Operations and Management Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: 30 November 2026

D. Lopez
A. Pastor
Telefonica
A. Huang Feng
INSA-Lyon
A. Mendez
Telefonica
H. Birkholz
Fraunhofer SIT
29 May 2026

Applying COSE Signatures for YANG Data Provenance
draft-ietf-opsawg-yang-provenance-05

Abstract

This document defines a mechanism based on CBOR Object Signing and Encryption (COSE) signatures to provide and verify the provenance of YANG data, so it is possible to verify the origin and integrity of a dataset, even when those data are going to be processed and/or applied in workflows where a crypto-enabled data transport directly from the original data source is not available. As the application of evidence-based OAM automation and the use of tools such as AI/ML grow, provenance validation becomes more relevant in all scenarios, in support of the assuring the origin and integrity of data. The use of compact signatures facilitates the inclusion of provenance strings in any YANG schema requiring them.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://dr2lopez.github.io/yang-provenance/draft-ietf-opsawg-yang-provenance.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-opsawg-yang-provenance/.

Discussion of this document takes place on the Operations and Management Area Working Group Working Group mailing list (mailto:opsawg@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/opsawg/. Subscribe at https://www.ietf.org/mailman/listinfo/opsawg/.

Source for this draft and an issue tracker can be found at https://github.com/dr2lopez/yang-provenance.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

OAM automation, generally based on closed-loop principles, requires at least two datasets to be used. Using the common terms in Control Theory, we need those from the plant (the network device or segment under control) and those to be used as reference (the desired values of the relevant data). The usual automation behavior compares these values and takes a decision, by whatever the method (algorithmic, rule-based, an AI model tuned by ML...) to decide on a control action according to this comparison. Assurance of the origin and integrity of these datasets, what we refer in this document as "provenance", becomes essential to guarantee a proper behavior of closed-loop automation.

When datasets are made available as an online data flow, provenance can be assessed by properties of the data transport protocol, as long as some kind of cryptographic protocol is used for source authentication, with TLS, SSH and IPsec as the main examples. But when these datasets are stored, go through some pre-processing or aggregation stages, or even cryptographic data transport is not available, provenance must be assessed by other means.

The original use case for this provenance mechanism is associated with [YANGmanifest], in order to provide a proof of the origin and integrity of the provided metadata, and therefore the examples in this document use the modules described there, but it soon became

clear that it could be extended to any YANG datamodel to support provenance evidence. An analysis of other potential use cases suggested the interest of defining an independent, generally applicable mechanism.

Provenance verification by signatures incorporated in YANG data can be applied to any data processing pipeline, whether they rely on an online flow or use some kind of data store, such as data lakes or time-series databases. The application of recorded data for ML training or validation constitute the most relevant examples of these scenarios.

This document provides a mechanism for including digital signatures within YANG data. It applies COSE [RFC9052] to make the signature compact and reduce the resources required for calculating it. This mechanism is applicable to any serialization of the YANG data supporting a clear method for canonicalization, but this document considers three base ones: CBOR, JSON and XML.

1.1. Target Deployment Scenarios

The provenance mechanisms described in this document are designed to be flexible and applicable in multiple deployment contexts within operational and management practices. The following non-exhaustive list provides examples of intended deployment scenarios:

* Device Configuration Integrity: Digital signatures may be applied to device configuration elements to ensure that specific configuration fragments originate from an authorized source (e.g., controller, automation system) and have not been altered in transit. This is useful for zero-touch provisioning and secure configuration distribution in programmable networks.

* Telemetry and Monitoring Data: When applied to operational state or telemetry data (e.g., YANG-Push updates or Subscription Notifications), provenance signatures can help verify the integrity and authenticity of data collected from network elements, especially when the data may traverse untrusted collection pipelines.

* Network-Wide Service Orchestration: In multi-vendor or multi-domain environments, provenance can be used to track and validate contributions from different orchestrators or domain controllers in composite service models. This enables trustable service chaining and auditability.

2. Conventions and Definitions

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.

The term "data provenance" refers to information describing the origin of a piece of data and, when available, the history of transformations or transfers it has undergone.

The signature mechanisms defined in this document provide integrity protection and origin authentication for YANG data at the time of signing. When applied iteratively by different entities along a data path, these signatures can contribute to building a provenance trail. However, such a trail is only as complete as the set of signatures present and its continuity is not guaranteed by these mechanisms alone.

3. Defining Provenance Elements

The provenance for a given YANG element MUST be convened by a leaf element, containing the COSE signature bitstring built according to the procedure defined below in this section. The provenance leaf MUST be of type provenance-signature, defined as follows:

```
typedef provenance-signature {
     type binary;
     description
      "The provenance-signature type represents a digital signature
       corresponding to the associated YANG element. The signature is based
       on COSE and generated using a canonicalized version of the
       associated element.";
     reference
      "RFC 9052: CBOR Object Signing and Encryption (COSE): Structures and Process
       draft-ietf-opsawg-yang-provenance";
}
```

The use of this type is the proper method for identifying signature leaves, and therefore whenever this type is used for a leaf element, it MUST be considered a provenance signature element, to be generated or verified according to the procedures described in this section.

3.1. Provenance Signature Strings

Provenance signature strings are COSE single signature messages with [nil] payload, according to COSE conventions and registries, and with the following structure (as defined by [RFC9052], Section 4.2):

```
COSE_Sign1 = [
protected /algorithm-identifier, kid, serialization-method/
unprotected /algorithm-parameters/
signature /using as external data the content of the YANG
           (meta-)data without the signature leaf/
]
```

The COSE_Sign1 procedure yields a bitstring when building the signature and expects a bitstring for checking it, hence the proposed type for provenance signature leaves. The structure of the COSE_Sign1 consists of:

* The algorithm-identifier, which MUST follow COSE conventions and registries.

* The kid (Key ID), to be locally agreed, used and interpreted by the signer and the signature validator. URIs [RFC3986] and RFC822-style [RFC5322] identifiers are typical values to be used as kid.

* The serialization-method, a string identifying the YANG serialization in use. It MUST be one of the three possible values "xml" (for XML serialization [RFC7950]), "json" (for JSON serialization [RFC7951]) or "cbor" (for CBOR serialization [RFC9254]).

* The value algorithm-parameters, which MUST follow the COSE conventions for providing relevant parameters to the signing algorithm.

* The signature for the YANG element provenance is being established for, to be produced and verified according to the procedure described below for each one of the enclosing methods for the provenance string described below.

3.2. Signature and Verification Procedures

To keep a concise signature and avoid the need for wrapping YANG constructs in COSE envelopes, the whole signature MUST be built and verified by means of externally supplied data, as defined in [RFC9052], Section 4.3, with a [nil] payload.

The byte strings to be used as input to the signature and verification procedures MUST be built by:

* Selecting the exact YANG content to be used, according to the corresponding enclosing methods.

* Applying the corresponding canonicalization method as described in the following section.

In order to guarantee proper verification, the signature procedure MUST be the last action to be taken before the YANG construct being signed is made available, whatever the means (sent as a reply to a poll or a notification, written to a file or record, etc.), and verification SHOULD take place in advance of any processing by the consuming application. The actions to be taken if the verification fails are specific to the consuming application, but it is RECOMMENDED to at least issue an error warning.

> *Note:* In deployments where YANG data are transported through message broker systems, verification can be applied after message deserialization and before instance data processing, consistently with the placement described in [YANGmsgbroker]. In such scenarios, additional schema validation steps (e.g., YANG schema validation performed at the broker level) may complement the provenance mechanism, further strengthening data integrity before application-level processing. The deployment architecture is out of scope for this document, as the provenance mechanisms defined here are intentionally designed for general-purpose applicability across any YANG data processing system.

