RATS Working Group
Internet-Draft
Intended status: Standards Track
Expires: 22 January 2026

H. Birkholz
Fraunhofer SIT
E. Voit
Cisco
W. Pan
Huawei

21 July 2025

Attestation Event Stream Subscription
draft-ietf-rats-network-device-subscription-07

Abstract

This document defines how to subscribe to YANG Event Streams for Remote Attestation Procedures (RATS). In RATS, the Conceptional Messages defined can potentially be subscribed to. Specifically, the YANG module defined in this document augments the YANG module for TPM-based Challenge-Response based Remote Attestation (CHARRA) to allow for subscription to the Conceptual Message type Evidence. Additionally, this document provides the methods and means to define additional Event Streams for other Conceptual Messages than Evidence as illustrated in the RATS Architecture, e.g., Attestation Results, Reference Values, or Endorsements. The module defined requires at least one TPM 1.2, TPM 2.0, or equivalent hardware implementation providing the same protected capabilities as TPMs to be available in the Attester the YANG server is running on.

Table of Contents

1. Introduction

[RFC9683] and [RFC9684] define the operational prerequisites and a YANG Model for the acquisition of Evidence and other Conceptional Messages from a network device containing at least one TPM 1.2 or TPM 2.0 or equivalent hardware implementations that include the protected capabilities as provided by TPMs. However, there are limitations inherent in the challenge-response based conceptual interaction model (CHARRA [I-D.ietf-rats-reference-interaction-models]) upon which these documents are based. One of these limitation is that it is up to a Verifier to request signed Evidence as provided by [RFC9684], from a separate Attester which contains a TPM. The result is that the interval between the occurrence of a security-relevant change event, and the event's visibility within the interested RATS entity, such as a Verifier or a Relying Party, can be unacceptably long. It is common to convey Conceptual Messages ad-hoc or periodically via requests. As new technologies emerge, some of these solutions require Conceptual Messages to be conveyed from one RATS entity to another without the need of continuous polling. Subscription to YANG Notifications [RFC8639] provides a set of standardized tools to facilitate these emerging requirements. This memo specifies a YANG augment to subscribe to YANG modeled remote attestation Evidence as defined in [RFC9684]. Additionally, this memo provides the means to define further Event Streams to convey Conceptional Messages other than Evidence, such as Attestation Results, Endorsements, or Event Logs.

In essence, the limitation of poll-based interactions results in two adverse effects:

1. Conceptual Messages are not streamed to an interested consumer of information, e.g., Verifiers or Relying Parties, as soon as they are generated.

2. If they were to be streamed, Conceptual Messages are not appraisable for their freshness in every scenario. This becomes more important with Conceptional Messages that have a strong dependency on freshness, such as Evidence and corresponding Attestation Results.

This specification addresses the first adverse effect by enabling a consumer of Conceptual Messages (the subscriber) to request a continuous stream of new or updated Conceptual Messages via an [RFC8639] subscription to an <attestation> Event Stream. This new Event Stream is defined in this document and exists upon the producer of Conceptual Messages (the publisher). In the case of a Verifier's subscription to an Attester's Evidence, the Attester will continuously stream a requested set of freshly generated Evidence to

the subscribing Verifier. For example, when a network device's Evidence changes after the occurrence of events, such as booting, updating, control unit fall-over, plugging in or out forwarding units, being attacked, or certificate lifetime change, the network device will generate fresh Evidence available to the subscribing Verifier.

The second adverse effect results from the use of nonces in the challenge-response interaction model [I-D.ietf-rats-reference-interaction-models] realized in [RFC9684]. In [RFC9684], an Attester must wait for a new nonce from a Verifier before it generates a new TPM Quote. To address delays resulting from such a wait, this specification enables freshness to be asserted asynchronously via the streaming attestation interaction model [I-D.ietf-rats-reference-interaction-models]. To convey a RATS Conceptual Message, an initial nonce is provided during the subscription to an Event Stream.

There are several options to refresh a nonce provided by the initial subscription or its freshness characteristics. All of these methods are out-of-band of an established subscription to YANG Notifications. Two complementary methods are taken into account by this document:

1. a central provider supplies new fresh nonces, e.g. via a Handle Provider that distributes Epoch IDs to all entities in a domain as described in [RFC9334] and as facilitated by the Uni-Directional Remote Attestation described in [I-D.ietf-rats-reference-interaction-models] or

2. the freshness characteristics of a received nonce are updated by -- potentially periodic or ad-hoc -- out-of-band TPM Quote requests as facilitated by [RFC9684].

