas. The above recommendations for validation and logging still support overall network reliability and security.

6. Security Considerations

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

The "ietf-amt" YANG module defines a data model that is 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 [I-D.ietf-tls-rfc8446bis], 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 this YANG module 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:

Under /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/: Unauthorized access to any data nodes in these subtrees can adversely affect the AMT subsystem of both the local device and the network. This may lead to network malfunctions, delivery of packets to inappropriate destinations, and other problems.

amt/relay/addresses/address:
This subtree specifies the IPv4 or IPv6 address information for an AMT Relay. Modifying the configuration may cause the AMT tunnel to be torn down or established.

amt/relay/secret-key-rotation-interval:
This data node defines the maximum validity period or rotation interval for the private secret key. Modifying this value can weaken security or disrupt operations, as AMT protocol security fundamentally depends on this key. Using a secret key beyond its validity period extends its exposure to potential attacks, increasing the likelihood of compromise and weakening the intended security protections. Therefore, write access to this node MUST be restricted to authorized administrators, and all changes SHOULD be logged. Note that while key provisioning is out of scope of this document, it MUST also be performed securely.

amt/relay/relay-dns-resource-records/record:
This subtree specifies the DNS RR configuration used to discover AMT Relays. Modifying this configuration may cause the AMT Gateway to discover new AMT Relay devices, or fail to discover AMT Relay devices.

amt/gateway/pseudo-interfaces/interface:
This subtree specifies the parameters of AMT pseudo-interface for an AMT Gateway. Modifying this configuration may cause the AMT Gateway to establish or tear down tunnels with multiple AMT Relays.

Some of the readable data nodes in this YANG module 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:

Under /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol/: amt/relay and amt/gateway. Unauthorized access to any data nodes in these subtrees can disclose operational state information about the AMT Relay or AMT Gateway on this device.

There are no particularly sensitive RPC or action operations.

7. IANA Considerations

7.1. IETF XML 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-amt
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

7.2. YANG Module Names Registry

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

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

8. Acknowledgments

Thanks to Mohamed Boucadair for review and comments.

Thanks to David Blacka for the DNSDIR, Mike Ounsworth for the SECDIR, Michael P for the OPSDIR, Robert Wills for the YANGDOCTORS, and Behcet Sarikaya for the GENART review.

Thanks to Deb Cooley, Eric Vyncke, Gorry Fairhurst, Roman Danyliw, Andy Newton, Christopher Inacio, Mahesh Jethanandani, Mike Bishop, Charles Eckel, Gunter Van de Velde, Jim Guichard, Ketan Talaulikar, and Tommy Jensen for the IESG review.

9. References

9.1. 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, <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>.

[RFC7450] Bumgardner, G., "Automatic Multicast Tunneling", RFC 7450, DOI 10.17487/RFC7450, February 2015, <https://www.rfc-editor.org/info/rfc7450>.

[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>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[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>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[RFC8777] Holland, J., "DNS Reverse IP Automatic Multicast Tunneling (AMT) Discovery", RFC 8777, DOI 10.17487/RFC8777, April 2020, <https://www.rfc-editor.org/info/rfc8777>.

[RFC9911] Schowalder, J., Ed., "Common YANG Data Types", RFC 9911, DOI 10.17487/RFC9911, December 2025, <https://www.rfc-editor.org/info/rfc9911>.

9.2. Informative References

[I-D.ietf-netconf-udp-client-server]
Feng, A. H., Francois, P., and K. Watsen, "YANG Groupings for UDP Clients and UDP Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-udp-client-server-10, 16 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-udp-client-server-10>.

[I-D.ietf-tls-rfc8446bis]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-rfc8446bis-14, 13 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-rfc8446bis-14>.

[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>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

[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>.

[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>.

[W3C.REC-xml-20081126]
Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", World Wide Web Consortium Recommendation REC-xml-20081126, November 2008, <https://www.w3.org/TR/2008/REC-xml-20081126>.