3.3. Canonicalization

Signature generation and verification require a canonicalization method to be applied, that depends on the serialization used. According to the three types of serialization defined, the following canonicalization methods MUST be applied:

* For CBOR, length-first core deterministic encoding, as defined by [RFC8949].

* For JSON, JSON Canonicalization Scheme (JCS), as defined by [RFC8785].

* For XML, Exclusive XML Canonicalization 1.0, as defined by [XMLSig].

3.4. Provenance-Signature YANG Module

This module defines a provenance-signature type to be used in other YANG modules.

```
<CODE BEGINS> file "ietf-yang-provenance@2025-05-09.yang"
module ietf-yang-provenance {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-yang-provenance";
  prefix iyangprov;

  organization "IETF OPSAWG (Operations and Management Area Working Group)";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>

     Authors:  Alex Huang Feng
               <mailto:alex.huang-feng@insa-lyon.fr>
               Diego Lopez
               <mailto:diego.r.lopez@telefonica.com>
               Antonio Pastor
               <mailto:antonio.pastorperales@telefonica.com>
               Henk Birkholz
               <mailto:henk.birkholz@sit.fraunhofer.de>";

  description
    "Defines a binary provenance-signature type to be used in other YANG
    modules.

    Copyright (c) 2025 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).

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

  revision 2025-05-09 {
    description
      "First revision";
    reference
      "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
  }
```

```
    typedef provenance-signature {
      type binary;
      description
        "The provenance-signature type represents a digital signature
        corresponding to the associated YANG element. The signature is based
        on COSE and generated using a canonicalized version of the
        associated element.";
      reference
        "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
    }
  }
  <CODE ENDS>
```

4. Enclosing Methods

Once defined the procedures for generating and verifying the provenance signature string, let’s consider how these signatures can be integrated with the associated YANG data by enclosing the signature in the data structure. This document considers four different enclosing methods, suitable for different stages of the YANG schema and usage patterns of the YANG data. The enclosing method defines not only how the provenance signature string is combined with the signed YANG data but also the specific procedure for selecting the specific YANG content to be processed when signing and verifying.

Appendix A includes a set of examples of the different enclosing methods, applied to the same YANG fragment, to illustrate their use.

4.1. Including a Provenance Leaf in a YANG Element

This enclosing method requires a specific element in the YANG schema defining the element to be signed (the enclosing element), and thus implies considering provenance signatures when defining a YANG module, or the use of augmentation to include the provenance signature element in a existing YANG module.

When defining a provenance signature leaf element to appear in a YANG schema by means of this enclosing method, the provenance-signature leaf MAY be defined to be at any position in the enclosing element, but only one such leaf MUST be defined for this enclosing element. If the enclosing element contains other non-leaf elements, they MAY define their own provenance-signature leaf, according to the same rule. In this case, the provenance-signature leaves in the children elements are applicable to the specific child element where they are enclosed, while the provenance-signature leaf enclosed in the top-most element is applicable to the whole element contents, including the children provenance-signature leaf themselves. This allows for

recursive provenance validation, data aggregation, and the application of provenance verification of relevant children elements at different stages of any data processing pipeline.

The specific YANG content to be processed SHALL be generated by taking the whole enclosing element and eliminiating the leaf containing the provenance signature string.

As example, let us consider the two modules proposed in [YANGmanifest]. For the platform-manifest module, the provenance for a platform would be provided by augmenting the current schema with the optional platform-provenance leaf shown below:

```
module: ietf-platform-manifest
  +--ro platforms
     +--ro platform* [id]
       +--ro id                       string
       +--ro name?                    string
       +--ro vendor?                  string
       +--ro vendor-pen?              uint32
       +--ro software-version?        string
       +--ro software-flavor?         string
       +--ro os-version?              string
       +--ro os-type?                 string
       +--ro platform-provenance?     provenance-signature
       +--ro yang-push-streams
       |  +--ro stream* [name]
       |     +--ro name
       |     +--ro description?
       +--ro yang-library
       + . . .
       .
       .
       .
```

For data collections, the provenance of each one would be provided by augmenting the schema with an optional collector-provenance leaf, as shown below:

```
module: ietf-data-collection-manifest
  +--ro data-collections
     +--ro data-collection* [platform-id]
     +--ro platform-id
     |       -> /p-mf:platforms/platform/id
     +--ro collector-provenance?   provenance-signature
     +--ro yang-push-subscriptions
       +--ro subscription* [id]
         +--ro id
         |      sn:subscription-id
         +
         .
         .
         .
     + . . .
     |
     .
     .
     .
```

Note how, in the two examples, the element bearing the provenance signature appears at different positions in the enclosing element. And note that, for processing the element for signature generation and verification, the signature element MUST be eliminated from the enclosing element before applying the corresponding canonicalization method.

Note that, in application of the recursion mechanism described above, a provenance element could be included at the top of any of the collections, supporting the verification of the provenance of the collection itself (as provided by a specific collector), without interfering with the verification of the provenance of each of the collection elements. As an example, in the case of the platform manifests it would look like:

```
module: ietf-platform-manifest
  +--ro platforms
     +--ro platform-collection-provenance? provenance-signature
     +--ro platform* [id]
       +--ro platform-provenance?          provenance-signature
       +--ro id                            string
       +--ro name?                         string
       +--ro vendor?                       string
       + . . .
       .
       .
       .
```

Note here that, to generate the YANG content to be processed in the case of the collection the provenance leafs of the indivual elements SHALL NOT be eliminated, as it SHALL be the case when generating the YANG content to be processed for each individual element in the collection.

4.2. Including a Provenance Signature in YANG-Push Notifications

The signature mechanism proposed in this document MAY be used with YANG-Push [RFC8641] to sign notifications generated directly by publisher nodes. The signature is carried inside the notification envelope header defined in [I-D.ietf-netconf-notif-envelope] as a new extension.

The YANG content to be processed MUST consist of the content defined by the "contents" element in [I-D.ietf-netconf-notif-envelope].

The following sections define the YANG module that augments the "ietf-yp-notification" module. It extends the notification envelope header with a new leaf for the provenance signature and an augmentation to the "ietf-notification-capabilities" to enable clients discover the support of the provenance signature.

4.2.1. YANG Tree Diagram

The following is the YANG tree diagram [RFC8340] for the "ietf-yp-provenance" module.

```
module: ietf-yp-provenance

  augment /sysc:system-capabilities/notc:subscription-capabilities
            /inotenv:notification-metadata/inotenv:metadata:
    +--ro notification-provenance?   boolean

  augment-structure /inotenv:envelope:
    +-- provenance?   iyangprov:provenance-signature
```

And the following is the full YANG tree diagram for the notification structure.

```
module: ietf-notification

  structure envelope:
    +-- event-time         yang:date-and-time
    +-- hostname?          inet:host
    +-- sequence-number?   yang:counter32
    +-- provenance?        iyangprov:provenance-signature
    +-- contents?          <anydata>
```

Unlike the first enclosing method, in this second enclosing method the provenance leaf is added by augmenting a structure (/inotenv:envelope). The provenance leaf is inserted before the contents leaf. This ordering is important because the provenance signature MUST cover the content of the notification but MUST NOT include itself in the signature computation. This ensures the signature remains valid and verifiable. YANG augmented structures typically respect the convention that the anydata node, when present, should appear as the last element in the structure. Therefore, any newly augmented elements are automatically placed before it.

4.2.2. YANG Module

The "ietf-yp-provenance" module augments "ietf-yp-notification" module [I-D.ietf-netconf-notif-envelope] adding the provenance leaf to the notification envelope structure. It also adds the notification-provenance capability to allow clients to discover if provenance signatures are supported.

