Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 19 December 2025

A. Huang-Feng
P. Francois
INSA-Lyon
T. Graf
Swisscom
B. Claise
Huawei

17 June 2025

Extensible YANG Model for YANG-Push Notifications
draft-ietf-netconf-notif-envelope-02

Abstract

This document defines a new extensible notification structure, defined in YANG, for use in YANG-Push Notification messages enabling any YANG-compatible encodings such as XML, JSON, or CBOR. Additionally, it defines two essential extensions to this structure, the support of a hostname and a sequence number and the support of a timestamp characterizing the moment when the changed data was observed.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 19 December 2025.

Copyright Notice

Table of Contents

1. Introduction

YANG-Push [RFC8639] allows publishers to send notifications to a data collection. The YANG-Push receiver decodes the message and optionally validates the header and the content before forwarding to the next process in the data collection system.

The notification container from YANG-Push is currently based on the XML model from NETCONF Event Notifications [RFC5277]. This model has the drawback that only a single mandatory "eventTime" leaf is defined and does not offer a way to extend this header with new notification metadata. Additionally, this XML model is only valid for XML-based environments. When messages are encoded in other YANG encodings, such as JSON [RFC7951] or CBOR [RFC9254], validators cannot use YANG to validate the message schema.

YANG data consumers receiving notifications require additional notification metadata to understand the full context of the received message. For example, in addition to the timestamp of when the event was encoded, it is also important to know the timestamp when the metrics were observed, the hostname that sourced the message, and have sequence numbers in generated messages so that lost notification messages can be detected in unreliable transports. This additional notification metadata is also helpful in correlating the data with other sources of Network Telemetry [RFC9232] information.

For such reasons, this document proposes the following:

* First, it provides an extensible YANG notification header allowing implementors and IETF contributors to easily add new notification metadata to the notification message.

* Second, it provides the first crucial extensions enabling operators to identify which network node publishes which YANG-Push messages and when the events or metrics were observed on the network node.

* And finally, it provides a way to enable and disable these extensions globally at the server, making the coexistence of different YANG-Push and NETCONF Event Notification [RFC5277] possible.

1.1. Terminology

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

The terms "subscriber", "publisher", and "receiver" are used as defined in [RFC8639].

The term "client" is used as defined in [RFC6241] for NETCONF and [RFC8040] for RESTCONF.

The terms "implementation-time information" and "runtime information" are used as defined in [RFC9196].

In addition, this document defines the following terms:

Notification Metadata: Additional data describing the context of a notification that is sent in each message, e.g. which node generated the messsage or at which time the notification was published.

Notification Envelope: YANG structure encapsulating the payload of a notification, allowing the inclusion of metadata.

2. Relationship to past documents

This section shows the relationship to [RFC5277], [RFC8639], [RFC7951] and [RFC9254].

2.1. Relationship to RFC5277

[RFC5277] defines a mechanism for NETCONF nodes to send notifications to a collector. These are the key relationships between the current document and [RFC5277]:

* This document does not change the header defined by [RFC5277] nor update any behavior defined in [RFC5277]. Implementations of [RFC5277] use the header defined in Section 2.2.1 of [RFC5277].

* The co-existence of the notification model defined in [RFC5277] and the model defined in the current document is possible. The co-existence is discussed in Section 4.

2.2. Relationship to RFC8639

Subscribed Notifications [RFC8639] defines a mechanism on top of [RFC5277] to stream notifications from the NETCONF node. These are the key relationships between the current document and [RFC8639]:

* Section 1.4 of [RFC8639] states that the solution uses the notification header defined in [RFC5277]. This document proposes a new header, which clients can enable and replace the previously [RFC5277] defined. When this new header is used, notification messages are encoded as defined in Section 3.3.

* Section 2.4.2 of [RFC8639] defines how a YANG-Push subscription is defined via a 'establish-subscription' RPC. This document extends the RPCs from Subscribed Notifications [RFC8639] to support enabling the new header defined in this document.

