Stream: Internet Engineering Task Force (IETF)
RFC: 10006
Category: Standards Track
Published: August 2026
ISSN: 2070-1721
Authors: K. Inamdar *Unaffiliated*
S. Narayanan *Unaffiliated*
C. Jennings *Cisco Systems*

RFC 10006
Automatic SIP Trunking and Peering

Abstract

This document specifies a framework that enables enterprise telephony Session Initiation Protocol (SIP) networks to solicit and obtain a capability set document from a SIP service provider. The capability set document encodes a set of characteristics that enable easy peering between enterprise and service provider SIP networks.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc10006.

Copyright Notice

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

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

Table of Contents

1. Introduction

The deployment of an infrastructure based on SIP [RFC3261] in enterprise and service provider communication networks is increasing at a rapid pace. Consequently, direct IP peering between enterprise and service provider networks is quickly replacing conventional methods of interconnection between enterprise and service provider networks. Currently published standards provide a strong foundation over which direct IP peering can be realized (note that "peering" and "trunking" can be used interchangeably). However, given the sheer number of these standards, it is often not clear which behavioral subsets, extensions to baseline protocols, and operating principles ought to be implemented by service provider and enterprise networks to ensure successful peering.

The SIPconnect technical recommendations [SIPconnect-TR] aim to solve this problem by providing a central reference that promotes seamless peering between enterprise and service provider SIP networks. However, despite the extensive set of implementation rules and operating guidelines, interoperability issues between service provider and enterprise networks persist. This is in large part because the guidelines of the technical specifications are not hard requirements that can be enforced by the peer. Consequently, enterprise administrators usually undertake a fairly rigorous regimen of testing, analysis, and troubleshooting to arrive at a configuration block that ensures seamless service provider peering. However, this workflow complements the SIPconnect technical recommendations, in that both endeavors aim to promote and achieve interoperability between the enterprise and service provider.

Another set of interoperability problems arise when enterprise administrators are required to translate a set of technical recommendations from service providers to configuration blocks across one or more devices in the enterprise network, which is usually an error-prone exercise. Additionally, such technical recommendations might not be nuanced enough to intuitively allow the generation of specific configuration blocks.

This document introduces the framework for Automatic Peering and Trunking over SIP by which an enterprise network can solicit a detailed capability set from a SIP service provider; the detailed capability set can subsequently be used by automation or an administrator to generate configuration blocks across one or more devices within the enterprise network to ensure successful service provider peering.

2. Requirements Language

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

3. Overview of Operations

This section provides a reference architecture against which the SIP Automatic Peering framework may be implemented. Additionally, terms that are commonly used in the context of the document are defined. Last, considerations for the configuration workflow and the choice of network transport between enterprise and service provider telephony networks are discussed.

3.1. Reference Architecture

Figure 1 illustrates a reference architecture that may be deployed to support the mechanism described in this document. The enterprise network consists of a SIP Private Branch Exchange (SIP-PBX), media endpoints (ME), and a Session Border Controller (SBC) [RFC7092]. It may also include additional components such as application servers for voicemail, recording, fax, etc. At a high level, the service provider consists of a SIP signaling entity (SP-SSE), a media entity for handling media streams of calls set up by the SP-SSE, and an HTTP [RFC9110] server that stores the capability set document (indicated as Cap Server in Figure 1).

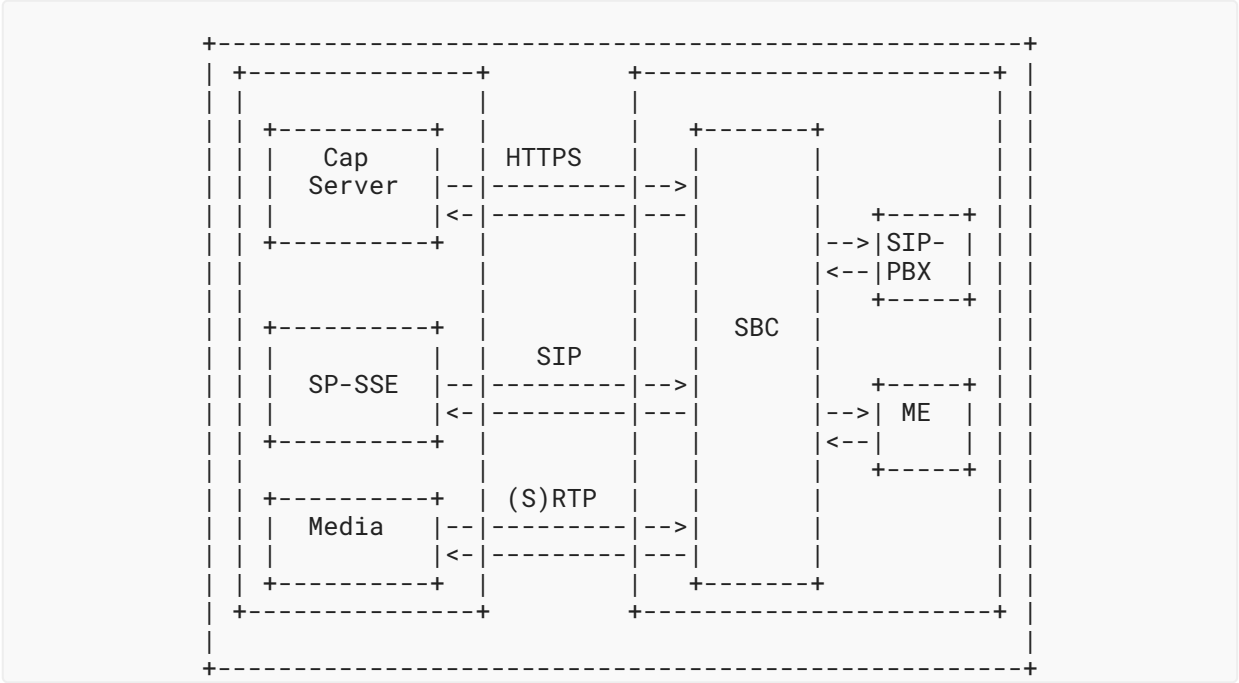

Figure 1: Reference Architecture

3.2. Terminology

This document makes use of the following terminology:

Enterprise Network:
: A communications network infrastructure deployed by an enterprise that interconnects with the service provider network over SIP. The enterprise network could include devices such as application servers, endpoints, call agents, and edge devices, among others.

Edge Device:
: A device that is the last hop in the enterprise network and that is the transit point for traffic entering and leaving the enterprise. An edge device is typically a back-to-back user agent (B2BUA) [RFC7092] such as a Session Border Controller (SBC).

Service Provider Network:
: A communications network infrastructure deployed by service providers. In the context of this document, the service provider network is accessible over SIP for the establishment, modification, and termination of calls and is accessible over HTTP for the transfer of the capability set document. The service provider network is also referred to as a SIP Service Provider (SSP) or Internet Telephony Service Provider (ITSP) network.

Call Control:
: Call control within telephony networks refers to software that is responsible for delivering core telephony functions. Call control not only provides the basic functionality of setting up, sustaining, and terminating calls, but it also provides the necessary control and logic required for additional services within the telephony network, such as registration of endpoints, integration with application servers (voicemail, instant messaging, presence), among others.

Capability Server:
: A server hosted in the service provider network, such that this server is the target for capability set document requests from the enterprise network.

Capability Set (or Capability Set Document):
: Refers collectively to a set of characteristics within the service provider network, which when communicated to the enterprise network, provides the enterprise network the information required to interconnect with the service provider network. The various parameters that constitute the capability set relate to characteristics that are specific to signaling, media, transport, and security. Certain aspects of interconnecting with service providers are out of scope of the capability set, for example, the access technology used to interconnect with service provider networks.

3.3. Configuration Workflow

A workflow that enables an enterprise network to solicit the capability set of a SIP service provider ought to take into account the following considerations:

- The configuration workflow must be based on a protocol or a set of protocols commonly used between enterprise and service provider telephony networks.
- The configuration workflow must be flexible enough to allow the service provider network to dynamically offload different capability sets to different enterprise networks based on the identity of the enterprise network.
- Capability set documents obtained as a result of the configuration workflow must be conducive to easy parsing by automation. Subsequently, automation may be used for the generation of appropriate configuration blocks on the edge element or across one or more elements in the enterprise network.

Taking the above considerations into account, this document proposes an HTTP-based workflow that the enterprise network can use to solicit and ultimately obtain the service provider capability set. The enterprise network creates a well-formed HTTP GET request to solicit the service provider capability set. Subsequently, the HTTP response from the SIP service provider includes the capability set. The capability set is encoded in JSON, thus ensuring that the response can be easily parsed by automation.

3.4. Transport

To solicit the capability set of a SIP service provider, the edge element in an enterprise network generates a well-formed HTTP GET request. There are two reasons why it makes sense for the enterprise edge element to generate the HTTP request:

1. Edge elements are devices that normalize any mismatches between the enterprise and service provider networks in the media and signaling planes. As a result, when the capability set is received from the SIP service provider network, the edge element can generate appropriate configuration blocks (possibly across multiple devices) to enable interconnection.
2. Given that edge elements are configured to "talk" to networks external to the enterprise, the complexity in terms of NAT traversal and firewall configuration would be minimal.