<CODE BEGINS> file "ietf-yp-provenance@2025-05-09.yang"

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

  import ietf-system-capabilities {
    prefix sysc;
    reference
      "RFC 9196: YANG Modules Describing Capabilities for
       Systems and Datastore Update Notifications";
  }
  import ietf-notification-capabilities {
    prefix notc;
    reference
      "RFC 9196: YANG Modules Describing Capabilities for
       Systems and Datastore Update Notifications";
  }
  import ietf-yang-provenance {
    prefix iyangprov;
    reference
      "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
  }
  import ietf-yang-structure-ext {
    prefix sx;
    reference
      "RFC 8791: YANG Data Structure Extensions";
  }
```

```
  import ietf-yp-notification {
    prefix inotenv;
    reference
      "RFC YYYY: Extensible YANG Model for YANG-Push Notifications";
  }

  organization "IETF OPSAWG (Operations and Management Area Working Group)";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>

     Authors:  Alex Huang Feng
               <mailto:alex.huang-feng@insa-lyon.fr>
               Diego Lopez
               <mailto:diego.r.lopez@telefonica.com>
               Antonio Pastor
               <mailto:antonio.pastorperales@telefonica.com>";

  description
    "Defines a bynary provenance-signature type to be used in other YANG
    modules.

    Copyright (c) 2025 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).

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

  revision 2025-05-09 {
    description
      "First revision";
    reference
      "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
  }

  sx:augment-structure "/inotenv:envelope" {
    leaf provenance {
      type iyangprov:provenance-signature;
      description
        "COSE signature of the content of the Notification for
        provenance verification.";
    }
```

```
    }

    augment "/sysc:system-capabilities"
          + "/notc:subscription-capabilities"
          + "/inotenv:notification-metadata/inotenv:metadata" {
      description
        "Extensions to Notification Capabilities enabling clients to
        know whether the provenance signature is supported.";
      leaf notification-provenance {
        type boolean;
        default "false";
        description
          "Support of the provenance signature on YANG-Push
          Notifications.";
      }
    }
  }
  <CODE ENDS>
```

4.3. Including Provenance as Metadata in YANG Instance Data

Provenance signature strings can be included as part of the metadata in YANG instance data files, as defined in [RFC9195] for data at rest. The augmented YANG tree diagram including the provenance signature is as follows:

```
module: ietf-yang-instance-data-provenance
  augment-structure /id:instance-data-set:
    +-- provenance?   iyangprov:provenance-signature
```

Note: As in the second enclosing method, since this is a data structure, the provenance leaf appears before the content-data element.

The resulting YANG tree structure is:

```
  structure instance-data-set:
    + . . .
    +-- timestamp?           yang:date-and-time
    +-- provenance?          iyangprov:provenance-signature
    +-- content-data?        <anydata>
```

The provenance signature defined in this enclosing method applies to the whole content of the instance-data-set structure. This is independent of any other provenance signature strings that might be present within the content-data itself through other enclosing methods.

The specific YANG content to be processed SHALL be generated by taking the contents of instance-data-set structure, excluding the provenance signature element itself and applying the corresponding canonicalization method.

4.3.1. YANG Module

This module defines the provenance signature element to be included as metadata of a YANG data instance.

```
<CODE BEGINS>
  file "ietf-yang-instance-data-provenance@2025-07-07.yang"
module ietf-yang-instance-data-provenance {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yang-instance-data-provenance";
  prefix "yidprov";
  import ietf-yang-instance-data {
    prefix "id";
    reference
      “RFC 9195 A File Format for YANG Instance Data”
  }
  import ietf-yang-provenance {
    prefix iyangprov;
    reference
      "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
  }
  import ietf-yang-structure-ext {
    prefix sx;
    reference
      "RFC 8791: YANG Data Structure Extensions";
  }
  organization "IETF OPSAWG (Operations and Management Area Working Group)";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
     WG List:  <mailto:opsawg@ietf.org>

     Authors:  Ana Mendez
               <mailto:ana.mendezperz@telefonica.com>
               Diego Lopez
               <mailto:diego.r.lopez@telefonica.com>";
  description
        "Defines a binary provenance-signature type to be used as metadata
         in a YANG data instance.

         Copyright (c) 2025 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).

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

     revision 2025-07-07 {
       description "First revision.";
       reference "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
     }
     sx:augment-structure "/id:instance-data-set" {
       leaf provenance {
         type iyangprov:provenance-signature;
         description
           "Provenance signature that applies to the full content-data block of an instan
ce dataset.This signature can be used to verify the integrity and authenticity of the ins
tance data.";
       }
     }
   }
   <CODE ENDS>
```

4.4. Including Provenance in YANG Annotations

The use of annotations as defined in [RFC7952] seems a natural enclosing method, dealing with the provenance signature string as metadata extension and not requiring modification of existing YANG schemas. The provenance-string annotation is defined as follows:

```
    md:annotation provenance {
          type provenance-signature;
          description
            "This annotation contains a digital signature corresponding
             to the YANG element in which it appears.";
        }
```

The specific YANG content to be processed SHALL be generated by eliminating the provenance annotation (encoded according to what is described in Section 5 of [RFC7952]) from the element it applies to, before invoking the corresponding canonicalization method. In application of the general recursion principle for provenance signature strings, any other provenance strings within the element to which the provenance-string applies SHALL be left as they appear, whatever the enclosing method used for them.

4.4.1. YANG Module

This module defines a metadata annotation to include a provenance signature for a YANG element.

```
<CODE BEGINS> file "ietf-yang-provenance-annotation@2024-06-30.yang"
module ietf-yang-provenance-annotation {
     yang-version 1.1;
     namespace
       "urn:ietf:params:xml:ns:yang:ietf-yang-annotation";
     prefix "ypmd";
     import ietf-yang-metadata {
       prefix "md";
     }
     organization "IETF OPSAWG (Operations and Management Area Working Group)";
     contact
       "WG Web:   <https://datatracker.ietf.org/wg/opsawg/>
        WG List:  <mailto:opsawg@ietf.org>

        Authors: Diego Lopez
                 <mailto:diego.r.lopez@telefonica.com>
                 Alex Huang Feng
                 <mailto:alex.huang-feng@insa-lyon.fr>
                 Antonio Pastor
                 <mailto:antonio.pastorperales@telefonica.com>
                 Henk Birkholz
                 <mailto:henk.birkholz@sit.fraunhofer.de>";
        description
        "Defines a binary provenance-signature type to be used in YANG
         metadata annotations

         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 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).