Both approaches to update the freshness characteristics of the Conceptual Messages conveyed via subscription to YANG Notification that are taken into account by this document assume that clock drift between involved entities can occur. In consequence, in some usage scenarios the timing considerations for freshness [RFC9334] might have to be updated in some regular interval. Analogously, there are can be additional methods that are not describe by but nevertheless supported by this document.

This document enables to remove the two adverse effects described by using the YANG augment specified. The YANG augment supports, for example, a RATS Verifier to maintain a continuous appraisal procedure of verifiably fresh Attester Evidence without relying on continuous polling.

2. Terminology

The following terms are imported from [RFC9334]: Attester, Conceptual Message, Evidence, Relying Party, and Verifier. Also imported are the time definitions time(VG), time(NS), time(EG), time(RG), and time(RA) from that document's Appendix A. The following terms are imported from [RFC8639]: Event Stream, Subscription, Publisher, Event Stream Filter, Dynamic Subscription.

2.1. Requirements Notation

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. Operational Model

[RFC9683] describes the conveyance of TPM-based Evidence from a Verifier to an Attester using the CHARRA interaction model [I-D.ietf-rats-reference-interaction-models]. The operational model and corresponding sequence diagram described in this section is based on [RFC9684]. The basis for interoperability required for additional types of Event Streams is covered in Section 6. The following sub-section focuses on subscription to YANG Notifications to the <attestation> Event Stream.

3.1. Sequence Diagram

Figure 1 below is a sequence diagram which updates Figure 5 of [RFC9683]. This sequence diagram adapts [RFC9683] by replacing the TPM-specific challenge-response interaction model with a [RFC8639] Dynamic Subscription to an <attestation> Event Stream. The contents of the <attestation> Event Stream are defined below within Section 4.

Figure 1: YANG Subscription Model for Remote Attestation

* time(VG,RG,RA) are identical to the corresponding time definitions from [RFC9683].

* time(VG',RG',RA') are subsequent instances of the corresponding times from Figure 5 in [RFC9683].

* time(NS) - the subscriber generates a nonce and makes an [RFC8639] <establish-subscription> request based on a nonce. This request also includes the augmentations defined in this document's YANG model. Key subscription RPC parameters include:

  - the nonce,

  - a set of PCRs of interest which the wants to appraise, and

  - an optional filter which can reduce the logged events on the <attestation> stream pushed to the Verifier.

* time(EG) - an initial response of Evidence is returned to the Verifier. This includes:

  - a replay of filtered log entries which have extended into a PCR of interest since boot are sent in the <pcr-extend> notification, and

  - a signed TPM quote that contains at least the PCRs from the <establish-subscription> RPC are included in a <tpm12-attestation> or <tpm20-attestation>). This quote must have included the nonce provided at time(NS).

* time(VG',EG') - this occurs when a PCR is extended subsequent to time(EG). Immediately after the extension, the following information needs to be pushed to the Verifier:

  - any values extended into a PCR of interest,

  - a signed TPM Quote showing the result the PCR extension, and

  - and a handle (see Section 6. in [I-D.ietf-rats-reference-interaction-models], which is either the initially received nonce or a more recently received Epoch ID (see Section 10.3. in [RFC9334] that contains a new nonce or equivalent qualified data.

One way to acquire a new time synchronisation that allows for the reuse of the initially received nonce as a fresh handle is elaborated on in the follow section Section 3.2.

3.2. Continuously Verifying Freshness

As there is no new Verifier nonce provided at time(EG'), it is important to validate the freshness of TPM Quotes which are delivered at that time. The method of doing this verification will vary based on the capabilities of the TPM cryptoprocessor used.

3.2.1. TPM 1.2 Quote

The [RFC8639] notification format includes the <eventTime> object. This can be used to determine the amount of time subsequent to the initial subscription each notification was sent. However this time is not part of the signed results which are returned from the Quote, and therefore is not trustworthy as objects returned in the Quote. Therefore a Verifier MUST periodically issue a new nonce, and receive this nonce within a TPM quote response in order to ensure the freshness of the results. This can be done using the <tpm12-challenge-response-attestation> RPC from [RFC9684].