The HTTP GET request is targeted at a capability server that is managed by the SIP service provider such that this server processes, and on successfully processing the request, includes the capability set document in the response. The capability set document is constructed according to the guidelines of the YANG data model described in this document. The capability set document included in a successful response is formatted in JSON. More details about the formatting of the HTTP request and response are provided in Section 4.

There could be situations wherein an enterprise telephony network interconnects with its SIP service provider such that traffic between the two networks traverses an intermediary SIP service provider network. This could be a result of interconnect agreements between the terminating and transit SIP service provider networks. In such situations, the capability set provided to the enterprise network by its SIP service provider must account for the characteristics of the transit SIP service provider network from a signaling and media perspective. For example, if the terminating SIP service provider network supports the G.729 codec and the transit SIP service provider network does not, G.729 must not be advertised in the capability set. As another example, if the transit SIP service provider network does not support a SIP extension, for instance, the SIP extension for reliable provisional responses as defined in [RFC3262], the terminating SIP service provider network must not advertise support for this extension in the capability set provided to the enterprise network. How a terminating SIP service provider obtains the characteristics of the intermediary SIP service provider network is out of the scope of this document; however, one method could be for the terminating SIP service provider to obtain the characteristics of the intermediary SIP service provider by leveraging the YANG data model introduced in this document.

4. HTTP Transport

This section describes the use of HTTP [RFC9110] as a transport protocol for the peering workflow.

4.1. HTTP Methods

The workflow defined in this document leverages the HTTP GET method and its corresponding response(s) to request and subsequently obtain the service provider capability set document.

4.2. Integrity and Confidentiality

Peering requests and responses are defined over HTTP [RFC9110]. However, due to the sensitive nature of information transmitted between client and server, it is required to secure HTTP communications using Transport Layer Security (TLS) [RFC8446]; therefore, the enterprise edge element and the capability server **MUST** support TLS version 1.2 [RFC5246] or later [RFC8446]. When HTTP/3 [RFC9114] is used, TLS is incorporated within QUIC for the transport of the capability set document. The usage of SIP or RTP-over-QUIC is beyond the scope of this document. Additionally, the enterprise edge element and capability server **MUST** support the use of the https URI scheme as defined in [RFC9110].

4.3. Authenticated Client Identity

HTTP usually adopts asymmetric methods of authentication. For example, clients typically use certificate-based authentication to verify the server they are talking to, whereas servers typically use methods such as HTTP digest authentication or OAuth 2.0 [RFC6749] to authenticate clients. Though OAuth 2.0 is not an authentication protocol, it nonetheless allows for client authentication to be carried out with the use of OAuth tokens.

In the context of the SIP Automatic Peering framework, OAuth 2.0 **MUST** be used to carry out client authentication. Enterprise edge elements could use the various grant types outlined in the OAuth 2.0 specification and supported by the service provider in order to obtain the capability set document. This document does not mandate a specific grant type. The implementation of OAuth 2.0 to obtain the capability set is beyond the scope of this document. However, it provides an example of how an enterprise SBC could leverage the authorization code grant flow (Section 4.1 of [RFC6749]) to acquire the capability set document from the service provider in Figure 2.

Using the resource owner password credentials grant type (Section 1.3.3 of [RFC6749]) requires the existence of a trust relationship between the resource owner (in this context, the administrator/enterprise network) and the client (in this context, an edge element such as an SBC). In SIP trunking deployments between enterprise and service provider networks, such a trust relationship between the client (edge element) and the administrator, resource owner, and enterprise network already exists, as SIP trunk registration (and refreshing registrations) require credentials, typically a username and password, that are configured on the edge element by the administrator. However, it is important for the enterprise network administrator and service provider to factor in security issues associated with this grant type.

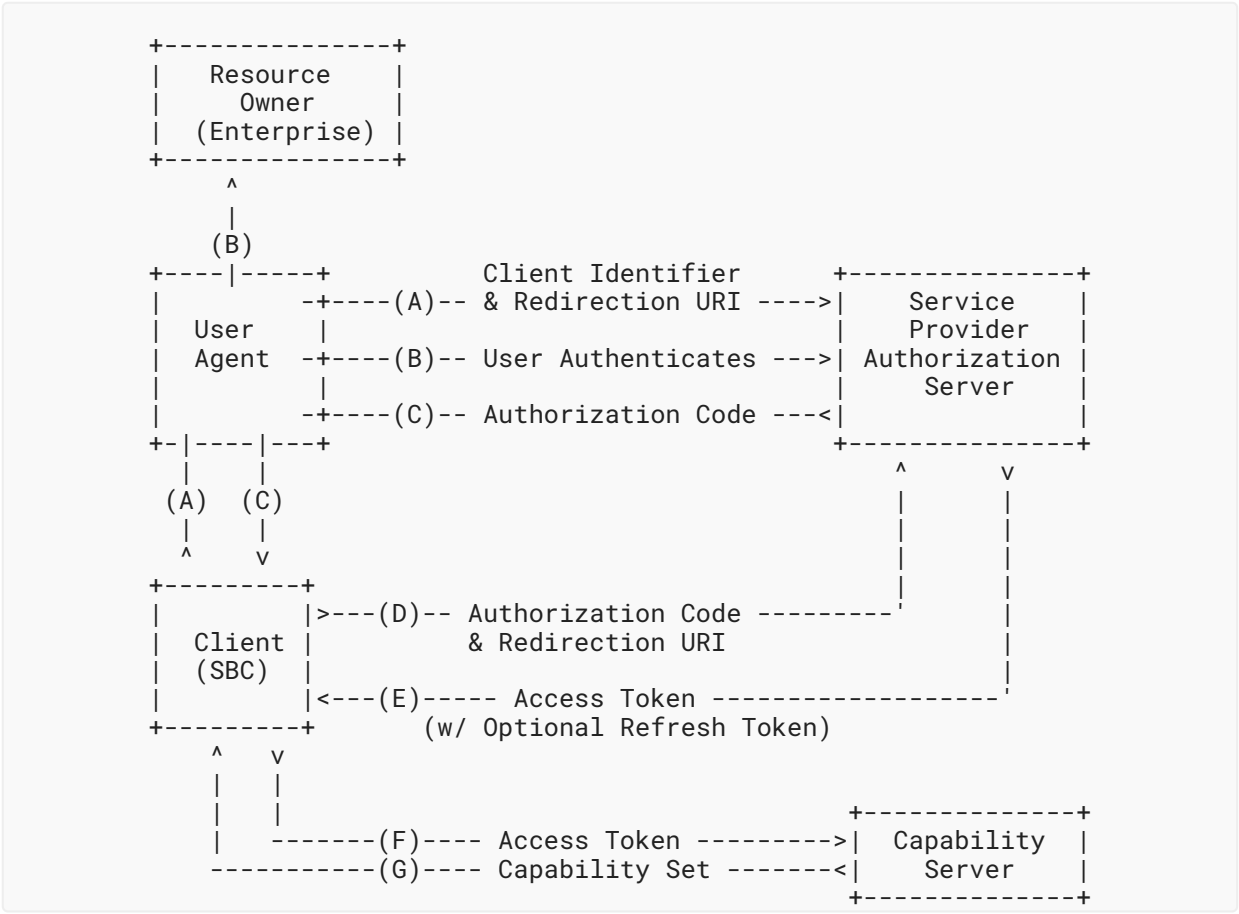

Figure 2: Client Authentication Mechanism

The flow illustrated in Figure 2 includes the following steps:

A. The enterprise SBC (client) initiates the flow by directing the resource owner's (enterprise network administrator) user agent to the authorization endpoint. The SBC includes its client identifier, requested scope, local state, and a redirection URI to which the authorization server will send the user agent back once access is granted (or denied). As a precursor to the flow, the enterprise network administrator has already obtained a unique client identifier for their network and provided a redirection URI populated with a target within their network to obtain the authorization code.

B. The authorization server within the service provider network authenticates the network administrator (via the user agent) and establishes whether the network administrator grants or denies the client's access request.

C. Assuming the network administrator grants access, the authorization server redirects the user agent back to the enterprise SBC using the redirection URI provided earlier (in the request or during client registration). The redirection URI includes an authorization code and any local state provided by the client earlier.

D. The enterprise SBC requests an access token from the authorization server's token endpoint by including the authorization code received in the previous step. When making the request,

the enterprise SBC authenticates with the authorization server and includes the redirection URI used to obtain the authorization code for verification.

E. The authorization server authenticates the enterprise SBC, validates the authorization code, and ensures that the redirection URI received matches the URI used to redirect the SBC in step (C). If valid, the authorization server responds back with an access token and, optionally, a refresh token.

F. The enterprise SBC then contacts the capability server located in the service provider network with an HTTP GET request along with the access token to retrieve the capability set document.

G. The capability server checks for a valid access token and returns the capability set document to the enterprise SBC. The service provider will host a unique document for each enterprise network that will peer with it.