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

     revision 2024-06-30 {
        description
        "First revision";
        reference
        "RFC XXXX: Applying COSE Signatures for YANG Data Provenance";
```

```
        }
        md:annotation provenance {
          type iyangprov:provenance-signature;
          description
            "This annotation contains the provenance signature for
             the YANG element associated with it";
        }
    }
    <CODE ENDS>
```

5. Security Considerations

The provenance assessment mechanism described in this document relies on COSE [RFC9052] and the deterministic encoding or canonicalization procedures described by [RFC8949], [RFC8785] and [XMLSig]. The security considerations made in these references are fully applicable here.

The considered threat model assumes an attacker with the ability to intercept, observe, copy, replay, or modify YANG data in transit or at rest.

The mechanisms defined here protect against data tampering: any modification of signed YANG data will result in signature verification failure, providing integrity protection from the point of signing onward. Additionally, the signature binds the data to the entity holding the corresponding private key, providing origin authentication for the signed data.

The following threats are explicitly outside the scope of this mechanism:

* If the signing entity’s private key is compromised, an attacker can produce valid signatures; protection against key compromise must be addressed by the key management infrastructure (e.g., PKI, certificate revocation).

* These mechanisms do not guarantee that all intermediate steps in a data path provides a signature: a provenance trail is only as complete as the set of signatures that are present, and gaps in signing by intermediate entities are not detectable by these mechanisms alone.

* A legitimate entity with access to a valid private key may sign incorrect or malicious data; these mechanisms provide no protection against a signing entity that intentionally or unintentionally produces erroneous data.

* Finally, these mechanisms do not inherently guarantee the freshness of signed data; replay of previously signed valid data is not prevented unless additional mechanisms, such as timestamps or nonces bound to the signature context, are employed.

The verification step depends on the association of the kid (Key ID) with the proper public key. This is a local matter for the verifier and its specification is out of the scope of this document. Similarly, key association with reliable data sources is a deployment decision, though a couple of deployment patterns can be considered, depending on the application scenario under consideration. On the one hand, identities may be associated to controller entities (a domain controller, a person in charge of operational aspects, an organizational unit managing an administrative domain, ec.) owning the private keys to be use in generating the provenance signatures for YANG data such as configurations or telemetry. Alternatively, individual devices may hold the identities and corresponding private keys to generate provenance signatures for locally originated data (e.g., telemetry updates). The use of certificates, PKI mechanisms, or any other secure out-of-band distribution of id-public key mappings is RECOMMENDED.

6. IANA Considerations

6.1. IETF XML Registry

This document registers the following URIs in the "IETF XML Registry" [RFC3688]:

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

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

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

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

6.2. YANG Module Name

This document registers the following YANG modules in the "YANG Module Names" registry [RFC6020]:

```
  name: ietf-yang-provenance
  namespace: urn:ietf:params:xml:ns:yang:ietf-yang-provenance
  prefix: iyangprov
  reference: RFC XXXX

  name: ietf-yp-provenance
  namespace: urn:ietf:params:xml:ns:yang:ietf-yp-provenance
  prefix: inotifprov
  reference: RFC XXXX

name: ietf-yang-instance-data-provenance
namespace: urn:ietf:params:xml:ns:yang:ietf-yang-instance-data-provenance
prefix: yidprov
reference: RFC XXXX

  name: ietf-yang-provenance-annotation
  namespace: urn:ietf:params:xml:ns:yang:ietf-yang-annotation-pmd
  prefix: ypmd
  reference: RFC XXXX
```

6.3. YANG SID-file

IANA is requested to register a new ".sid" file in the "IETF YANG-SID Ranges" [RFC9595]:

```
SID range entry point: TBD
SID range size: 20
YANG module name: ietf-yang-provenance
reference: RFC-to-be
```

A ".sid" file is proposed in Appendix B.

7. Implementation Status

An open-source reference implementation, written in Java, is available at https://github.com/tefiros/cose-provenance (https://github.com/tefiros/cose-provenance). This implementation has been used to generate the examples in the appendix of this document, and was first demonstrated at the IETF 122 Hackathon. Work is ongoing to explore its integration with other open-source YANG modules. A Kafka message broker integration was presented at the IETF 123 Hackathon aiming at convergence with current efforts on YANG Push. The implementation is available at https://github.com/tefiros/

kafka-provenance (https://github.com/tefiros/kafka-provenance).

8. References

8.1. Normative References

[I-D.ietf-netconf-notif-envelope]
Feng, A. H., Francois, P., Graf, T., and B. Claise, "Extensible YANG Model for YANG-Push Notifications", Work in Progress, Internet-Draft, draft-ietf-netconf-notif-envelope-05, 18 May 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-notif-envelope-05>.

[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/rfc/rfc2119>.

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

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/rfc/rfc5322>.

[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/rfc/rfc6020>.

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

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

[RFC7952] Lhotka, L., "Defining and Using Metadata with YANG", RFC 7952, DOI 10.17487/RFC7952, August 2016, <https://www.rfc-editor.org/rfc/rfc7952>.

[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/rfc/rfc8174>.

[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/rfc/rfc8340>.

[RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <https://www.rfc-editor.org/rfc/rfc8641>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <https://www.rfc-editor.org/rfc/rfc8785>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/rfc/rfc8949>.

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/rfc/rfc9052>.

[RFC9195] Lengyel, B. and B. Claise, "A File Format for YANG Instance Data", RFC 9195, DOI 10.17487/RFC9195, February 2022, <https://www.rfc-editor.org/rfc/rfc9195>.

[RFC9254] Veillette, M., Ed., Petrov, I., Ed., Pelov, A., Bormann, C., and M. Richardson, "Encoding of Data Modeled with YANG in the Concise Binary Object Representation (CBOR)", RFC 9254, DOI 10.17487/RFC9254, July 2022, <https://www.rfc-editor.org/rfc/rfc9254>.

[RFC9595] Veillette, M., Ed., Pelov, A., Ed., Petrov, I., Ed., Bormann, C., and M. Richardson, "YANG Schema Item iDentifier (YANG SID)", RFC 9595, DOI 10.17487/RFC9595, July 2024, <https://www.rfc-editor.org/rfc/rfc9595>.

[XMLSig] "XML Signature Syntax and Processing Version 2.0", n.d., <https://www.w3.org/TR/xmldsig-core2/>.

8.2. Informative References

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <https://www.rfc-editor.org/rfc/rfc7223>.

[YANGmanifest] Claise, B., Quilbeuf, J., Lopez, D., Martinez-Casanueva, I. D., and T. Graf, "A Data Manifest for Contextualized Telemetry Data", Work in Progress, Internet-Draft, draft-ietf-opsawg-collected-data-manifest-10, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-collected-data-manifest-10>.

[YANGmsgbroker] Graf, T. and A. Elhassany, "An Architecture for YANG-Push to Message Broker Integration", Work in Progress, Internet-Draft, draft-ietf-nmop-yang-message-broker-integration-12, 13 May 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-nmop-yang-message-broker-integration-12>.

Appendix A. Examples of Application of the Different Enclosing Methods

In the examples that follow, the signature strings have been wrapped and, in some cases, indented to improve readability. If these examples are used for any kind of validation, all intermediate carriage returns and whitespace should be deleted to build the actual signature string to be considered.

XML

Let us consider the following YANG instance, corresponding to a monitoring interface statement, as defined in [RFC7223]:

```
<?xml version="1.0" encoding="UTF-8"?>
<interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <interface>
        <name>GigabitEthernet1</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
            ianaift:ethernetCsmacd</type>
        <admin-status>up</admin-status>
        <oper-status>up</oper-status>
        <last-change>2024-02-03T11:22:41.081+00:00</last-change>
        <if-index>1</if-index>
        <phys-address>0c:00:00:37:d6:00</phys-address>
        <speed>1000000000</speed>
        <statistics>
            <discontinuity-time>2024-02-03T11:20:38+00:00</discontinuity-time>
            <in-octets>8157</in-octets>
            <in-unicast-pkts>94</in-unicast-pkts>
            <in-broadcast-pkts>0</in-broadcast-pkts>
            <in-multicast-pkts>0</in-multicast-pkts>
            <in-discards>0</in-discards>
            <in-errors>0</in-errors>
            <in-unknown-protos>0</in-unknown-protos>
            <out-octets>89363</out-octets>
            <out-unicast-pkts>209</out-unicast-pkts>
            <out-broadcast-pkts>0</out-broadcast-pkts>
            <out-multicast-pkts>0</out-multicast-pkts>
            <out-discards>0</out-discards>
            <out-errors>0</out-errors>
        </statistics>
    </interface>
</interfaces>
```

Using the first enclosing method, we will demonstrate how to augment the previous ietf-interfaces YANG module by defining it in the new example module below:

```
module interfaces-provenance-augmented {
  yang-version 1.1;
  namespace "urn:example:interfaces-provenance-augmented";
  prefix ifprov;

  import ietf-interfaces {
    prefix if;
  }
  import ietf-yang-provenance {
    prefix iyangprov;
  }

  description
    "Augments ietf-interfaces with provenance information";

  revision "2025-10-08" {
    description
      "Initial revision of the augment module adding provenance information to ietf-inter
faces.";
  }

  augment "/if:interfaces" {
    leaf interfaces-provenance {
      type iyangprov:provenance-signature;
      description
        "Signature proving provenance of the interface configuration";
    }
  }
}
```

The following tree diagram illustrates the augmentation of the ietf-interfaces module with a provenance-signature at the root container:

```
module: ietf-interfaces
  +--rw interfaces
     +--rw interface* [name]
     |  +--rw name          string
     |  +--rw type          identityref
     |  ...
     +--rw ifprov:interfaces-provenance?  iyangprov:provenance-signature
```

The following example illustrates how a provenance signature can be attached to the root interfaces container to protect the entire set of interface configuration and operational data. This augmentation adds a provenance-signature leaf at the root interfaces container (named "interfaces-provenance" in this case) and produces the following output:

```
<?xml version="1.0" encoding="UTF-8"?>
<interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <interfaces-provenance xmlns="urn:example:interfaces-provenance-augmented">
      0oRRowNjeG1sBGdlYzIua2V5ASag9lhAvzyFP5HP0nONaqTRxKmSqerrDS6C
      QXJSK+5NdprzQZLf0QsHtAi2pxzbuDJDy9kZoy1JTvNaJmMxGTLdm4ktug==
    </interfaces-provenance>
    <interface>
        <name>GigabitEthernet1</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
            ianaift:ethernetCsmacd</type>
        <admin-status>up</admin-status>
        <oper-status>up</oper-status>
        <last-change>2024-02-03T11:22:41.081+00:00</last-change>
        <if-index>1</if-index>
        <phys-address>0c:00:00:37:d6:00</phys-address>
        <speed>1000000000</speed>
        <statistics>
            <discontinuity-time>2024-02-03T11:20:38+00:00</discontinuity-time>
            <in-octets>8157</in-octets>
            <in-unicast-pkts>94</in-unicast-pkts>
            <in-broadcast-pkts>0</in-broadcast-pkts>
            <in-multicast-pkts>0</in-multicast-pkts>
            <in-discards>0</in-discards>
            <in-errors>0</in-errors>
            <in-unknown-protos>0</in-unknown-protos>
            <out-octets>89363</out-octets>
            <out-unicast-pkts>209</out-unicast-pkts>
            <out-broadcast-pkts>0</out-broadcast-pkts>
            <out-multicast-pkts>0</out-multicast-pkts>
            <out-discards>0</out-discards>
            <out-errors>0</out-errors>
        </statistics>
    </interface>
</interfaces>
```

The second enclosing method shows a notification with the provenance signature included inside the notification envelope. The provenance element is placed immediately before the contents element:

```
<?xml version="1.0" encoding="UTF-8"?>
<envelope xmlns="urn:ietf:params:xml:ns:yang:ietf-yp-notification">
    <event-time>2024-02-03T11:37:25.94Z</event-time>
    <provenance xmlns="urn:ietf:params:xml:ns:yang:ietf-yp-provenance">
    0oRRowNjeG1sBGdlYzIua2V5ASag9lhAzyJBpvnpzI/TirrjckAA29q6Qmf
    u56L8ZhUXXhu0KFcKh1qSRFx2wGR/y+xgKigVHYicC7fp/0AlHSXWiKB2sg==
    </provenance>
    <contents>
        <push-update xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
            <id>1011</id>
            <datastore-contents>
                <interfaces-state xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
                    <interface>
                        <name>GigabitEthernet1</name>
                        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
                            ianaift:ethernetCsmacd</type>
                        <admin-status>up</admin-status>
                        <oper-status>up</oper-status>
                        <last-change>2024-02-03T11:22:41.081+00:00</last-change>
                        <if-index>1</if-index>
                        <phys-address>0c:00:00:37:d6:00</phys-address>
                        <speed>1000000000</speed>
                        <statistics>
                            <discontinuity-time>2024-02-03T11:20:38+00:00</discontinuity-
time>
                            <in-octets>8157</in-octets>
                            <in-unicast-pkts>94</in-unicast-pkts>
                            <in-broadcast-pkts>0</in-broadcast-pkts>
                            <in-multicast-pkts>0</in-multicast-pkts>
                            <in-discards>0</in-discards>
                            <in-errors>0</in-errors>
                            <in-unknown-protos>0</in-unknown-protos>
                            <out-octets>89363</out-octets>
                            <out-unicast-pkts>209</out-unicast-pkts>
                            <out-broadcast-pkts>0</out-broadcast-pkts>
                            <out-multicast-pkts>0</out-multicast-pkts>
                            <out-discards>0</out-discards>
                            <out-errors>0</out-errors>
                        </statistics>
                    </interface>
                </interfaces-state>
            </datastore-contents>
        </push-update>
    </contents>
</envelope>
```

The third enclosing method, applicable if the instance is to be stored as YANG instance data at rest, by adding the corresponding metadata, would produce a results as shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<instance-data-set xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-instance-data">
  <name>atRestYANG</name>
  <content-schema></content-schema>
  <revision>
    <date>2024-11-03</date>
    <description>For demos</description>
  </revision>
  <description>Sample for demonstrating provenance signatures</description>
  <contact>diego.r.lopez@telefonica.com</contact>
  <provenance xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-instance-data-provenance">
  0oRRowNjeG1sBGdlYzIua2V5ASag9lhAWff+fMbfNChKUYZ52UTOBmAlYPFe4
  vlZOLyZeW0CU7/2OutDeMCG28+m3rm58jqLjKbcueKLFq8qFJb4mvPY+Q==
  </provenance>
  <content-data>
   <interfaces-state xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <interface>
        <name>GigabitEthernet1</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
            ianaift:ethernetCsmacd</type>
        <admin-status>up</admin-status>
        <oper-status>up</oper-status>
        <last-change>2024-02-03T11:22:41.081+00:00</last-change>
        <if-index>1</if-index>
        <phys-address>0c:00:00:37:d6:00</phys-address>
        <speed>1000000000</speed>
        <statistics>
            <discontinuity-time>2024-02-03T11:20:38+00:00</discontinuity-time>
            <in-octets>8157</in-octets>
            <in-unicast-pkts>94</in-unicast-pkts>
            <in-broadcast-pkts>0</in-broadcast-pkts>
            <in-multicast-pkts>0</in-multicast-pkts>
            <in-discards>0</in-discards>
            <in-errors>0</in-errors>
            <in-unknown-protos>0</in-unknown-protos>
            <out-octets>89363</out-octets>
            <out-unicast-pkts>209</out-unicast-pkts>
            <out-broadcast-pkts>0</out-broadcast-pkts>
            <out-multicast-pkts>0</out-multicast-pkts>
            <out-discards>0</out-discards>
            <out-errors>0</out-errors>
        </statistics>
    </interface>
   </interfaces-state>
 </content-data>
</instance-data-set>
```