Appendix A. Full Tree

```
module: ietf-amt
  augment /rt:routing/rt:control-plane-protocols:
    +--rw amt!
       +--rw relay {amt-relay}?
       |  +--rw addresses
       |  |  +--rw address* [family]
       |  |     +--rw family              identityref
       |  |     +--rw anycast-prefix?     inet:ip-prefix
       |  |     +--rw local-address?      inet:ip-address
       |  +--rw tunnel-limit?             uint32
       |  +--rw secret-key-rotation-interval?      uint32
       |  +--rw relay-dns-resource-records
       |  |  +--rw record* [source-address]
       |  |     +--rw source-address         inet:ip-address
       |  |     +--rw precedence?            uint32
       |  |     +--rw d-bit?                 boolean
       |  |     +--rw relay-type?            enumeration
       |  |     +--rw discovery-address?     inet:ip-address
       |  |     +--rw domain-name?           inet:domain-name
       |  +--ro tunnels
       |  |  +--ro tunnel* [gateway-address gateway-port]
       |  |     +--ro gateway-address     inet:ip-address
       |  |     +--ro gateway-port        inet:port-number
       |  |     +--ro local-address?      inet:ip-address
       |  |     +--ro local-port?         inet:port-number
       |  |     +--ro state?              identityref
       |  |     +--ro multicast-flows
       |  |     |  +--ro flow* [source-address
       |  |     |     |         group-address]
       |  |     |     +--ro source-address
       |  |     |     |         ip-multicast-source-address
       |  |     |     +--ro group-address
       |  |     |               rt-types:ip-multicast-group-address
       |  |     +--ro multicast-group-num?        yang:gauge32
       |  |     +--ro request-message-count?
       |  |     |             yang:zero-based-counter64
       |  |     +--ro membership-query-message-count?
       |  |     |             yang:zero-based-counter64
       |  |     +--ro membership-update-message-count?
```

```
|  |     |              yang:zero-based-counter64
|  |     +--ro discontinuity-time?          yang:date-and-time
|  +--ro relay-message-statistics
|     +--ro received
|     |  +--ro relay-discovery?      yang:zero-based-counter64
|     |  +--ro request?              yang:zero-based-counter64
|     |  +--ro membership-update?    yang:zero-based-counter64
|     |  +--ro teardown?             yang:zero-based-counter64
|     +--ro sent
|     |  +--ro relay-advertisement? yang:zero-based-counter64
|     |  +--ro membership-query?     yang:zero-based-counter64
|     +--ro error
|     |  +--ro incomplete-packet?    yang:zero-based-counter64
|     |  +--ro invalid-mac?          yang:zero-based-counter64
|     |  +--ro unexpected-type?      yang:zero-based-counter64
|     |  +--ro invalid-relay-discovery-address?
|     |  |                           yang:zero-based-counter64
|     |  +--ro invalid-membership-request-address?
|     |  |                           yang:zero-based-counter64
|     |  +--ro invalid-membership-update-address?
|     |  |                           yang:zero-based-counter64
|     |  +--ro incomplete-relay-discovery-messages?
|     |  |                           yang:zero-based-counter64
|     |  +--ro incomplete-membership-request-messages?
|     |  |                           yang:zero-based-counter64
|     |  +--ro incomplete-membership-update-messages?
|     |  |                           yang:zero-based-counter64
|     |  +--ro no-active-gateway?    yang:zero-based-counter64
|     |  +--ro invalid-inner-header-checksum?
|     |  |                           yang:zero-based-counter64
|     |  +--ro gateways-timed-out?   yang:zero-based-counter64
|     +--ro discontinuity-time?      yang:date-and-time
+--rw gateway {amt-gateway}?
   +--rw pseudo-interfaces
   |  +--rw interface* [name]
   |     +--rw name                       if:interface-ref
   |     +--rw discovery-method?          identityref
   |     +--rw relay-discovery-address?   inet:ip-address
   |     +--rw relay-address?             inet:ip-address
   |     +--rw relay-port?                inet:port-number
   |     +--ro local-address?             inet:ip-address
   |     +--ro local-port?                inet:port-number
   |     +--rw upstream-interface?        if:interface-ref
   |     +--rw discovery-timeout?         uint32
   |     +--rw discovery-retrans-count?   uint32
   |     +--rw request-timeout?           uint32
   |     +--rw request-retrans-count?     uint32
   |     +--rw dest-unreach-retry-count? uint32
```

```
|     +--ro tunnel-state?             identityref
|     +--ro relay-discovery-message-count?
|     |                       yang:zero-based-counter64
|     +--ro relay-advertisement-message-count?
|     |                       yang:zero-based-counter64
|     +--ro request-message-count?
|     |                       yang:zero-based-counter64
|     +--ro membership-query-message-count?
|     |                       yang:zero-based-counter64
|     +--ro membership-update-message-count?
|     |                       yang:zero-based-counter64
|     +--ro discontinuity-time?     yang:date-and-time
+--ro gateway-message-statistics
   +--ro discontinuity-time?        yang:date-and-time
   +--ro received
   |  +--ro relay-advertisement? yang:zero-based-counter64
   |  +--ro membership-query? yang:zero-based-counter64
   +--ro sent
      +--ro relay-discovery? yang:zero-based-counter64
      +--ro request?           yang:zero-based-counter64
      +--ro membership-update? yang:zero-based-counter64
      +--ro teardown?          yang:zero-based-counter64
```

Appendix B. Data Model Example

This section presents a simple and illustrative example of how to configure AMT.

The example is represented in both XML [W3C.REC-xml-20081126] and JSON [RFC7951] formats.