4.4. Encoding the Request

The edge element in the enterprise network generates an HTTP GET request such that the request target is obtained using the procedure outlined in Section 4.5. This document does not specify any content negotiation. The server **MUST** set the response content type header to the application/json media type.

4.5. Identifying the Request Target

HTTP GET requests from enterprise edge elements **MUST** carry a valid request target. The enterprise edge element might obtain the URL of the resource hosted on the capability server in one of two ways:

1. Manual configuration
2. Discovery using the WebFinger protocol

The complete https URLs to be used when authenticating the enterprise edge element (optional) and obtaining the SIP service provider capability set can be obtained from the SIP service provider beforehand and entered into the edge element manually via some interface, for example, a CLI or GUI.

However, if the resource URL is unknown to the administrator (and by extension, to the edge element), the WebFinger protocol [RFC7033] and the sip-trunking-capability [RFC9409] link relation type may be leveraged assuming that the SIP service provider has implemented WebFinger within their network and hosts the capability set at the respective location.

If an enterprise edge element attempts to discover the URL of the endpoints hosted in the ssp1.example.com domain, it issues the following request.

```
GET /.well-known/webfinger?
    resource=https%3A%2F%2Fssp1.example.com
    rel=sipTrunkingCapability
    HTTP/1.1
Host: ssp1.example.com


HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/jrd+json

{
  "subject" : "https://ssp1.example.com",
  "links" :
  [
    {
      "rel" : "sipTrunkingCapability",
      "href" :
          "https://capserver.ssp1.com/capserver/capdoc.json"
    }
  ]
}
```

Once the target URI is obtained by an enterprise telephony network, the URI may be dereferenced to obtain a unique capability set document that is specific to that given enterprise telephony network. The ITSP may use credentials to determine the identity of the enterprise telephony network and provide the appropriate capability set document.

4.6. Generating Status Codes

Capability servers include the capability set documents in the body of a successful response. Capability set documents **MUST** be formatted in JSON. For requests that are incorrectly formatted (e.g., an incorrect query parameter in the URI), the capability server **MUST** generate a "400 Bad Request" status code for the incorrect request. If requests contain an invalid token, the capability server **MUST** generate a "403 Forbidden" status code clearly indicating that this token does not have the permission to view the capability set document.

The capability server can respond to client requests with redirect status codes (3xx).

The server **SHOULD** include the Location header field in such responses. If the Location header is not included with the status code, this can lead to the client being unable to find the capability set document, leading to a failure in the peering process or requiring manual intervention by an administrator.

The enterprise edge element **SHOULD** handle the 3xx status codes from the capability server in accordance with [RFC9110].

5. Monitoring for Updates

Given that the service provider capability set is largely expected to remain static, the work needed to implement an asynchronous push mechanism to encode minor changes in the capability set document (state deltas) is not commensurate with the benefits. Rather, enterprise edge elements can poll capability servers at predefined intervals to obtain the full capability set document. It is recommended that capability servers are polled every 24 hours. Alternatively, the enterprise edge elements can leverage preconditions specified in [RFC9110] to conditionally retrieve the capability set document if any changes have occurred.

6. Encoding the Service Provider Capability Set

In the context of this document, the capability set of a service provider refers collectively to a set of characteristics, which when communicated to an enterprise network, provides it with sufficient information to directly peer with the service provider network. The capability set document is not designed to encode extremely granular details of all features, services, and protocol extensions that are supported by the service provider network. For example, it is sufficient to encode that the service provider uses T.38 relay for faxing; it is not required to know the value of the "T38FaxFillBitRemoval" parameter.

The parameters within the capability set document represent a wide array of characteristics, such that these characteristics collectively disseminate sufficient information to enable direct IP peering between enterprise and service provider networks. The various parameters represented in the capability set are chosen based on existing practices and common problem sets typically seen between enterprise and service provider SIP networks.

7. Data Model for Capability Set

This section contains a tree diagram (Section 7.1), the YANG module [RFC7950] for encoding the service provider capability set (Section 7.2), and a discussion about extending the capability set (Section 7.3).

7.1. Tree Diagram

The meanings of the symbols in YANG tree diagrams are defined in "YANG Tree Diagrams" [RFC8340].

The data model for the peering capability document has the following structure:

```
module: ietf-sip-auto-peering
  +--ro sip-auto-peering
     +--ro variant           identityref
     +--ro revision
     |  +--ro not-before    yang:date-and-time
     |  +--ro location      inet:uri
```

```
      +--ro transport-info
      |  +--ro transport*        identityref
      |  +--ro registrar* [host port]
      |  |  +--ro host    union
      |  |  +--ro port    inet:port-number
      |  +--ro realm* [name]
      |  |  +--ro name        string
      |  |  +--ro username?   string
      |  |  +--ro password?   ianach:crypt-hash
      |  +--ro call-control* [host port]
      |  |  +--ro host    union
      |  |  +--ro port    inet:port-number
      |  +--ro dns-server*       inet:ip-address
      |  +--ro outbound-proxy* [host port]
      |     +--ro host    union
      |     +--ro port    inet:port-number
      +--ro call-spec
      |  +--ro early-media?         boolean
      |  +--ro signaling-forking?   boolean
      |  +--ro supported-method*    enumeration
      |  +--ro caller-id
      |  |  +--ro e164-format?        boolean
      |  |  +--ro preferred-method?   enumeration
      |  +--ro number-range* [index]
      |     +--ro index    uint16
      |     +--ro type?    enumeration
      |     +--ro count?   uint16
      |     +--ro value*   string
      +--ro media
      |  +--ro media-type-audio* [media-format]
      |  |  +--ro media-format    identityref
      |  |  +--ro rate?           uint16
      |  |  +--ro ptime?          uint8
      |  |  +--ro parameter?      string
      |  +--ro fax
      |  |  +--ro protocol*   enumeration
      |  +--ro rtp
      |  |  +--ro rtp-trigger?     boolean
      |  |  +--ro symmetric-rtp?   boolean
      |  +--ro rtcp
      |     +--ro symmetric-rtcp?   boolean
      |     +--ro rtcp-feedback?    boolean
      +--ro dtmf
      |  +--ro payload-number?   uint8
      |  +--ro iteration?        boolean
      +--ro security
      |  +--ro signaling
      |  |  +--ro secure?    boolean
      |  |  +--ro version*   identityref
      |  +--ro media-security
      |  |  +--ro key-management*   enumeration
      |  +--ro certificate-location?        inet:uri
      |  +--ro secure-telephony-identity
      |     +--ro stir-compliance?          boolean
      |     +--ro certificate-delegation?   boolean
      |     +--ro acme-directory?           inet:uri
      +--ro extension*        iana-sip-option-tags:sip-option-tag
```