Finally, using the fourth enclosing method, the YANG instance would incorporate the corresponding provenance metadata as an annotation, using the namespace prefix specified in the ietf-yang-provenance-annotation module, as introduced in Section 4.4:

```
<?xml version="1.0" encoding="UTF-8"?>
<interfaces-state xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
                  xmlns:ypmd="urn:ietf:params:xml:ns:yang:ietf-yang-annotation-pmd"
                  ypmd:provenance=
                  "0oRRowNjeG1sBGdlYzIua2V5ASag9lhAzen3Bm9AZoyXuetpoTB70SzZqKVxeu
                  OMW099sm+NXSqCfnqBKfXeuqDNEkuEr+E0XiAso986fbAHQCHbAJMOhw==">
    <interface>
        <name>GigabitEthernet1</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
            ianaift:ethernetCsmacd</type>
        <admin-status>up</admin-status>
        <oper-status>up</oper-status>
        <last-change>2024-02-03T11:22:41.081+00:00</last-change>
        <if-index>1</if-index>
        <phys-address>0c:00:00:37:d6:00</phys-address>
        <speed>1000000000</speed>
        <statistics>
            <discontinuity-time>2024-02-03T11:20:38+00:00</discontinuity-time>
            <in-octets>8157</in-octets>
            <in-unicast-pkts>94</in-unicast-pkts>
            <in-broadcast-pkts>0</in-broadcast-pkts>
            <in-multicast-pkts>0</in-multicast-pkts>
            <in-discards>0</in-discards>
            <in-errors>0</in-errors>
            <in-unknown-protos>0</in-unknown-protos>
            <out-octets>89363</out-octets>
            <out-unicast-pkts>209</out-unicast-pkts>
            <out-broadcast-pkts>0</out-broadcast-pkts>
            <out-multicast-pkts>0</out-multicast-pkts>
            <out-discards>0</out-discards>
            <out-errors>0</out-errors>
        </statistics>
    </interface>
</interfaces>
```

JSON

Let us consider the following YANG instance, corresponding to the same monitoring interface statement, with JSON serialization:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "GigabitEthernet1",
        "type": "ianaift:ethernetCsmacd",
        "admin-status": "up",
        "oper-status": "up",
        "last-change": "2024-02-03T11:22:41.081+00:00",
        "if-index": 1,
        "phys-address": "0c:00:00:37:d6:00",
        "speed": 1000000000,
        "statistics": {
          "discontinuity-time": "2024-02-03T11:20:38+00:00",
          "in-octets": 8157,
          "in-unicast-pkts": 94,
          "in-broadcast-pkts": 0,
          "in-multicast-pkts": 0,
          "in-discards": 0,
          "in-errors": 0,
          "in-unknown-protos": 0,
          "out-octets": 89363,
          "out-unicast-pkts": 209,
          "out-broadcast-pkts": 0,
          "out-multicast-pkts": 0,
          "out-discards": 0,
          "out-errors": 0
        }
      }
    ]
  }
}
```

Applying the first enclosing method, a provenance-signature leaf at the top element (named "interfaces-provenance" in this case") would be included following the augmentation module previously defined for the XML example. This will produce the following output:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "GigabitEthernet1",
        "type": "ianaift:ethernetCsmacd",
        "admin-status": "up",
        "oper-status": "up",
        "last-change": "2024-02-03T11:22:41.081+00:00",
        "if-index": 1,
        "phys-address": "0c:00:00:37:d6:00",
        "speed": 1000000000,
        "statistics": {
          "discontinuity-time": "2024-02-03T11:20:38+00:00",
          "in-octets": 8157,
          "in-unicast-pkts": 94,
          "in-broadcast-pkts": 0,
          "in-multicast-pkts": 0,
          "in-discards": 0,
          "in-errors": 0,
          "in-unknown-protos": 0,
          "out-octets": 89363,
          "out-unicast-pkts": 209,
          "out-broadcast-pkts": 0,
          "out-multicast-pkts": 0,
          "out-discards": 0,
          "out-errors": 0
        }
      }
    ],
    "interfaces-provenance-augmented:interfaces-provenance":
      "0oRRowNjeG1sBGdlYzIua2V5ASag9lhAvzyFP5HP0nONaqTRxKmSqerrDS6C
       QXJSK+5NdprzQZLf0QsHtAi2pxzbuDJDy9kZoy1JTvNaJmMxGTLdm4ktug=="
  }
}
```

The second enclosing method would translate into a notification including the "provenance" element as follows:

```
{
  "ietf-yp-notification:envelope" : {
    "event-time" : "2013-12-21T00:01:00Z",
    "contents": {
      "ietf-yang-push:push-update": {
        "id": 1011,
        "datastore-contents": {
          "ietf-interfaces:interfaces-state": {
            "interface": [ {
                "name": "GigabitEthernet1",
                "type": "ianaift:ethernetCsmacd",
                "admin-status": "up",
                "oper-status": "up",
                "last-change": "2024-02-03T11:22:41.081+00:00",
                "if-index": 1,
                "phys-address": "0c:00:00:37:d6:00",
                "speed": 1000000000,
                "statistics": {
                  "discontinuity-time": "2024-02-03T11:20:38+00:00",
                  "in-octets": 8157,
                  "in-unicast-pkts": 94,
                  "in-broadcast-pkts": 0,
                  "in-multicast-pkts": 0,
                  "in-discards": 0,
                  "in-errors": 0,
                  "in-unknown-protos": 0,
                  "out-octets": 89363,
                  "out-unicast-pkts": 209,
                  "out-broadcast-pkts": 0,
                  "out-multicast-pkts": 0,
                  "out-discards": 0,
                  "out-errors": 0
                }
              }
            ]
          }
        }
      }
    },
    "ietf-yp-provenance:provenance":
    "0oRRowNjeG1sBGdlYzIua2V5ASag9lhAiKEKLQKJT12LsNgxt8WllEI65lyi
    E/m12drCfl+wh7T61cTYhFGdEeX8A5F0vmUWROZebq/VVFewUZeVYGZBOQ=="
  }
}
```


The third enclosing method, applicable if the instance is to be stored as YANG instance data at rest, by adding the corresponding metadata, would produce a results as shown below:

```
{
  "ietf-yang-instance-data:instance-data-set" : {
    "name" : "interfaces-labTID-status",
    "contact" : "sofia.garciarincon.practicas@telefonica.com",
    "timestamp" : "Thu Jul 18 11:42:06 CEST 2024",
    "content-data" : {
      "ietf-interfaces:interfaces": {
        "interface": [
          {
            "name": "GigabitEthernet1",
            "iana-if-type:type": "ianaift:ethernetCsmacd",
            "admin-status": "up",
            "oper-status": "up",
            "last-change": "2024-02-03T11:22:41.081+00:00",
            "if-index": 1,
            "phys-address": "0c:00:00:37:d6:00",
            "speed": 1000000000,
            "statistics": {
              "discontinuity-time": "2024-02-03T11:20:38+00:00",
              "in-octets": 8157,
              "in-unicast-pkts": 94,
              "in-broadcast-pkts": 0,
              "in-multicast-pkts": 0,
              "in-discards": 0,
              "in-errors": 0,
              "in-unknown-protos": 0,
              "out-octets": 89363,
              "out-unicast-pkts": 209,
              "out-broadcast-pkts": 0,
              "out-multicast-pkts": 0,
              "out-discards": 0,
              "out-errors": 0
            }
          }
        ]
      }
    },
    "ietf-yang-instance-data-provenance:provenance" :
    "0oRRowNjeG1sBGdlYzIua2V5ASag9lhAmop/c7wMcjRmiSPVy65F/N6O21dsG
    kjGQjIDRizhu3WMwi9Je+VUf5sqwlhSwQCdv5u7mRXa6Pd9dhCwdxdRCA=="
  }
}
```