Figure 7 shows a sample configuration for an AMT Relay service in XML format. This example configures the protocol address family (IPv4 or IPv6), secret key rotation interval (120 minutes), and tunnel limit (10) for AMT Relay function. In addition, the AMT anycast prefix is set to 192.0.2.1/32 for IPv4 and 2001:db8::1/128 for IPv6, and the AMT local address is configured to 198.51.100.42 for IPv4 and 2001:db8:abcd:12::42 for IPv6.

```
<?xml version="1.0" encoding="UTF-8"?>
<routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing"
         xmlns:rt="urn:ietf:params:xml:ns:yang:ietf-routing">
  <control-plane-protocols>
    <amt xmlns="urn:ietf:params:xml:ns:yang:ietf-amt">
      <relay>
        <addresses>
          <address>
            <family>rt:ipv4</family>
            <anycast-prefix>192.0.2.1/32</anycast-prefix>
            <local-address>198.51.100.42</local-address>
          </address>
          <address>
            <family>rt:ipv6</family>
            <anycast-prefix>2001:db8::1/128</anycast-prefix>
            <local-address>2001:db8:abcd:12::42</local-address>
          </address>
        </addresses>
        <tunnel-limit>10</tunnel-limit>
        <secret-key-rotation-interval>
          120
        </secret-key-rotation-interval>
      </relay>
    </amt>
  </control-plane-protocols>
</routing>
```

Figure 7: Data Model Example for Relay in XML

Figure 8 shows the same sample configuration for an AMT Relay service in JSON format.

```
{
  "ietf-routing:routing": {
    "control-plane-protocols": {
      "ietf-amt:amt": {
        "relay": {
          "addresses": {
            "address": [
              {
                "family": "ietf-routing:ipv4",
                "anycast-prefix": "192.0.2.1/32",
                "local-address": "198.51.100.42"
              },
              {
                "family": "ietf-routing:ipv6",
                "anycast-prefix": "2001:db8::1/128",
                "local-address": "2001:db8:abcd:12::42"
              }
            ]
          },
          "tunnel-limit": 10,
          "secret-key-rotation-interval": 120
        }
      }
    }
  }
}
```

Figure 8: Data Model Example for Relay in JSON

Figure 9 shows a sample configuration for an AMT Gateway service in XML format. This example configures two AMT pseudo-interfaces, amt0 and amt1. The Relay Discovery method is set to send an AMT Discovery message. For amt0, the Relay Discovery Address is configured as 192.0.2.1; for amt1, it is 2001:db8::1. Additionally, the initial time to wait for a response to a Relay Discovery message is set to 60 seconds for both interfaces. Likewise, the initial time to wait for a response to a Request message is also configured to 60 seconds for both amt0 and amt1.

```
<?xml version="1.0" encoding="UTF-8"?>
<interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
            xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
  <interface>
    <name>Tunnel0</name>
    <type>ianaift:tunnel</type>
    <enabled>true</enabled>
  </interface>
  <interface>
    <name>Tunnel1</name>
    <type>ianaift:tunnel</type>
    <enabled>true</enabled>
  </interface>
</interfaces>
<routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <control-plane-protocols>
    <amt xmlns="urn:ietf:params:xml:ns:yang:ietf-amt">
      <gateway>
        <pseudo-interfaces>
          <interface>
            <name>Tunnel0</name>
            <discovery-method>by-amt-solicit</discovery-method>
            <relay-discovery-address>192.0.2.1</
             relay-discovery-address>
            <discovery-timeout>60</discovery-timeout>
            <request-timeout>60</request-timeout>
          </interface>
          <interface>
            <name>Tunnel1</name>
            <discovery-method>by-amt-solicit</discovery-method>
            <relay-discovery-address>2001:db8::1</
             relay-discovery-address>
            <discovery-timeout>60</discovery-timeout>
            <request-timeout>60</request-timeout>
          </interface>
        </pseudo-interfaces>
      </gateway>
    </amt>
  </control-plane-protocols>
</routing>
```

Figure 9: Data Model Example for Gateway in XML

Figure 10 shows the same sample configuration for an AMT Gateway service in JSON format.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "Tunnel0",
        "type": "iana-if-type:tunnel",
        "enabled": true
      },
      {
        "name": "Tunnel1",
        "type": "iana-if-type:tunnel",
        "enabled": true
      }
    ]
  },
  "ietf-routing:routing": {
    "control-plane-protocols": {
      "ietf-amt:amt": {
        "gateway": {
          "pseudo-interfaces": {
            "interface": [
              {
                "name": "Tunnel0",
                "discovery-method": "by-amt-solicit",
                "relay-discovery-address": "192.0.2.1",
                "discovery-timeout": 60,
                "request-timeout": 60
              },
              {
                "name": "Tunnel1",
                "discovery-method": "by-amt-solicit",
                "relay-discovery-address": "2001:db8::1",
                "discovery-timeout": 60,
                "request-timeout": 60
              }
            ]
          }
        }
      }
    }
  }
}
```

Figure 10: Data Model Example for Gateway in JSON

Authors' Addresses

Yisong Liu (editor)
China Mobile
China
Email: liuyisong@chinamobile.com

Changwang Lin (editor)
New H3C Technologies
China
Email: linchangwang.04414@h3c.com

Zheng(Sandy) Zhang
ZTE Corporation
China
Email: zhang.zheng@zte.com.cn

Xuesong Geng
Huawei Technologies
China
Email: gengxuesong@huawei.com

Vinod Kumar Nagaraj
Juniper Networks
Email: vinkumar@juniper.net