7.2. YANG Data Model

This section defines the YANG module for the peering capability set document. This module depends on existing YANG modules that provide common YANG data types [RFC9911] and system management [RFC7317]. In addition, this YANG module references [RFC2833], [RFC4585], [RFC4568], [RFC4733], [RFC4855], [RFC4961], [RFC5764], [RFC6716], [RFC7362], [RFC8555], [RFC9645], [iana-crypt-hash], and [iana-sip-option-tags].

```
<CODE BEGINS> file "ietf-sip-auto-peering@2026-06-15.yang"

module ietf-sip-auto-peering {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-sip-auto-peering";
  prefix sipap;

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 9911: Common YANG Data Types.";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 9911: Common YANG Data Types.";
  }
  import iana-crypt-hash {
    prefix ianach;
    reference
      "https://www.iana.org/assignments/iana-crypt-hash/";
  }
  import ietf-tls-common {
    prefix tlscmn;
    reference
      "RFC 9645: YANG Groupings for TLS Clients and TLS Servers.";
  }
  import iana-sip-option-tags {
    prefix iana-sip-option-tags;
    reference
      "https://www.iana.org/assignments/iana-sip-option-tags/";
  }

  organization
    "IETF ASAP (Automatic SIP trunking And Peering) Working Group";
  contact
    "WG Web: <https://datatracker.ietf.org/wg/asap/>
     WG List: <mailto:asap@ietf.org>

     Editor: Kaustubh Inamdar
     <mailto:kaustubh.ietf@gmail.com>

     Editor: Sreekanth Narayanan
     <mailto:sknth.n@protonmail.com>

     Editor: Cullen Jennings
```

```
      <mailto:fluffy@iii.ca>";
  description
    "Data model for encoding SIP service provider capability set.

     This YANG module defines a read-only data model intended for
     exchanging SIP service provider capabilities with enterprise
     networks.  The data is published by service providers and
     consumed by enterprises via an out-of-band interface.

     This module does NOT provide configuration capabilities; it
     serves purely as a standardized format for capability exchange.
     Service providers generate and host capability documents based
     on this schema, which enterprises retrieve and use to configure
     their SIP equipment.

     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 10006; see the
     RFC itself for full legal notices.";

  revision 2026-06-15 {
    description
      "Initial version";
    reference
      "RFC 10006: Automatic SIP Trunking and Peering";
  }

  identity capability-doc-variant {
    description
      "Base for capability document variants.";
  }

  identity v1-0 {
    base capability-doc-variant;
    description
      "Variant 1.0 of the capability set document.";
  }

  identity sip-transport-protocol {
    description
```

```
      "Base for transport protocols used to send SIP requests
       across.";
  }

  identity udp {
    base sip-transport-protocol;
    description
      "UDP used for SIP requests and responses.";
  }

  identity tcp {
    base sip-transport-protocol;
    description
      "TCP used for SIP requests and responses.";
  }

  identity codec-variant {
    description
      "Base for variants of codec supported by the service
       provider.";
  }

  identity pcmu {
    base codec-variant;
    description
      "PCMU (G.711 μ-law) audio codec.";
  }

  identity pcma {
    base codec-variant;
    description
      "PCMA (G.711 A-law) audio codec.";
  }

  identity opus {
    base codec-variant;
    description
      "Opus audio codec.";
    reference
      "RFC 6716: Definition of the Opus Audio Codec.";
  }

  identity g722 {
    base codec-variant;
    description
      "G.722 audio codec.";
  }

  identity g729 {
    base codec-variant;
    description
      "G.729 codec.";
  }

  grouping entity {
    description
      "Grouping that provides a reusable list named 'entity', with
       each entry containing a host and a port.";
```

```
    leaf host {
      type union {
        type inet:ip-address;
        type inet:domain-name;
      }
      description
        "IP address or host name of the entity.";
    }
    leaf port {
      type inet:port-number;
      description
        "Entity's port number.";
    }
  }

  container sip-auto-peering {
    config false;
    description
      "Root container for SIP service provider capability data.  This
       container holds read-only operational data that represents the
       capabilities and requirements of a SIP service provider.
       Enterprise networks retrieve this data to automatically
       configure their SIP trunking parameters.";
    leaf variant {
      type identityref {
        base capability-doc-variant;
      }
      mandatory true;
      description
        "A node that identifies the version number of the capability
         set document.  RFC 10006 defines the parameters for
         variant 1.0; future specifications might define a richer
         parameter set, in which case the variant must be changed
         to 2.0, 3.0, and so on.  Future extensions to the
         capability set document MUST also ensure that the
         corresponding YANG module is defined.";
      reference
        "RFC 10006: Automatic SIP Trunking and Peering";
    }
    container revision {
      description
        "A container that encapsulates information regarding the
         availability of a new version of the capability set document
         for the enterprise.";
      leaf not-before {
        type yang:date-and-time;
        mandatory true;
        description
          "A node that identifies the absolute UTC time at which the
           parameters in this capability set document are activated
           or considered valid.  This node has been set to mandatory
           as it is the service provider's responsibility to inform
           when new peering settings take effect.  Without being
           aware of a start time, the enterprise network will
           experience failures.";
      }
      leaf location {
        type inet:uri;
```

```
        mandatory true;
        description
          "A node that identifies the URL of a new revision of the
           service provider capability set document.  Without this
           URL, an enterprise network would not be aware of changes
           that have occurred in the service provider network.";
      }
    }
    container transport-info {
      description
        "A container that encapsulates transport characteristics of
         SIP sessions between enterprise and service provider
         networks.";
      leaf-list transport {
        type identityref {
          base sip-transport-protocol;
        }
        min-elements 1;
        description
          "A list that enumerates the different transport-layer
           protocols supported by the SIP service provider.  Valid
           transport-layer protocols include UDP, TCP, and TLS.";
      }
      list registrar {
        key "host port";
        uses entity;
        max-elements 3;
        description
          "A list that specifies the transport address of one or more
           registrar servers in the service provider network.  The
           transport address of the registrar can be provided using a
           combination of a valid IP address and port number, a
           subdomain of the SIP service provider network, or the
           fully qualified domain name (FQDN) of the SIP service
           provider network.  If the transport address of a registrar
           is specified using either a subdomain or a FQDN, the DNS
           element must be populated with one or more valid DNS
           server IP addresses.";
      }
      list realm {
        key "name";
        description
          "A container that encapsulates the set of realms or
           protection domains the SIP service provider is responsible
           for.";
        leaf name {
          type string;
          description
            "A node specifying the SIP service provider realm or
             protection domain.  This node is encoded as a string;
             the value of this node must be identical to the value of
             the 'realm' parameter in a WWW-Authenticate header field
             that the SIP service provider might send in response to
             requests that do not contain a valid Authorization
             header field.";
        }
        leaf username {
          type string;
```

```
          description
            "A node that encodes the username for the given realm.
             The username is one of many inputs used by the
             enterprise network in generating the response parameter
             of the Authorization header field.";
        }
        leaf password {
          type ianach:crypt-hash;
          description
            "A node that encodes the password for the given realm.
             The password is one of many inputs used by the
             enterprise network in generating the response parameter
             of the Authorization header field.  The password is
             stored as a cryptographic hash.";
        }
      }
      list call-control {
        key "host port";
        uses entity;
        max-elements 3;
        description
          "A list that specifies the transport address of the call
           server(s) in the service provider network.  The enterprise
           network must use an applicable transport protocol in
           conjunction with the call control server(s) transport
           address when transmitting call setup requests.  The
           transport address of a call server(s) within the service
           provider network can be specified using a combination of
           a valid IP address and port number, a subdomain of the
           SIP service provider network, or a FQDN of the SIP service
           provider network.  If the transport address of a call
           control server(s) is specified using either a subdomain or
           a FQDN, the DNS element must be populated with one or more
           valid DNS server IP addresses.  The transport address
           specified in this element can also serve as the target for
           non-call requests such as SIP OPTIONS.";
      }
      leaf-list dns-server {
        type inet:ip-address;
        max-elements 2;
        description
          "A list that encodes the IP address of one or more DNS
           servers hosted by the SIP service provider.  If the
           enterprise network is unaware of the IP address, port
           number, and transport protocol of servers within the
           service provider network (for example, the registrar
           and call control server), it must use DNS NAPTR and
           SRV.  Alternatively, if the enterprise network has the
           FQDN of the SIP service provider network, it must use
           DNS to resolve the said FQDN to an IP address.
           The dns element encodes the IP address of one or more
           DNS servers hosted in the service provider network.
           If, however, either the registrar or call-control lists
           or both are populated with a valid IP address and port
           pair, the dns element can be omitted.";
      }
      list outbound-proxy {
        key "host port";
```

```
        uses entity;
        description
          "A list that specifies the transport address of one or more
           outbound proxies.  The transport address can be specified
           by using a combination of an IP address and a port number,
           a subdomain of the SIP service provider network, or a FQDN
           and port number of the SIP service provider network.
           If the outbound-proxy list is populated with a valid
           transport address, it represents the default destination
           for all outbound SIP requests; therefore, the registrar
           and call-control lists can be omitted.";
      }
    }
    container call-spec {
      description
        "A container that encapsulates information about call
         specifications, restrictions, and additional handling
         criteria for SIP calls between the enterprise and service
         provider network.";
      leaf early-media {
        type boolean;
        description
          "A node that specifies whether the service provider network
           is expected to deliver in-band announcements/tones before
           call connect.  The P-Early-Media header field can be used
           to indicate pre-connect delivery of tones and
           announcements on a per-call basis.  However, given that
           signaling and media could traverse a large number of
           intermediaries with varying capabilities (in terms of
           handling of the P-Early-Media header field) within the
           enterprise, such devices can be appropriately configured
           for media cut through if it is known beforehand that
           early media is expected for some or all of the outbound
           calls.  This element is a boolean type, where a value of
           true signifies that the service provider is capable of
           early media.  A value of false signifies that the service
           provider is not expected to generate early media.";
      }
      leaf signaling-forking {
        type boolean;
        description
          "A node that specifies whether outbound call requests from
           the enterprise might be forked on the service provider
           network that MAY lead to multiple early dialogs.  This
           information would be useful to the enterprise network in
           appropriately handling multiple early dialogs reliably
           and in enforcing local policy.  This element is a boolean
           type, where a value of true signifies that the service
           provider network can potentially fork outbound call
           requests from the enterprise.  A value of false indicates
           that the service provider will not fork outbound call
           requests.";
      }
      leaf-list supported-method {
        type enumeration {
          enum invite {
            description
              "Initiate a dialog or session.";
```

```
          }
          enum ack {
            description
              "Acknowledge final response to INVITE.";
          }
          enum bye {
            description
              "Terminate a dialog or session.";
          }
          enum cancel {
            description
              "Cancel a pending request.";
          }
          enum register {
            description
              "Register contact information.";
          }
          enum options {
            description
              "Query capabilities of a server.";
          }
          enum prack {
            description
              "Provisional acknowledgement.";
          }
          enum subscribe {
            description
              "Subscribe to an event.";
          }
          enum notify {
            description
              "Notify subscriber of an event.";
          }
          enum publish {
            description
              "Publish an event state.";
          }
          enum info {
            description
              "Send mid-session information.";
          }
          enum refer {
            description
              "Refer recipient to a third party.";
          }
          enum message {
            description
              "Instant message transport.";
          }
          enum update {
            description
              "Update session parameters within a dialog.";
          }
        }
        description
          "A list that specifies the various SIP methods supported by
           the SIP service provider.  The list of supported methods
           help to appropriately configure various devices within the
```

```
            enterprise network.  For example, if the service provider
            enumerates support for the OPTIONS method, the enterprise
            network could periodically send OPTIONS requests as a
            keep-alive mechanism.";
        }
        container caller-id {
          description
            "A container that encodes the preferences of SIP service
             providers in terms of calling number presentation by the
             enterprise network.  Certain ITSPs require that the
             calling number be formatted in E.164, whereas others place
             no such restrictions.  Additionally, some ITSPs require
             that the calling number be included in a specific SIP
             header field, for example, the P-Asserted-ID header field
             or the From header field, whereas others place no
             restrictions on the specific SIP header field used to
             convey the calling number.";
          leaf e164-format {
            type boolean;
            description
              "A node that indicates whether the service provider
               requires the enterprise network to normalize the calling
               number into E.164 format.  A value of true mandates the
               enterprise network to format calling numbers to E.164
               format, while a value of false leaves the formatting
               of the calling number up to the enterprise network.";
          }
          leaf preferred-method {
            type enumeration {
              enum p-asserted-identity {
                description
                  "Use the P-Asserted-Identity header to determine
                   remote party identity.";
              }
              enum from {
                description
                  "Use the From header to determine remote party
                   identity.";
              }
            }
            description
              "A node that specifies which SIP header MUST be used
               by the enterprise network to communicate caller
               information.  The value of this node is a string that
               contains the name of the SIP header required to
               carry caller information.";
          }
        }
        list number-range {
          key "index";
          description
            "A list that specifies the Direct Inward Dial (DID) number
             range allocated to the enterprise network by the SIP
             service provider.  The DID number ranges allocated by the
             service provider to the enterprise network might be a
             contiguous or a non-contiguous block.  The number ranges
             allocated to an enterprise can be communicated as a value
             or as a reference.  For large enterprise networks, the
```

```
           size of the DID range might run into several hundred
           numbers. For situations in which the enterprise is
           allocated a large DID number range or a non-contiguous