Finally, using the fourth enclosing method, the YANG instance would incorporate the corresponding provenance metadata as an annotation, using the namespace prefix specified in the yang-provenance-metadata module, as introduced in Section 4.4, and the recommendations in section 5.2.3 of [RFC7952]:

```
{
  "ietf-interfaces:interfaces" : {

    "interface" : [
      {
        "name" : "GigabitEthernet1",
        "iana-if-type:type" : "ianaift:ethernetCsmacd",
        "admin-status" : "up",
        "oper-status" : "up",
        "last-change" : "2024-02-03T11:22:41.081+00:00",
        "if-index" : 1,
        "phys-address" : "0c:00:00:37:d6:00",
        "speed" : 1000000000,
        "statistics" : {
          "discontinuity-time" : "2024-02-03T11:20:38+00:00",
          "in-octets" : 8157,
          "in-unicast-pkts" : 94,
          "in-broadcast-pkts" : 0,
          "in-multicast-pkts" : 0,
          "in-discards" : 0,
          "in-errors" : 0,
          "in-unknown-protos" : 0,
          "out-octets" : 89363,
          "out-unicast-pkts" : 209,
          "out-broadcast-pkts" : 0,
          "out-multicast-pkts" : 0,
          "out-discards" : 0,
          "out-errors" : 0
        }
      }
    ],
    "@": {
        "ypmd:provenance": "0oRRowNjeG1sBGdlYzIua2V5ASag9lhAM/Dx3HVc4GL91jmuU5nWgcmOPPVpA
RLJkWo5wwQYvGFJpKMXTkjAtArPp8v6Sl1ZD1qHimKMhAoHLMHVxBtrcA=="
    }
  }
}
```


CBOR

According to [RFC9254], provenance information MAY be represented in CBOR using either YANG names (CBOR diagnostic notation) or YANG SIDs as map keys. The CBOR diagnostic notation when using name keys would be essentially similar to the JSON encoding presented in the previous section. Both representations are included in the examples below to provide a full reference.

NOTE TO THE RFC EDITOR: The SID values shown below are illustrative and must be replaced by IANA-assigned values before publication.

The first enclosing method will produce the following output, in CBOR diagnostic notation:


```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "GigabitEthernet1",
        "type": "ianaift:ethernetCsmacd",
        "admin-status": "up",
        "oper-status": "up",
        "last-change": "2024-02-03T11:22:41.081+00:00",
        "if-index": 1,
        "phys-address": "0c:00:00:37:d6:00",
        "speed": 1000000000,
        "statistics": {
          "discontinuity-time": "2024-02-03T11:20:38+00:00",
          "in-octets": 8157,
          "in-unicast-pkts": 94,
          "in-broadcast-pkts": 0,
          "in-multicast-pkts": 0,
          "in-discards": 0,
          "in-errors": 0,
          "in-unknown-protos": 0,
          "out-octets": 89363,
          "out-unicast-pkts": 209,
          "out-broadcast-pkts": 0,
          "out-multicast-pkts": 0,
          "out-discards": 0,
          "out-errors": 0
        }
      }
    ],
    "interfaces-provenance-augmented:interfaces-provenance": h'd28451a30363786d6c04676563
322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d681c98e3cf5690112c9916a39c30418bc6149a
4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1d5c41b6b70'
    }
}
```


Note that the provenance leaf in JSON would be represented in BASE64 value and in CBOR diagnostic is a byte string represented with an hexadecimal value.

And the corresponding representation of the first enclosing method using YANG SIDs will be:

```
{
  1500: {                      / ietf-interfaces:interfaces /
    1501: [                     / interface list /
      {
        1502: "GigabitEthernet1",     / name /
        1503: 1800,                   / type identityref /
        1504: 1,                      / admin-status: up /
        1505: 1,                      / oper-status: up /
        1506: "2024-02-03T11:22:41.081+00:00", / last-change /
        1507: 1,                      / if-index /
        1508: "0c:00:00:37:d6:00",        / phys-address /
        1509: 1000000000,             / speed /
        1510: {                       / statistics /
          1511: "2024-02-03T11:20:38+00:00",
          1512: 8157,
          1513: 94,
          1514: 0,
          1515: 0,
          1516: 0,
          1517: 0,
          1518: 0,
          1519: 89363,
          1520: 209,
          1521: 0,
          1522: 0,
          1523: 0,
          1524: 0
        }
      }
    ],
    3162: h'd28451a30363786d6c04676563322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d
681c98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1
d5c41b6b70'   / provenance-signature leaf /
  }
}
```

The following example illustrates the notification-based enclosing method (second method), represented in CBOR diagnostic notation.

```
{
    "ietf-yp-notification:envelope": {
        "event-time": "2013-12-21T00:01:00Z",
        "contents": {
            "ietf-yang-push:push-update": {
                "id": 1011,
                "datastore-contents": {
                    "ietf-interfaces:interfaces-state": {
                        "interface": [
                            {
                                "name": "GigabitEthernet1",
                                "type": "iana-if-type:ethernetCsmacd",
                                "admin-status": "up",
                                "oper-status": "up",
                                "last-change": "2024-02-03T11:22:41.081+00:00",
                                "if-index": 1,
                                "phys-address": "0c:00:00:37:d6:00",
                                "speed": 1000000000,
                                "statistics": {
                                    "discontinuity-time": "2024-02-03T11:20:38+00:00",
                                    "in-octets": 8157,
                                    "in-unicast-pkts": 94,
                                    "in-broadcast-pkts": 0,
                                    "in-multicast-pkts": 0,
                                    "in-discards": 0,
                                    "in-errors": 0,
                                    "in-unknown-protos": 0,
                                    "out-octets": 89363,
                                    "out-unicast-pkts": 209,
                                    "out-broadcast-pkts": 0,
                                    "out-multicast-pkts": 0,
                                    "out-discards": 0,
                                    "out-errors": 0
                                }
                            }
                        ]
                    }
                }
            }
        },
        "ietf-yp-provenance:provenance": h’d28451a30363786d6c04676563322e6b65790126a0f658
4033f0f1dc755ce062fdd639ae5399d681c98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7c
bfa4a5d590f5a878a628c840a072cc1d5c41b6b70’
    }
}
```


And thhis would be the second enclosing method example in YANG SID notation:

```
{
  2957: {                          / ietf-yp-notification:envelope /
    2959: "2024-10-10T08:00:11.22Z", / event-time /
    2958: {                          / contents /
      4000: {                        / push-update /
        4001: 1011,                  / id /
        4002: {                      / datastore-contents /
          1500: {                    / ietf-interfaces:interfaces /
            1501: [                   / interface list /
              {
                1502: "GigabitEthernet1",    / name /
                1503: 1800,                  / type identityref /
                1504: 1,                     / admin-status: up /
                1505: 1,                     / oper-status: up /
                1506: "2024-02-03T11:22:41.081+00:00", / last-change /
                1507: 1,                     / if-index /
                1508: "0c:00:00:37:d6:00",       / phys-address /
                1509: 1000000000,            / speed /
                1510: {                     / statistics /
                  1511: "2024-02-03T11:20:38+00:00", / discontinuity-time /
                  1512: 8157,
                  1513: 94,
                  1514: 0,
                  1515: 0,
                  1516: 0,
                  1517: 0,
                  1518: 0,
                  1519: 89363,
                  1520: 209,
                  1521: 0,
                  1522: 0,
                  1523: 0,
                  1524: 0
                }
              }
            ]
          }
        }
      }
    },
    3162: h’d28451a30363786d6c04676563322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d
681c98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1
d5c41b6b70’  / provenance-signature/
  }
}
```