3.2.2. TPM 2 Quote

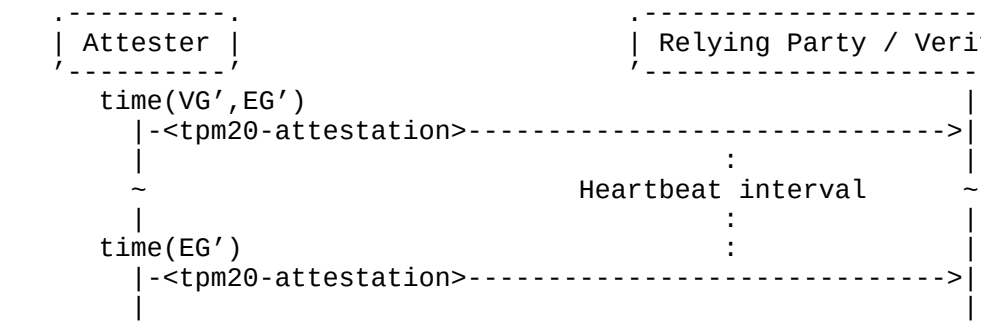
When the Attester includes a TPM2 compliant cryptoprocessor, internal time-related counters are included within the signed TPM Quote. By including a initial nonce in the [RFC8639] subscription request, fresh values for these counters are pushed as part of the first TPM Quote returned to the Verifier. And then as shown by [I-D.birkholz-rats-tuda], subsequent TPM Quotes delivered to the Verifier can the be appraised for freshness based on the predictable incrementing of these time-related countersr.

The relevant internal time-related counters defined within [TPM2.0] can be seen within <tpms-clock-info>. These counters include the <clock>, <reset-counter>, and <restart-counter> objects. The rules for appraising these objects are as follows:

* If the <clock> has incremented for no more than the same duration as both the <eventTime> and the Verifier’s internal time since the initial time(EG) and any previous time(EG’), then the TPM Quote may be considered fresh. Note that [TPM2.0] allows for +/- 15% clock drift. However many chips significantly improve on this maximum drift. If available, chip specific maximum drifts SHOULD be considered during the appraisal process.

* If the <reset-counter>, <restart-counter> has incremented. The existing subscription MUST be terminated, and a new <establish-subscription> SHOULD be generated.

* If a TPM Quote on any subscribed PCR has not been pushed to the Verifier for a duration of an Attester defined heartbeat interval, then a new TPM Quote notification should be sent to the Verifier. This may often be the case, as certain PCRs might be infrequently updated.



4. Remote Attestation Event Stream

The <attestation> Event Stream is an [RFC8639] compliant Event Stream which is defined within this section and within the YANG Module of [RFC9684]. This Event Stream contains YANG notifications which carry Evidence to assists a Verifier in appraising the Trustworthiness Level of an Attester. Data Nodes within Section 4.6 allow the configuration of this Event Stream's contents on an Attester.

This <attestation> Event Stream may only be exposed on Attesters supporting [RFC9683]. As with [RFC9683], it is up to the Verifier to understand which types of cryptoprocessors and keys are acceptable.

4.1. Subscription to the <attestation> Event Stream

To establish a subscription to an Attester in a way which provides provably fresh Evidence, initial randomness must be provided to the Attester. This is done via the augmentation of a <nonce-value> into [RFC8639] the <establish-subscription> RPC. Additionally, a Verifier must ask for PCRs of interest from a platform.

```
augment /sn:establish-subscription/sn:input:
  +---w nonce-value    binary
  +---w pcr-index*     tpm:pcr
```

The result of the subscription will be that passing of the following information:

1. <tpm12-attestation> and <tpm20-attestation> notifications which include the provided <nonce-value>. These attestation notifications MUST at least include all the <pcr-indicies> requested in the RPC.

2. a series of <pcr-extend> notifications which reference the requested PCRs on all TPM based cryptoprocessors on the Attester.

3. <tpm12-attestation> and <tpm20-attestation> notifications generated within a few seconds of the <pcr-extend> notifications. These attestation notifications MUST at least include any PCRs extended.

If the Verifier does not want to see the logged extend operations for all PCRs available from an Attester, an Event Stream Filter should be applied. This filter will remove Evidence from any PCRs which are not interesting to the Verifier.