           number range, it is RECOMMENDED that the SIP service
           provider communicate this information by reference, that
           is, through a URL.  The enterprise network is required to
           dereference this URL in order to obtain the DID number
           ranges allocated by the SIP service provider.";
        leaf index {
          type uint16;
          description
            "Index for the number ranges.";
        }
        leaf type {
          type enumeration {
            enum range {
              description
                "Numbers specified as a range.";
            }
            enum collection {
              description
                "Numbers specified in the form of a collection.";
            }
            enum reference {
              description
                "Number range available at a URL.";
            }
          }
          description
            "A node that indicates whether the DID range
             is communicated by value or by reference.  It can have a
             value of 'range', 'collection', or 'reference'.";
        }
        leaf count {
          when "../type = 'range' or ../type = 'collection'";
          type uint16;
          description
            "Indicates the size of the DID number range.  This leaf
             MUST NOT be included when using the 'reference'
             type.";
        }
        leaf-list value {
          type string;
          description
            "A list that encapsulates the DID number range allocated
             to the enterprise.  If the num-ranges 'type' is set to
             'range' or 'collection', the 'count' node MUST have a
             valid, non-zero, positive integer.  If the number-range
             'type' value is set to 'range', then the number in this
             field represents the first phone number of a DID range
             allocated to the enterprise.  The value of subsequent
             numbers of the given DID range are obtained by adding
             one to the value of this field.  The number of times we
             need to add one is indicated by the 'count' field.";
        }
      }
    }
    container media {
```

```
      description
        "A container that is used to collectively encapsulate the
         characteristics of UDP-based audio streams.  A future
         extension to RFC 10006 may extend the media container
         to describe other media types.  The media container is
         also used to encapsulate basic information about
         Real-Time Transport Protocol (RTP) and Real-Time
         Transport Control Protocol (RTCP) from the perspective
         of the service provider network.  At the time of writing
         RFC 10006, video media streams are not exchanged
         between enterprise and service provider SIP networks.";
      reference
        "RFC 10006: Automatic SIP Trunking and Peering";
      list media-type-audio {
        key "media-format";
        description
          "A list encoding the various audio media formats
           supported by the SIP service provider.  The relative
           ordering of different media formats in the list indicates
           preference from the perspective of the service provider.
           Each element in the list begins with the encoding name
           of the media format, which is the same encoding name as
           used in the 'RTP/AVP' and 'RTP/SAVP' profiles.  The
           encoding name is followed by the sampling rate for the
           encoding and the packetization time.  Additionally, any
           other required and optional parameters for the given media
           format as specified when the media format is registered
           are described the 'param' field.
           Given that the parameters of media formats can vary from
           one communication session to another (e.g., across two
           separate communication sessions), the packetization
           time (ptime) used for the PCMU media format might vary
           from 10 to 30 ms, and the parameters included in the
           format element must be the ones that are expected to be
           invariant from the perspective of the service provider.
           Providing information about supported media formats and
           their respective parameters allows enterprise networks to
           configure the media plane characteristics of various
           devices such as endpoints and middleboxes.";
        reference
          "RFC 4855: Media Type Registration of RTP Payload Formats";
        leaf media-format {
          type identityref {
            base codec-variant;
          }
          description
            "The audio media format.";
        }
        leaf rate {
          type uint16;
          units "Hz";
          description
            "Sampling rate in Hz.";
        }
        leaf ptime {
          type uint8;
          units "milliseconds";
          description
```

```
            "Packetization time in milliseconds.";
        }
        leaf parameter {
          type string;
          description
            "Optional parameter for additional media details
             regarding the encoding.";
        }
      }
      container fax {
        description
          "A container that encapsulates the fax
           protocol(s) supported by the SIP service provider.  The
           fax container encloses a list (protocol) that enumerates
           whether the service provider supports T.38 relay,
           protocol-based fax passthrough, or both.  The relative
           ordering of nodes within the lists indicates preference.";
        leaf-list protocol {
          type enumeration {
            enum pass-through {
              description
                "Protocol-based fax passthrough.";
            }
            enum t38 {
              description
                "T.38 relay.";
            }
          }
          max-elements 2;
          description
            "List indicating the different fax protocols supported by
             the service provider.";
        }
      }
      container rtp {
        description
          "A container that encapsulates generic characteristics of
           RTP sessions between the enterprise and service provider
           network.";
        leaf rtp-trigger {
          type boolean;
          description
            "A node indicating whether the SIP service
             provider network always expects the enterprise network
             to send the first RTP packet for an established
             communication session.  This information is useful in
             scenarios such as 'hairpinned' calls, in which the
             caller and callee are on the service provider network
             and, because of sub-optimal media routing, an enterprise
             device such as an SBC is retained in the media path.
             Based on the encoding of this node, it is possible to
             configure enterprise devices such as SBCs to start
             streaming media (possibly filled with silence payloads)
             toward the address:port tuples provided by caller and
             callee.  This node is a boolean type.  A value of true
             indicates that the service provider expects the
             enterprise network to send the first RTP packet, whereas
             a value of false indicates that the service provider
```

```
            network does not require the enterprise network to send
            the first media packet.  While the practice of
            preserving the enterprise network in a hairpinned call
            flow is fairly common, it is recommended that SIP
            service providers avoid this practice.  In the context
            of a hairpinned call, the enterprise device retained in
            the call flow can easily eavesdrop on the conversation
            between the offnet parties.";
        }
        leaf symmetric-rtp {
          type boolean;
          description
            "A node indicating whether the SIP service provider
             expects the enterprise network to use symmetric RTP.
             Enforcement of this requirement by service providers
             on enterprise networks is typically useful in scenarios
             such as media latching.  This node is a boolean type.  A
             value of true indicates that the service provider
             expects the enterprise network to use symmetric RTP,
             whereas a value of false indicates that the enterprise
             network can use asymmetric RTP.";
          reference
            "RFC 4961: Symmetric RTP / RTP Control Protocol (RTCP),
             RFC 7362: Latching: Hosted NAT Traversal (HNT) for Media
             in Real-Time Communication";
        }
      }
      container rtcp {
        description
          "A container that encapsulates generic characteristics of
           RTCP sessions between the enterprise and service provider
           network.";
        leaf symmetric-rtcp {
          type boolean;
          description
            "A node indicating whether the SIP service
             provider expects the enterprise network to use symmetric
             RTCP.  This node is a boolean type.  A value of true
             indicates that the service provider expects symmetric
             RTCP reports, whereas a value of false indicates that
             the enterprise can use asymmetric RTCP.";
          reference
            "RFC 4961: Symmetric RTP / RTP Control Protocol (RTCP)";
        }
        leaf rtcp-feedback {
          type boolean;
          description
            "A node that indicates whether the SIP service
             provider supports the RTP profile extension for
             RTCP-based feedback.  Media sessions spanning
             enterprise and service provider networks are rarely
             made to flow directly between the caller and callee;
             rather, it is often the case that media traffic flows
             through network intermediaries such as SBCs.
             As a result, RTCP traffic from the service provider
             network is intercepted by these intermediaries, which
             in turn can either pass across RTCP traffic unmodified
             or modify RTCP traffic before it is forwarded to the
```

```
            endpoint in the enterprise network.  Modification of
            RTCP traffic would be required, for example, if the
            intermediary has performed media payload transformation
            operations such as transcoding or transrating.
            In a similar vein, for the RTCP-based feedback mechanism
            as defined in RFC 4585 to be truly effective,
            intermediaries must ensure that feedback messages are
            passed reliably and with the correct formatting to
            enterprise endpoints.
            This might require additional configuration and
            considerations that need to be dealt with at the time
            of provisioning the intermediary device.  This node
            is a boolean type.  A value of true indicates that the
            service provider supports the RTP profile extension for
            RTP-based feedback, and a value of false indicates that
            the service provider does not support the RTP profile
            extension for RTP-based feedback.";
          reference
            "RFC 4585: Extended RTP Profile for Real-time Transport
             Control Protocol (RTCP)-Based Feedback (RTP/AVPF)";
        }
      }
    }
    container dtmf {
      description
        "A container that describes the various aspects of
         DTMF relay via RTP Named Telephony Events.  The dtmf
         container allows SIP service providers to specify two facets
         of DTMF relay via Named Telephony Events.";
      leaf payload-number {
        type uint8 {
          range "96..127";
        }
        description
          "Indicates the payload type number.";
      }
      leaf iteration {
        type boolean;
        description
          "A value of true indicates that the service provider
           supports the newer standard while a value of false
           indicates that the service provider prefers the
           older standard";
        reference
          "RFC 4733: RTP Payload for DTMF Digits, Telephony
           Tones, and Telephony Signals,
           RFC 2833: RTP Payload for DTMF Digits, Telephony
           Tones and Telephony Signals";
      }
    }
    container security {
      description
        "A container that encapsulates characteristics about
         encrypting signaling streams between the enterprise
         and SIP service provider networks.";
      container signaling {
        description
          "A container that encapsulates the type of security
```

```
           protocol for the SIP communication between the
           enterprise SBC and the service provider.";
        leaf secure {
          type boolean;
          description
            "A node that specifies whether the service provider
             allows the use of TLS to secure SIP signaling
             messages between the enterprise and service provider
             network.  This node is a boolean type.  A value of
             true indicates that the service provider supports
             SIP sessions over TLS, whereas a value of false
             indicates that the service provider does not support
             SIP over TLS.";
        }
        leaf-list version {
          when "../secure = 'true'";
          type identityref {
            base tlscmn:tls-version-base;
          }
          description
            "A list that specifies the version(s) of TLS supported.";
        }
      }
      container media-security {
        description
          "A container that describes the various characteristics of
           securing media streams between enterprise and service
           provider networks.";
        leaf-list key-management {
          type enumeration {
            enum sdes {
              description
                "Simplified Data Encryption Standard (SDES)
                 key management.";
            }
            enum dtls-srtp {
              description
                "Secure Real-time Transport Protocol (SRTP) keys
                 managed using DTLS.";
            }
          }
          description
            "A list that specifies the key management method(s)
             used by the service provider.  Possible values in this
             list include 'SDES' and 'DTLS-SRTP'.";
          reference
            "RFC 4568: Session Description Protocol (SDP) Security
             Descriptions for Media Streams,
             RFC 5764: Datagram Transport Layer Security (DTLS)
             Extension to Establish Keys for the Secure Real-time
             Transport Protocol (SRTP)";
        }
      }
      leaf certificate-location {
        type inet:uri;
        description
          "If the enterprise network is required to exchange SIP
           traffic over TLS with the SIP service provider, and if the
```

```
           SIP service provider is capable of accepting TLS
           connections from the enterprise network, it may be
           required for the SIP service provider certificates to be
           pre-installed on the enterprise edge element.  In such
           situations, the certificate-location node is populated
           with a URL, which when dereferenced, provides a single
           Privacy-Enhanced Mail (PEM) encoded file that contains all
           certificates in the chain of trust.";
      }
      container secure-telephony-identity {
        description
          "Encapsulates Secure Telephony Identity (STIR)
           characteristics.";
        leaf stir-compliance {
          type boolean;
          description
            "A node that indicates whether the SIP service
             provider is STIR compliant.  This node is a boolean
             type.  A value of true indicates that the SIP service
             provider is STIR compliant.  A value of false indicates
             that the SIP service provider is not STIR compliant.  A
             SIP service provider being STIR compliant has
             implications for inbound and outbound calls, from the
             perspective of the enterprise network.";
        }
        leaf certificate-delegation {
          type boolean;
          description
            "A node that indicates whether a SIP service
             provider that allocates one or more number ranges to an
             enterprise network is willing to delegate authority to
             the enterprise network over that number range(s).  This
             node is a boolean type.  A value of true indicates that
             the SIP service provider is willing to delegate
             authority to the enterprise network over one or more
             number ranges.  A value of false indicates that the SIP
             service provider is not willing to delegate authority to
             the enterprise network over one or more number ranges.
             This node MUST only be included in the capability set if
             the value of the stir-compliance leaf node is set to
             true.  In order to obtain delegate certificates, the
             enterprise network must be made aware of the scope of
             delegation, i.e., the number or number range(s) over
             which the SIP service provider is willing to delegate
             authority.  This information is included in the
             num-ranges container.";
        }
        leaf acme-directory {
          when "../certificate-delegation = 'true'";
          type inet:uri;
          description
            "A node that provides the URL of the directory object for
             delegate certificates using Automatic Certificate
             Management Environment (ACME).  The directory object
             URL, when dereferenced, provides a collection of field
             name-value pairs.  Certain field name-value pairs
             provided in the response are used to bootstrap the
             process of obtaining delegate certificates.
```

```
              This node MUST only be included in the capability
              set if the value of the certificate-delegation leaf node
              is set to true.";
            reference
              "RFC 8555: Automatic Certificate Management Environment
              (ACME)";
          }
        }
      }
      leaf-list extension {
        type iana-sip-option-tags:sip-option-tag;
        description
          "A list of SIP option tags (extensions) supported by the
           service provider network.";
        reference
          "https://www.iana.org/assignments/iana-sip-option-tags/";
      }
    }
  }

<CODE ENDS>
```