The following example illustrates the third enclosing method, represented in CBOR diagnostic notation.

```
{
  "ietf-yang-instance-data:instance-data-set": {
    "name": "interfaces-labTID-status",
    "contact": "sofia.garciarincon.practicas@telefonica.com",
    "timestamp": "Thu Jul 18 11:42:06 CEST 2024",
    "content-data": {
      "ietf-interfaces:interfaces": {
        "interface": [
          {
            "name": "GigabitEthernet1",
            "iana-if-type:type": "ianaift:ethernetCsmacd",
            "admin-status": "up",
            "oper-status": "up",
            "last-change": "2024-02-03T11:22:41.081+00:00",
            "if-index": 1,
            "phys-address": "0c:00:00:37:d6:00",
            "speed": 1000000000,
            "statistics": {
              "discontinuity-time": "2024-02-03T11:20:38+00:00",
              "in-octets": 8157,
              "in-unicast-pkts": 94,
              "in-broadcast-pkts": 0,
              "in-multicast-pkts": 0,
              "in-discards": 0,
              "in-errors": 0,
              "in-unknown-protos": 0,
              "out-octets": 89363,
              "out-unicast-pkts": 209,
              "out-broadcast-pkts": 0,
              "out-multicast-pkts": 0,
              "out-discards": 0,
              "out-errors": 0
            }
          }
        ]
      }
    },
    "ietf-yang-instance-data-provenance:provenance":
      h'd28451a30363786d6c04676563322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d681c
98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1d5c4
1b6b70'
  }
}
```


And this would be the third enclosing method using YANG SID notation:

```
{
  3170: {                              / ietf-yang-instance-data:instance-data-set /
    3171: "interfaces-labTID-status", / name /
    3172: "sofia.garciarincon.practicas@telefonica.com", / contact /
    3173: "Thu Jul 18 11:42:06 CEST 2024", / timestamp /
    3174: {                             / content-data /
      1500: {                           / ietf-interfaces:interfaces /
        1501: [                          / interface list /
          {
            1502: "GigabitEthernet1",
            1503: 1800,
            1504: 1,
            1505: 1,
            1506: "2024-02-03T11:22:41.081+00:00",
            1507: 1,
            1508: "0c:00:00:37:d6:00",
            1509: 1000000000,
            1510: {
              1511: "2024-02-03T11:20:38+00:00",
              1512: 8157,
              1513: 94,
              1514: 0,
              1515: 0,
              1516: 0,
              1517: 0,
              1518: 0,
              1519: 89363,
              1520: 209,
              1521: 0,
              1522: 0,
              1523: 0,
              1524: 0
            }
          }
        ]
      }
    },
    3162: h'd28451a30363786d6c04676563322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d
681c98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1
d5c41b6b70'   / provenance-signature /
  }
}
```

Note that there is no IANA registered SID for ietf-yang-instance-data:instance-data-set, so a provisional one is used to complete the example.

Finaly, the following example illustrates the fourth and last enclosing method using CBOR diagonstic notation:

```
{
  "ietf-interfaces:interfaces": {

    "interface": [
      {
        "name": "GigabitEthernet1",
        "iana-if-type:type": "ianaift:ethernetCsmacd",
        "admin-status": "up",
        "oper-status": "up",
        "last-change": "2024-02-03T11:22:41.081+00:00",
        "if-index": 1,
        "phys-address": "0c:00:00:37:d6:00",
        "speed": 1000000000,
        "statistics": {
          "discontinuity-time": "2024-02-03T11:20:38+00:00",
          "in-octets": 8157,
          "in-unicast-pkts": 94,
          "in-broadcast-pkts": 0,
          "in-multicast-pkts": 0,
          "in-discards": 0,
          "in-errors": 0,
          "in-unknown-protos": 0,
          "out-octets": 89363,
          "out-unicast-pkts": 209,
          "out-broadcast-pkts": 0,
          "out-multicast-pkts": 0,
          "out-discards": 0,
          "out-errors": 0
        }
      }
    ],

    "@": {
      "ypmd:provenance":
        h'd28451a30363786d6c04676563322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d68
1c98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1d5
c41b6b70'
    }
  }
}
```

While this last example uses YANG SID notation for the fourth enclosing method:

```
{
  1500: {                              / ietf-interfaces:interfaces /
    1501: [                               / interface list /
      {
        1502: "GigabitEthernet1",
        1503: 1800,
        1504: 1,
        1505: 1,
        1506: "2024-02-03T11:22:41.081+00:00",
        1507: 1,
        1508: "0c:00:00:37:d6:00",
        1509: 1000000000,
        1510: {
          1511: "2024-02-03T11:20:38+00:00",
          1512: 8157,
          1513: 94,
          1514: 0,
          1515: 0,
          1516: 0,
          1517: 0,
          1518: 0,
          1519: 89363,
          1520: 209,
          1521: 0,
          1522: 0,
          1523: 0,
          1524: 0
        }
      }
    ],
    3162: h'd28451a30363786d6c04676563322e6b65790126a0f6584033f0f1dc755ce062fdd639ae5399d
681c98e3cf5690112c9916a39c30418bc6149a4a3174e48c0b40acfa7cbfa4a5d590f5a878a628c840a072cc1
d5c41b6b70'  / provenance-signature/
  }
}
```

In the example above, the provenance-signature leaf is represented using the proposed assigned SID (3162). The representation of this leaf is the same whether it is added via the first enclosing method (as a direct augmentation of the interfaces container) or via the fourth enclosing method (as an annotation using the yp-provenance-metadata module). The only difference between these two methods is the semantic context and location of the leaf within the YANG data: in the first method it is part of the root container structure, while in the fourth method it is included as metadata in the @ annotation object. From the perspective of CBOR representation, SIDs are identical in both methods.

NOTE TO THE RFC EDITOR: Replace the illustrative SID values with the final values allocated by IANA according to [RFC9595].

Appendix B. Provisional YANG SID File (.sid) for ietf-yang-provenance

The following .sid file is provided as a provisional example for implementers. It maps schema nodes defined in the ietf-yang-provenance module to numeric SIDs for use in CBOR compact encoding. These SIDs are provisional and will be replaced by IANA-assigned values upon publication of the RFC.

<CODE BEGINS> file "ietf-yang-provenance@2025-05-09.yang"

```
{
    "ietf-sid-file:sid-file": {
        "module-name": "ietf-yang-provenance",
        "module-revision": "2025-05-09",
        "sid-file-status": "unpublished",
        "description": "Provisional SIDs for ietf-yang-provenance module",
        "reference": "RFC-to-be: Applying COSE Signatures for YANG Data Provenance",
        "dependency-revision": [],
        "assignment-range": [
            {
                "entry-point": "3161",
                "size": "20"
            }
        ],
        "item": [
            {
                "status": "unstable",
                "namespace": "module",
                "identifier": "ietf-yang-provenance",
                "sid": "3161"
            },
            {
                "status": "unstable",
                "namespace": "data",
                "identifier": "/ietf-yang-provenance:provenance-signature",
                "sid": "3162"
            }
        ]
    }
}
```

<CODE ENDS>

Acknowledgments

Thanks to Sofia Garcia (UC3M, sgarciarincon01@gmail.com) for being instrumental in demonstrating the feasibility of the proposed approach, providing a first proof of concept of YANG provenance signatures.

This document is based on work partially funded by the EU Horizon Europe projects iTrust6G (grant 101139198), CYBERNEMO (grant 101168182), cPAID (grant 101168407), MARE (grant 101191436), and 6G-DALI (grant 101192750).

Authors' Addresses

Diego Lopez
Telefonica
Email: diego.r.lopez@telefonica.com

Antonio Pastor
Telefonica
Email: antonio.pastorperales@telefonica.com

Alex Huang Feng
INSA-Lyon
Email: alex.huang-feng@insa-lyon.fr

Ana Mendez
Telefonica
Email: ana.mendezperez@telefonica.com

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany
Email: henk.birkholz@sit.fraunhofer.de