4.2. Replaying a history of previous TPM extend operations

Unless it is relying on Known Good Values, a Verifier will need to acquire a history of PCR extensions since the Attester has been booted. This history may be requested from the Attester as part of the <establish-subscription> RPC. This request is accomplished by placing a very old <replay-start-time> within the original RPC request. As the very old <replay-start-time> will pre-date the time of Attester boot, a <replay-start-time-revision> will be returned in the <establish-subscription> RPC response, indicating when the Attester booted. Immediately following the response (and before the notifications above) one or more <pcr-extend> notifications which document all extend operations which have occurred for the requested PCRs since boot will be sent. Many extend operations to a single PCR index on a single TPM SHOULD be included within a single notification.

Note that if a Verifier has a partial history of extensions, the <replay-start-time> can be adjusted so that known extensions are not forwarded.

The end of this history replay will be indicated with the [RFC8639] <replay-completed> notification. For more on this sequence, see Section 2.4.2.1 of [RFC8639].

After the <replay-complete> notification is provided, a TPM Quote will be requested and the result passed to the Verifier via a <tpm12-attestation> and <tpm20-attestation> notification. If there have been any additional extend operations which have changed a subscribed PCR value in this quote, these MUST be pushed to the Verifier before the <tpm12-attestation> and <tpm20-attestation> notification.

At this point the Verifier has sufficient Evidence appraise the reported extend operations for each PCR, as well compare the expected value of the PCR value against that signed by the TPM.

4.2.1. TPM2 Heartbeat

For TPM2, make sure that every requested PCR is sent within an <tpm20-attestation> no less frequently than once per heartbeat interval. This MAY be done with a single <tpm20-attestation> notification that includes all requested PCRs every heartbeat interval. This MAY be done with several <tpm20-attestation> notifications at different times during that heartbeat interval.

4.3. YANG notifications placed on the <attestation> Event Stream

4.3.1. pcr-extend

This notification documents when a subscribed PCR is extended within a single TPM cryptoprocessor. It SHOULD be emmitted no less than the <marshalling-period> after an the PCR is first extended. (The reason for the marshalling is that it is quite possible that multiple extensions to the same PCR have been made in quick succession, and these should be reflected in the same notification.) This notification MUST be emmitted prior to a <tpm12-attestation> or <tpm20-attestation> notification which has included and signed the results of any specific PCR extension. If pcr extending events occur during the generation of the <tpm12-attestation> or <tpm20-attestation> notification, the marshalling period MUST be extended so that a new <pcr-extend> is not sent until the corresponding notifications have been sent.

```
   +---n pcr-extend
      +--ro certificate-name     certificate-name-ref
      +--ro pcr-index-changed*   tpm:pcr
      +--ro attested-event* []
         +--ro attested-event
            +--ro extended-with             binary
            +--ro (event-details)?
               +--:(bios-event-log) {tpm:bios}?
               |  +--ro bios-event-entry* [event-number]
               |     +--ro event-number    uint32
               |     +--ro event-type?     uint32
               |     +--ro pcr-index?      pcr
               |     +--ro digest-list* []
               |     |  +--ro hash-algo?   identityref
               |     |  +--ro digest*      binary
               |     +--ro event-size?     uint32
               |     +--ro event-data*     uint8
               +--:(ima-event-log) {tpm:ima}?
               |  +--ro ima-event-entry* [event-number]
               |     +--ro event-number               uint64
               |     +--ro ima-template?              string
               |     +--ro filename-hint?             string
               |     +--ro filedata-hash?             binary
               |     +--ro filedata-hash-algorithm?   string
               |     +--ro template-hash-algorithm?   string
               |     +--ro template-hash?             binary
               |     +--ro pcr-index?                 pcr
               |     +--ro signature?                 binary
               +--:(netequip-boot-event-log) {tpm:netequip_boot}?
                  +--ro boot-event-entry* [event-number]
                     +--ro event-number               uint64
                     +--ro ima-template?              string
                     +--ro filename-hint?             string
                     +--ro filedata-hash?             binary
                     +--ro filedata-hash-algorithm?   string
                     +--ro template-hash-algorithm?   string
                     +--ro template-hash?             binary
                     +--ro pcr-index?                 pcr
                     +--ro signature?                 binary
```

Each <pcr-extend> MUST include one or more values being extended into the PCR. These are passed within the <extended-with> object. For each extension, details of the event SHOULD be provided within the <event-details> object. The format of any included <event-details> is identified by the <event-type>. This document includes two YANG structures which may be inserted into the <event-details>. These two structures are: <ima-event-log> and <bios-event-log>. Implementations wanting to provide additional documentation of a type of PCR extension may choose to define additional YANG structures which can be placed into <event-details>.