7.3. Extending the Capability Set

There are situations in which equipment manufacturers or service providers would benefit from extending the YANG module defined in this document. For example, service providers could extend the YANG module to include information that further simplifies direct IP peering. Such information could include trunk group identifiers, customer/enterprise account numbers, and service provider support numbers, among others. Extensions of the module can be achieved by importing the module defined in this document. An example is provided below.

Consider a new YANG module "example-vendor-config" specified for Vendor's enterprise SBC. The "example-vendor-config" YANG module is configured as follows:

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

  import ietf-sip-auto-peering {
    prefix sipap;
    reference
      "RFC 10006: Automatic SIP Trunking and Peering";
  }

  organization
    "Vendor Enterprise.";
  contact
    "Vendor Enterprise
     1234 Vendor Street
     Anytown, State 12345

     Tel: +1 424 254 5300
```

```
      <mailto:vendor@vendor.com>";
  description
    "Example of a vendor configuration data model that augments the
     IETF SIP auto-peering model to include vendor-specific SBC
     configuration parameters.";

  revision 2026-12-06 {
    description
      "Initial revision of Vendor Enterprise SBC
       configuration data model.";
    reference
      "RFC 10006: Automatic SIP Trunking and Peering";
  }

  augment "/sipap:sip-auto-peering" {
    description
      "Augmentation of the SIP auto-peering model to include vendor-
       specific SBC configuration parameters.";
    container vendorConfig {
      leaf vendorConfigParam1 {
        type int32;
        description
          "Vendor configuration parameter 1
           (SBC Device ID).";
      }
      leaf vendorConfigParam2 {
        type string;
        description
          "Vendor configuration parameter 2
           (SBC Device name).";
      }
      description
        "Container for vendor SBC configuration.";
    }
  }
}
```

In the example above, a custom module named "example-vendor-config" uses the "augment" statement as defined in Section 4.2.8 of [RFC7950] to extend the module defined in this document.

8. Processing the Capability Set Response

This section provides a non-normative description of the procedures that could be carried out by the enterprise network after obtaining the SIP service provider capability set. On obtaining the capability set, the enterprise edge element can parse the various fields within the capability set and generate configuration blocks. Examples of this include the configuration required to successfully register a SIP trunk with the SIP registrar hosted in the service provider network, the configuration required to ensure that fax calls are handled appropriately, and the configuration required to advertise only audio codecs supported by the SIP service provider, among many other configuration blocks. A configuration block generated for an almost identical SIP service provider capability set document is likely going to differ drastically from one vendor to the next.

Enterprise edge elements are usually capable of normalizing mismatches in the signaling and media planes between the enterprise and service provider SIP networks. As a result, most, if not all of the configuration blocks required to enable successful SIP service provider peering might need to be added on the edge element. In situations wherein configuration blocks need to be distributed across multiple devices, some mechanism that is out of scope of this document might be used to communicate the specific fields of capacity set and their corresponding value. Alternatively, a human administrator could go through the capability set document and configure the edge element (and if required, other devices in the enterprise network) appropriately.

9. Examples

This section provides examples of how capability set documents that leverage the YANG module defined in this document can be encoded over JSON as well as the exchange of messages between the enterprise edge element and the service provider to acquire the capability set document. The service provider will create a unique document for each enterprise network that will peer with it.

9.1. JSON Capability Set Document

NOTE: '\' line wrapping per [RFC8792].

```
<CODE BEGINS> file "asap-example.json"

{
    "ietf-sip-auto-peering:sip-auto-peering":
    {
        "variant": "ietf-sip-auto-peering:v1-0",
        "revision": {
            "not-before": "2026-06-15T10:30:00Z",
            "location":
                "https://capserver.example.org/capserver/capdoc.json"
        },
        "transport-info": {
            "transport": [
                "ietf-sip-auto-peering:tcp",
                "ietf-sip-auto-peering:udp"
            ],
            "registrar": [
                {
                    "host": "registrar1.voip.example.com",
                    "port": 5060
                },
                {
                    "host": "registrar2.voip.example.com",
                    "port": 5060
                }
            ],
            "realm": [
                {
                    "name": "voip.example.com",
                    "username": "voip",
```

```
                    "password":
                        "$6$OoEJwExxp6U/FRFq$4RkL2lSSGLoKdfGjX4lQLF\
Xo89gc0wtJsKiBxg/BBz6aNwu7C.D3kRUwD7lvJm6rhaCdhSzVh/XfkkAUY2dTu0"
                }
            ],
            "call-control": [
                {
                    "host": "callServer1.voip.example.com",
                    "port": 5060
                },
                {
                    "host": "192.0.2.40",
                    "port": 5065
                }
            ],
            "dns-server": [
                "192.0.2.50",
                "192.0.2.51"
            ],
            "outbound-proxy": [{
                "host": "192.0.2.35",
                "port": 5060
            }]
        },
        "call-spec": {
            "early-media": true,
            "signaling-forking": false,
            "supported-method": [
                "invite",
                "options",
                "bye",
                "cancel",
                "ack",
                "prack",
                "subscribe",
                "notify",
                "register"
            ],
            "caller-id": {
                "e164-format": true,
                "preferred-method": "from"
            },
            "number-range": [
                {
                    "index": 0,
                    "type": "range",
                    "count": 20,
                    "value": [
                        "19725455000"
                    ]
                },
                {
                    "index": 1,
                    "type": "collection",
                    "count": 2,
                    "value": [
                        "19725455000",
                        "19725455001"
```

```
                    ]
                }
            ]
        },
        "media": {
            "media-type-audio": [
                {
                    "media-format": "ietf-sip-auto-peering:pcmu",
                    "rate": 8000,
                    "ptime": 20
                },
                {
                    "media-format": "ietf-sip-auto-peering:g729",
                    "rate": 8000,
                    "ptime": 20,
                    "parameter": "annexb"
                }
            ],
            "fax": {
                "protocol": [
                    "t38",
                    "pass-through"
                ]
            },
            "rtp": {
                "rtp-trigger": true,
                "symmetric-rtp": true
            },
            "rtcp": {
                "symmetric-rtcp": true,
                "rtcp-feedback": true
            }
        },
        "dtmf": {
            "payload-number": 101,
            "iteration": false
        },
        "security": {
            "signaling": {
                "secure": true,
                "version": ["ietf-tls-common:tls12", "ietf-tls-comm\
on:tls13"]
            },
            "media-security": {
                "key-management": ["sdes", "dtls-srtp"]
            },
            "certificate-location":
                "https://sipserviceprovider.com/certificateList.pem",
            "secure-telephony-identity": {
                "stir-compliance": true,
                "certificate-delegation": true,
                "acme-directory":
                    "https://sipserviceprovider.com/acme.html"
            }
        },
        "extension": [
            "one-hundred-rel",
            "timer",
```

```
            "replaces",
            "path"
        ]
    }
}
<CODE ENDS>
```

9.2. Example Exchange

This section is an informational example depicting the configuration flow that ultimately results in the enterprise edge element obtaining the capability set document from the SIP service provider. Assuming the enterprise edge element has been preconfigured with the request target for the capability set document or has dynamically found the request target, the edge element generates an HTTP GET request. This request can be challenged by the service provider to authenticate the enterprise.

```
GET /capdoc?trunkid=trunkent1456 HTTP/1.1
Host: capserver.ssp1.com
Authorization: Bearer <clientToken>
```

The capability set document is obtained in the body of the response and is encoded in JSON.

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnn

{
    "ietf-sip-auto-peering:sip-auto-peering": ...
}
```

10. IANA Considerations

This document registers two new URIs in the "IETF XML Registry" [RFC3688]. Following the format in [RFC3688], the following registrations have been made.

URI: urn:ietf:params:xml:ns:yang:ietf-sip-auto-peering
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:iana-sip-option-tags
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

This document registers two new YANG modules in the "YANG Module Names" registry [RFC6020].

Name: ietf-sip-auto-peering
Maintained by IANA? N
Namespace: urn:ietf:params:xml:ns:yang:ietf-sip-auto-peering
Prefix: sipap
Reference: RFC 10006

Name: iana-sip-option-tags
Maintained by IANA? Y
Namespace: urn:ietf:params:xml:ns:yang:iana-sip-option-tags
Prefix: sip-option-tags
Reference: RFC 10006

10.1. IANA-Maintained Module for SIP Option Tags

This document defines the initial version of the IANA-maintained "iana-sip-option-tags" YANG module. The most recent version of the YANG module is available in the "YANG Parameters" registry group [YANG-PARAMS].

IANA has added the following to the "Notes" field of the "iana-sip-option-tags" entry in the "YANG Module Names" registry within the "YANG Parameters" registry group:

> New values must not be directly added to the "iana-sip-option-tags" YANG module. They must instead be added to the "Option Tags" registry [SIP-PARAMS].

When a value is added to the "Option Tags" registry, a new "enum" statement must be added to the "iana-sip-option-tags" YANG module. The "enum" statement, and substatements thereof, should be defined:

"enum": Replicates a name from the registry.

"description": Replicates the description from the registry.

"reference": Replicates the reference(s) from the registry with the title of the document(s) added.

Unassigned or reserved values are not present in the module.

When the "iana-sip-option-tags" YANG module is updated, a new "revision" statement with a unique revision date needs to be added in front of the existing "revision" statements. The "revision" statement **MUST** contain both "description" and "reference" substatements as follows.

The "description" substatement captures what changed in the revised version. Typically, the description enumerates the changes such as updates to existing entries (e.g., update a description or a reference) or notes about which "enums" were added or had their status changed (e.g., deprecated, discouraged, or obsoleted).

When such a description is not feasible, the description varies on how the update is triggered.

- If the update is triggered by an RFC, the "description" substatement should include or consist of this text:

  Applied updates as specified by RFC 10006.

- If the update is triggered following another IANA registration policy but not all the values in the registry are covered by the same policy, insert this text (where "Some_IANA_policy" refers to one of the defined registration policies in Section 4 of [RFC8126]):

  Applied updates as specified by the registration policy Some_IANA_policy.

The "reference" substatement points specifically to the published module at [YANG-PARAMS]. It may also point to an authoritative event triggering the update to the YANG module. In all cases, this event is cited from the underlying IANA registry. If the update is triggered by an RFC, that RFC must also be included in the "reference" substatement.

IANA has added this note to the "Option Tags" registry within the "Session Initiation Protocol (SIP) Parameters" registry group [SIP-PARAMS]:

> When this registry is modified, the YANG module "iana-sip-option-tags" <https://www.iana.org/assignments/iana-sip-option-tags> must be updated as defined in RFC 10006.

The service provider will filter out the advertised extensions using local policy.

11. Security Considerations

The capability set document contains sensitive information that must be protected from attackers. A capability set document leak can inflict considerable damage to both the enterprise as well as the service provider. An attacker that gains access to the capability set document can cause problems in multiple ways.

There are multiple attack points in the ASAP workflow. The sections below deal with the different points at which the workflow is vulnerable to attackers.

11.1. OAuth Credentials

In scenarios wherein client authentication is carried out using OAuth resource owner credentials, it is required to ensure that these credentials cannot be acquired by any unauthorized third party. If acquired by an unauthorized third party, these credentials may be used to obtain the capability set document from the SIP service provider and subsequently use the information in such a document to make unauthorized calls while posing as an enterprise telephony network that has legitimately paid for calling services from a SIP service provider.

11.2. Client-Server Communication

All communication used by the edge element to obtain the capability set document from the capability server **MUST** be secured using HTTPS. Failure to do so results in the capability set document being transmitted over clear text, thus exposing sensitive information such as targets for trunks registration, targets for outbound calling requests, and credentials used in building the Authorization header field provided in response to authentication challenges.