4.3.2. tpm12-attestation

This notification contains an instance of a TPM1.2 style signed cryptoprocessor measurement. It is supplemented by Attester information which is not signed. This notification is generated and emitted from an Attester when at least one PCR identified within the subscribed <pcr-indices> has changed from the previous <tpm12-attestation> notification. This notification MUST NOT include the results of any PCR extensions not previously reported by a <pcr-extend>. This notification SHOULD be emitted as soon as a TPM Quote can extract the latest PCR hashed values. This notification MUST be emitted prior to a subsequent <pcr-extend>.

```
    +---n tpm12-attestation {taa:TPM12}?
       +--ro certificate-name       tpm:certificate-name-ref
       +--ro up-time?               uint32
       +--ro TPM_QUOTE2?            binary
       +--ro TPM12-hash-algo?       identityref
       +--ro unsigned-pcr-values* []
          +--ro pcr-index*   tpm:pcr
          +--ro pcr-value*   binary
```

All YANG objects above are defined within [RFC9684]. The <tpm12-attestation> is not replayable.

4.3.3. tpm20-attestation

This notification contains an instance of TPM2 style signed cryptoprocessor measurements. It is supplemented by Attester information which is not signed. This notification is generated at two points in time:

* every time at least one PCR has changed from a previous tpm20-attestation. In this case, the notification SHOULD be emitted within 10 seconds of the corresponding <pcr-extend> being sent:

* after a locally configurable minimum heartbeat period since a previous tpm20-attestation was sent.

```
 +---n tpm20-attestation {taa:TPM20}?
    +--ro certificate-name       tpm:certificate-name-ref
    +--ro TPMS_QUOTE_INFO        binary
    +--ro quote-signature?       binary
    +--ro up-time?               uint32
    +--ro unsigned-pcr-values* []
       +--ro TPM20-hash-algo?   identityref
       +--ro pcr-values* [pcr-index]
          +--ro pcr-index    pcr
          +--ro pcr-value?   binary
```

All YANG objects above are defined within [RFC9684]. The <tpm20-attestation> is not replayable.

4.4. Filtering Evidence at the Attester

It can be useful _not_ to receive all Evidence related to a PCR. An example of this is would be a when a Verifier maintains known good values of a PCR. In this case, it is not necessary to send each extend operation.

To accomplish this reduction, when an RFC8639 <establish-subscription> RPC is sent, a <stream-filter> as per RFC8639, Section 2.2 can be set to discard a <pcr-extend> notification when the <pcr-index-changed> is uninteresting to the verifier.

4.5. Replaying previous PCR Extend events

To verify the value of a PCR, a Verifier must either know that the value is a known good value [KGV] or be able to reconstruct the hash value by viewing all the PCR-Extends since the Attester rebooted. Wherever a hash reconstruction might be needed, the <attestation> Event Stream MUST support the RFC8639 <replay> feature. Through the <replay> feature, it is possible for a Verifier to retrieve and sequentially hash all of the PCR extending events since an Attester booted. And thus, the Verifier has access to all the evidence needed to verify a PCR’s current value.

4.6. Configuring the <attestation> Event Stream

Figure 2 is tree diagram which exposes the operator configurable elements of the <attestation> Event Stream. This allows an Attester to select what information should be available on the stream. A fetch operation also allows an external device such as a Verifier to understand the current configuration of stream.

Almost all YANG objects below are defined via reference from [RFC9684]. There is one object which is new with this model however. <tpm2-heartbeat> defines the maximum amount of time which should pass before a subscriber to the Event Stream should get a <tpm20-attestation> notification from devices which contain a TPM2.

```
  augment /tpm:rats-support-structures:
    +--rw tras:marshalling-period?                  uint8
    +--rw tras:tpm12-subscribed-signature-scheme?
    |   -> ../tpm:attester-supported-algos/tpm12-asymmetric-signing
    |      {taa:TPM12}?
    +--rw tras:tpm20-subscribed-signature-scheme?
    |   -> ../tpm:attester-supported-algos/tpm20-asymmetric-signing
    |      {taa:TPM20}?
    +--rw tras:tpm20-subscription-heartbeat?        uint16
           {taa:TPM20}?

  augment /tpm:rats-support-structures/tpm:tpms:
     +--rw tras:subscription-aik?        tpm:certificate-name-ref
     +--rw (tras:subscribable)?
        +--:(tras:tpm12-stream) {taa:tpm12}?
        |  +--rw tras:tpm12-hash-algo?   identityref
        |  +--rw tras:tpm12-pcr-index*   tpm:pcr
        +--:(tras:tpm20-stream) {taa:tpm20}?
           +--rw tras:tpm20-hash-algo?   identityref
           +--rw tras:tpm20-pcr-index*   tpm:pcr
```

Figure 2: Configuring the \<attestation\> Event Stream

5. YANG Module

This YANG module imports modules from [RFC9684] and [RFC8639]. It is also work-in-progress.