11.3. YANG Security Considerations

The "ietf-sip-auto-peering" YANG module defines a data model that a service provider **MUST** adhere to while creating the capability set document, preferably in an automated fashion. The capability set document **SHOULD** be formatted as a JSON file as exhibited in Section 9. The service provider communicates the URL of this JSON file in an out-of-band manner to the enterprise. Alternatively, the enterprise uses WebFinger to discover the URL of the JSON file. The enterprise SBC downloads the JSON file and parses it. Once it has validated that the JSON file is correctly formatted, it applies the configuration and peers with the service provider's network for SIP calls to occur.

It is possible that enterprises may purchase numbers in different countries or regions. In this scenario, there would be multiple SIP trunks between the enterprise and the service provider. The service provider is responsible for creating the capability set documents for each SIP trunk. The capability set document cannot be modified by the enterprise. It can only be created one time by the service provider for each enterprise entering into an agreement with the service provider. Therefore, there are no particularly sensitive writable data nodes.

There are no particularly sensitive writable data nodes.

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:

- registrar: This list contains IP addresses or hostnames belonging to registration servers in the service provider network, which may be targeted by malicious actors.
- realms: This list contains sensitive credentials that are utilized by the enterprise to create a registration with the service provider's network. The registration is a prerequisite to making and receiving calls to and from the service provider, respectively.
- call-control: This list contains IP addresses or hostnames belonging to call processing servers in the service provider network, which may be targeted by malicious actors.
- outbound-proxy: This list contains IP addresses or hostnames belonging to SIP proxies in the service provider network, which may be targeted by malicious actors.
- number-range: This list contains a range of phone numbers allocated by the service provider to an enterprise that the service provider may want to conceal from other enterprises or customers.

There are no particularly sensitive RPC or action operations.

This YANG module uses groupings from other YANG modules that define nodes that may be considered sensitive or vulnerable in network environments. Refer to the Security Considerations of [RFC9911] and [RFC7317] for information as to which nodes may be considered sensitive or vulnerable in network environments.

The YANG module "iana-sip-option-tags" defines a set of types. These nodes are intended to be reused by other YANG modules. This module by itself does not expose any data nodes that are writable, data nodes that contain read-only state, or RPCs. As such, there are no additional security issues related to this YANG module that need to be considered.

12. References

12.1. Normative References

[iana-crypt-hash] IANA, "iana-crypt-hash YANG Module", <https://www.iana.org/assignments/iana-crypt-hash>.

[iana-sip-option-tags] IANA, "iana-sip-option-tags YANG Module", <https://www.iana.org/assignments/iana-sip-option-tags>.

[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>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, July 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <https://www.rfc-editor.org/info/rfc4855>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[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>.

[RFC6665] Roach, A.B., "SIP-Specific Event Notification", RFC 6665, DOI 10.17487/RFC6665, July 2012, <https://www.rfc-editor.org/info/rfc6665>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[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>.

[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>.

[RFC8792] Watsen, K., Auerswald, E., Farrel, A., and Q. Wu, "Handling Long Lines in Content of Internet-Drafts and RFCs", RFC 8792, DOI 10.17487/RFC8792, June 2020, <https://www.rfc-editor.org/info/rfc8792>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

[RFC9645] Watsen, K., "YANG Groupings for TLS Clients and TLS Servers", RFC 9645, DOI 10.17487/RFC9645, October 2024, <https://www.rfc-editor.org/info/rfc9645>.

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

12.2. Informative References

[RFC2833] Schulzrinne, H. and S. Petrack, "RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals", RFC 2833, DOI 10.17487/RFC2833, May 2000, <https://www.rfc-editor.org/info/rfc2833>.

[RFC3262] Rosenberg, J. and H. Schulzrinne, "Reliability of Provisional Responses in Session Initiation Protocol (SIP)", RFC 3262, DOI 10.17487/RFC3262, July 2002, <https://www.rfc-editor.org/info/rfc3262>.

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

[RFC4568] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, DOI 10.17487/RFC4568, July 2006, <https://www.rfc-editor.org/info/rfc4568>.

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <https://www.rfc-editor.org/info/rfc4585>.

[RFC4733] Schulzrinne, H. and T. Taylor, "RTP Payload for DTMF Digits, Telephony Tones, and Telephony Signals", RFC 4733, DOI 10.17487/RFC4733, December 2006, <https://www.rfc-editor.org/info/rfc4733>.

[RFC4961] Wing, D., "Symmetric RTP / RTP Control Protocol (RTCP)", BCP 131, RFC 4961, DOI 10.17487/RFC4961, July 2007, <https://www.rfc-editor.org/info/rfc4961>.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <https://www.rfc-editor.org/info/rfc5764>.

[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>.

[RFC6716] Valin, JM., Vos, K., and T. Terriberry, "Definition of the Opus Audio Codec", RFC 6716, DOI 10.17487/RFC6716, September 2012, <https://www.rfc-editor.org/info/rfc6716>.

[RFC7033] Jones, P., Salgueiro, G., Jones, M., and J. Smarr, "WebFinger", RFC 7033, DOI 10.17487/RFC7033, September 2013, <https://www.rfc-editor.org/info/rfc7033>.

[RFC7092] Kaplan, H. and V. Pascual, "A Taxonomy of Session Initiation Protocol (SIP) Back-to-Back User Agents", RFC 7092, DOI 10.17487/RFC7092, December 2013, <https://www.rfc-editor.org/info/rfc7092>.

[RFC7362] Ivov, E., Kaplan, H., and D. Wing, "Latching: Hosted NAT Traversal (HNT) for Media in Real-Time Communication", RFC 7362, DOI 10.17487/RFC7362, September 2014, <https://www.rfc-editor.org/info/rfc7362>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/info/rfc8555>.

[RFC9114] Bishop, M., Ed., "HTTP/3", RFC 9114, DOI 10.17487/RFC9114, June 2022, <https://www.rfc-editor.org/info/rfc9114>.

[RFC9409] Inamdar, K., Narayanan, S., Engi, D., and G. Salgueiro, "The 'sip-trunking-capability' Link Relation Type", RFC 9409, DOI 10.17487/RFC9409, July 2023, <https://www.rfc-editor.org/info/rfc9409>.

[SIP-PARAMS] IANA, "Session Initiation Protocol (SIP) Parameters", <https://www.iana.org/assignments/sip-parameters>.

[SIPconnect-TR] SIP Forum, "SIPconnect 2.0 Technical Recommendation", <https://www.sipforum.org/download/sipconnect-technical-recommendation-version-2-0/?wpdmdl=2818>.

[YANG-PARAMS] IANA, "YANG Parameters", <https://www.iana.org/assignments/yang-parameters>.

Appendix A. Alternative Mechanisms to Transmit the Capability Set

There are alternative mechanisms that the SIP service provider can use to offload its capability set. For example, the Session Initiation Protocol (SIP) can be extended to define a new event package [RFC6665], such that the enterprise network can establish a SIP subscription with the service provider for its capability set; the SIP service provider can subsequently use the SIP NOTIFY request to communicate its capability set or any state deltas to its baseline capability set.

This mechanism is likely to result in a barrier to adoption for SIP service providers and enterprise networks as equipment manufacturers would have to first add support for such a SIP extension. An HTTP-based approach would be relatively easier to adopt, as most edge devices deployed in enterprise networks today already support HTTP; from the perspective of service provider networks, all that is required is for them to deploy HTTP servers that function as capability servers. Additionally, most SIP service providers require enterprise networks to register with them (using a SIP REGISTER message) before any other SIP methods that initiate subscriptions (SIP SUBSCRIBE) or calls (SIP INVITE) are processed. As a result, a SIP-based framework to obtain a capability set would require operational changes on the part of service provider networks.

Yet another example of an alternative mechanism would be for service providers and enterprise equipment manufacturers to agree on YANG data models [RFC6020] [RFC7950] that enable configuration to be pushed over NETCONF [RFC6241] to enterprise networks from a centralized source hosted in service provider networks. The presence of proprietary software logic for call and media handling in enterprise devices would preclude the generation of a "one-size-fits-all" YANG data model. Additionally, service provider networks pushing configuration to enterprises devices might lead to the loss of implementation autonomy on the part of the enterprise network.

Acknowledgments

We would like to thank those who provided detailed and thoughtful comments on this document, especially Marc Petit-Huguenin, Paul Jones, Ram Mohan R, Nicola Serafini, Jonathan Rosenberg, Jon Peterson, Chris Wendt, and Henning Schulzrinne. Additional thanks to Murray Kucherawy, Joel Halpern, Dan Harkins, Éric Vyncke, Joerg Ott, Mahesh Jethanandani, Orie Steele, Harald Alvestrand, Ebben Aries, Jen Linkova, David Dong, Gorry Fairhurst, Mohamed Boucadair, Paul Wouters, Mike Bishop, Andy Newton, and Amanda Baber for their reviews and feedback.

Authors' Addresses

Kaustubh Inamdar
Unaffiliated
Email: kaustubh.ietf@gmail.com

Sreekanth Narayanan
Unaffiliated
Email: sknth.n@protonmail.com

Cullen Jennings
Cisco Systems
Email: fluffy@iii.ca