<CODE BEGINS> ietf-tpm-remote-attestation-stream@2025-01-06.yang

```
module ietf-tpm-remote-attestation-stream {
  yang-version 1.1;
  namespace
     "urn:ietf:params:xml:ns:yang:ietf-tpm-remote-attestation-stream";
  prefix tras;

  import ietf-subscribed-notifications {
    prefix sn;
    reference
      "RFC 8639: Subscription to YANG Notifications";
  }
  import ietf-tpm-remote-attestation {
    prefix tpm;
```

```
    reference
      "draft-ietf-rats-yang-tpm-charra";
  }
  import ietf-tcg-algs {
    prefix taa;
  }

  organization "IETF";
  contact
    "WG Web:   <http://tools.ietf.org/wg/rats/>
     WG List:  <mailto:rats@ietf.org>

     Editor:   Eric Voit
               <mailto:evoit@cisco.com>";

  description
    "This module contains YANG specification for subscribing
     to attestation streams which contain events that have
     been generated by TPM chips or equivalent hardware
     implementations that include the protected capabilities
     as provided by TPMs.

     Copyright (c) 2024 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 Simplified
     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).

     This version of this YANG module is part of RFC XXXX
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC
     itself for full legal notices.";

  revision 2024-07-06 {
    description
      "Initial version.";
    reference
      "draft-ietf-rats-network-device-subscription";
  }

  /*
   * IDENTITIES
   */

  identity pcr-unsubscribable {
```

```
      base sn:establish-subscription-error;
      description
        "Requested PCR is unsubscribable by the Attester.";
    }


    /*
     * Groupings
     */

    grouping heartbeat {
      description
        "Allows an Attester to push verifiable, current TPM PCR values
        even when there have been no recent changes to PCRs.";
      leaf tpm20-subscription-heartbeat {
        type uint16;
        units "seconds";
        description
          "Number of seconds before the Attestation stream should send
          a new notification with a fresh quote.  This allows
          confirmation that the PCR values haven't changed since the
          last tpm20-attestation.";
      }
    }


    /*
     * RPCs
     */

    augment "/sn:establish-subscription/sn:input" {
      when 'derived-from-or-self(sn:stream, "attestation")';
      description
        "This augmentation adds a nonce to as a subscription parameters
         that apply specifically to datastore updates to RPC input.";
      uses tpm:nonce;
      leaf-list pcr-index {
        type tpm:pcr;
        min-elements 1;
        description
          "The numbers/indexes of the PCRs. This will act as a filter
          for the subscription so that 'tpm-extend' notifications
          related to non-requested PCRs will not be sent to a
          subscriber.";
      }
    }

    /*
```

```
     * NOTIFICATIONS
     */

    notification pcr-extend {
      description
        "This notification indicates that one or more PCRs have been
        extended within a TPM based cryptoprocessor.  In less than the
        'marshalling-period', it MUST be followed with either a
        corresponding tpm12-attestation or tpm20-attestation
        notification which exposes the result of the PCRs updated.";
      uses tpm:certificate-name-ref;
      leaf-list pcr-index-changed {
        type tpm:pcr;
        min-elements 1;
        description
          "The number of each PCR extended.  This list MUST contain the
          set of PCRs descibed within the event log details.  This leaf
          can be derived from the list of attested events, but exposing
          it here allows for easy filtering of the notifications of
          interest to a verifier.";
      }
      list attested-event {
        description
          "A set of events which extended an Attester PCR.  The
          sequence of elements represented in list must match the
          sequence of events placed into the TPM's PCR.";
        container attested-event {
          description
            "An instance of an event which extended an Attester PCR";
          leaf extended-with {
            type binary;
            mandatory true;
            description
              "Information extending the PCR.";
          }
          choice event-details {
            description
              "Contains the event happened the Attester thought
              was worthy of recording in a PCR.

              choices are of types defined by the identityref
              base tpm:attested_event_log_type";
            case bios-event-log {
              if-feature "tpm:bios";
              description
                "BIOS/UEFI event log format";
              uses tpm:bios-event-log;
            }
```

```
            case ima-event-log {
              if-feature "tpm:ima";
              description
                "IMA event log format";
              uses tpm:ima-event-log;
            }
            case netequip-boot-event-log {
              if-feature "tpm:netequip_boot";
              description
                "IMA event log format";
              uses tpm:network-equipment-boot-event-log;
            }
          }
        }
      }
    }

    notification tpm12-attestation {
      if-feature "taa:tpm12";
      description
        "Contains an instance of TPM1.2 style signed cryptoprocessor
        measurements.  It is supplemented by unsigned Attester
        information.";
      leaf certificate-name {
        type tpm:certificate-name-ref;
        mandatory true;
        description
          "Allows a TPM quote to be associated with a certificate.";
      }
      uses tpm:tpm12-attestation;
      uses tpm:tpm12-hash-algo;
      list unsigned-pcr-values {
        description
          "Allows notifications to be filtered by PCR number or
          PCR value based on via YANG related mechanisms such as PATH.
          This is done without requiring the filtering structure to be
          applied against TCG structured data.";
        leaf-list pcr-index {
          type tpm:pcr;
          min-elements 1;
          description
            "PCR index number.";
        }
        leaf-list pcr-value {
          type binary;
          description
            "PCR value in a sequence which matches to the
            'pcr-index'.";
```

```
      }
    }
  }

  notification tpm20-attestation {
    if-feature "taa:tpm20";
    description
      "Contains an instance of TPM2 style signed cryptoprocessor
      measurements.  It is supplemented by unsigned Attester
      information.";
    leaf certificate-name {
      type tpm:certificate-name-ref;
      mandatory true;
      description
        "Allows a TPM quote to be associated with a certificate.";
    }
    uses tpm:tpm20-attestation {
      description
        "Provides the attestation info.  Also ensures PCRs can be
        XPATH filtered by refining the unsigned data so that it
        appears.";
      refine unsigned-pcr-values {
        min-elements 1;
      }
      refine unsigned-pcr-values/pcr-values {
        min-elements 1;
      }
    }
  }


  /*
   * DATA NODES
   */

  augment "/tpm:rats-support-structures" {
    description
      "Defines platform wide 'attestation' stream subscription
      parameters.";
    leaf marshalling-period {
      type uint8;
      default 5;
      description
        "The maximum number of seconds between the time an event
        extends a PCR, and the 'tpm-extend' notification which
        reports it to a subscribed Verifier.  This period allows
        multiple extend operations bundled together and handled as a
        group.";
```

```
      }
      leaf tpm12-subscribed-signature-scheme {
        if-feature "taa:tpm12";
        type leafref {
          path "../tpm:attester-supported-algos" +
                 "/tpm:tpm12-asymmetric-signing";
        }
        description
          "A single signature-scheme which will be used to sign the
          evidence from a TPM 1.2. which is then placed onto the
          'attestation' event stream.";
      }
      leaf tpm20-subscribed-signature-scheme {
        if-feature "taa:tpm20";
        type leafref {
          path "../tpm:attester-supported-algos" +
                 "/tpm:tpm20-asymmetric-signing";
        }
        description
          "A single signature-scheme which will be used to sign the
          evidence from a TPM 2.0. which is then placed onto the
          'attestation' event stream.";
      }
      uses heartbeat{
        if-feature "taa:tpm20";
      }
    }

    augment "/tpm:rats-support-structures/tpm:tpms" {
      description
        "Allows the configuration 'attestation' stream parameters for a
        TPM.";
      leaf subscription-aik {
        type tpm:certificate-name-ref;
        description
          "Identifies the certificate-name associated with the
          notifications in the 'attestation' stream.";
      }
      choice subscribable {
        config true;
        description
          "Indicates that the set of notifications which comprise the
          'attestation' event stream can be modified or tuned by a
          network administrator.";
        case tpm12-stream {
          if-feature "taa:tpm12";
          description
            "Configuration elements for a TPM1.2 event stream.";
```

```
            uses tpm:tpm12-hash-algo;
            leaf-list tpm12-pcr-index {
              type tpm:pcr;
              description
                "The numbers/indexes of the PCRs which can be
                subscribed.";
            }
          }
          case tpm20-stream {
            if-feature "taa:tpm20";
            description
              "Configuration elements for a TPM2.0 event stream.";
            uses tpm:tpm20-hash-algo;
            leaf-list tpm20-pcr-index {
              type tpm:pcr;
              description
                "The numbers/indexes of the PCRs which can be
                subscribed.";
            }
          }
        }
      }
    }
    <CODE ENDS>
```

6. Event Streams for Conceptual Messages

Analogous to the [RFC8639] compliant <attestation> Event Stream for the conveyance of remote attestation Evidence as defined in Section Section 4, additional Event Streams can be defined for this YANG augment. Additional Event Streams require separate YANG augment specifications that provide the Event Stream definition and optionally a content format definition either via subscriptions to YANG datastores or dedicated YANG Notifications. It is possible to use either YANG subscription methods to other YANG modules for RATS Conceptual Messages or to define Event Streams for other none-YANG-modeled data. In the context of RATS Conceptual Messages, both options MUST be a specified via YANG augments to this specification.

7. Privacy Considerations

The Privacy Considerations of [RFC9683] apply.

8. Security Considerations

The Security Considerations of [RFC9684] and [RFC9683] apply.

9. IANA Considerations

This document registers the following namespace URIs in the [xml-registry] as per [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-tpm-remote-attestation-stream

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

This document registers the following YANG module in the registry [yang-parameters] as per Section 14 of [RFC6020]:

Name: ietf-tpm-remote-attestation-stream

Namespace: urn:ietf:params:xml:ns:yang:ietf-tpm-remote-attestation-stream

Prefix: tras

Reference: draft-ietf-rats-network-device-subscription (RFC form)

10. References

10.1. Normative References

[I-D.ietf-rats-reference-interaction-models]
Birkholz, H., Eckel, M., Pan, W., and E. Voit, "Reference Interaction Models for Remote Attestation Procedures", Work in Progress, Internet-Draft, draft-ietf-rats-reference-interaction-models-14, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-reference-interaction-models-14>.

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://doi.org/10.17487/RFC3688>.

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

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

[RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <https://doi.org/10.17487/RFC8639>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://doi.org/10.17487/RFC9334>.

[RFC9683] Fedorkow, G. C., Ed., Voit, E., and J. Fitzgerald-McKay, "Remote Integrity Verification of Network Devices Containing Trusted Platform Modules", RFC 9683, DOI 10.17487/RFC9683, December 2024, <https://doi.org/10.17487/RFC9683>.

[RFC9684] Birkholz, H., Eckel, M., Bhandari, S., Voit, E., Sulzen, B., Xia, L., Laffey, T., and G. C. Fedorkow, "A YANG Data Model for Challenge-Response-Based Remote Attestation (CHARRA) Procedures Using Trusted Platform Modules (TPMs)", RFC 9684, DOI 10.17487/RFC9684, December 2024, <https://doi.org/10.17487/RFC9684>.

[TPM2.0] TCG, "TPM 2.0 Library Specification", n.d., <https://trustedcomputinggroup.org/resource/tpm-library-specification/>.

10.2. Informative References

[I-D.birkholz-rats-tuda] Fuchs, A., Birkholz, H., McDonald, I., and C. Bormann, "Time-Based Uni-Directional Attestation", Work in Progress, Internet-Draft, draft-birkholz-rats-tuda-07, 10 July 2022, <https://datatracker.ietf.org/doc/html/draft-birkholz-rats-tuda-07>.

[KGV] TCG, "KGV", October 2003, <https://trustedcomputinggroup.org/wp-content/uploads/TCG-NetEq-Attestation-Workflow-Outline_v1r9b_pubrev.pdf>.

[xml-registry] "IETF XML Registry", n.d., <https://www.iana.org/assignments/xml-registry/xml-registry.xhtml>.

[yang-parameters]
"YANG Parameters", n.d.,
<https://www.iana.org/assignments/yang-parameters/yang-parameters.xhtml>.

Appendix A. Change Log

v00-v05

* minor updates as Charra goes through IESG.

Acknowledgements

Thanks to ...

Authors' Addresses

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany
Email: henk.birkholz@ietf.contact

Eric Voit
Cisco Systems, Inc.
Email: evoit@cisco.com

Wei Pan
Huawei Technologies
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu
210012
China
Email: william.panwei@huawei.com