2.3. Relationship to RFC7950

[RFC7950] defines how YANG data is encoded in XML. These are the key relationship points between the current document and [RFC7950]:

* Section 7.16.2 of [RFC7950] defines the XML encoding of YANG notification. This document defines a new header for such notifications. When a YANG-Push publisher implements the specifications in this document with the XML encoding, the notifications are encoded according to Section 3.3.2.1.

2.4. Relationship to RFC7951

[RFC7951] defines how YANG data is encoded using JSON. These are the key relationship points between the current document and [RFC7951]:

* [RFC7951] does not define explicitly how a YANG notification should be encoded using JSON encoding. This document specifies a new header for such notifications. When a YANG-Push publisher implements the specifications in this document with JSON encoding, the notifications are encoded according to Section 3.3.2.2.

2.5. Relationship to RFC9254

[RFC9254] defines how YANG data is encoded using CBOR. These are the key relationship points between the current document and [RFC9254]:

* [RFC9254] does not define explicitly how a YANG notification should be encoded using CBOR encoding. When a YANG-Push publisher implements the specifications in this document in CBOR encoding, the notifications are encoded according to Section 3.3.2.3 in this document.

3. Solution

Section 4.2.10 of [RFC7950] defines the encoding of YANG notifications. A notification is created by defining a 'notification' statement in the YANG module. When a NETCONF server sends this notification, it comprises two parts: a header containing notification metadata that encapsulates the content and the content defined by the 'notification' statement.

In YANG 1.1 [RFC7950], the notification header is based on the model defined in [RFC5277] which contains a single metadata 'eventTime' leaf. An example extracted from [RFC7950] is shown in the following XML:

```
<notification
    xmlns="urn:ietf:params:netconf:capability:notification:1.0">
  <eventTime>2007-09-01T10:00:00Z</eventTime>
  <link-failure xmlns="urn:example:system">
    <if-name>so-1/2/3.0</if-name>
    <if-admin-status>up</if-admin-status>
    <if-oper-status>down</if-oper-status>
  </link-failure>
</notification>
```

This document defines a new notification header and enables extending this header with new notification metadata. The notification header and extensions defined in the following sections are to be used in YANG-Push [RFC8641] environments and can be implemented with NETCONF [RFC6241] and RESTCONF [RFC8040]. Thus, when enabled, this new header replaces the notifications defined in Subscribed Notifications [RFC8639] and YANG-Push [RFC8641] notifications globally for all the entire server.

Section 3.1 defines how a client enables the header defined in this document. Section 3.2 extends the model from [RFC9196] to enable clients to discover the capability of using the new notification header for both implementation-time and runtime information. Lastly, Section 3.3.2 defines the new notification header and how it is encoded using XML, JSON, and CBOR.

3.1. Enabling the Notification Envelope

The notification envelope defined in this document can be enabled during the configuration of a YANG-Push subscription. This document augments the "ietf-subscribed-notification" model [RFC8639] to support the configuration of the "notification-envelope". When enabled, all the notifications defined in Subscribed Notification [RFC8639] and YANG-Push [RFC8641] are encoded as defined in Section 3.3.

```
module: ietf-yp-notification

  augment /sn:subscriptions:
    +--rw enable-notification-envelope?   boolean
    +--rw metadata
```

When the node 'enable-notification-envelope' is set to true, the notifications published by a YANG-Push publisher MUST use the header defined in Section 3.3.1. If any notification metadata is enabled during the subcription configuration, the notification metadata nodes MUST be present in the header. When this node is disabled, notifications are encoded as defined in NETCONF Event Notifications [RFC5277].

When there are existing subscriptions and a client changes the node 'enable-notification-envelope', all existing subscriptions MUST be terminated. The publisher MUST send a 'subscription-terminated' notification to all the existing subscriptions using the header configured before the change. Any new subscription after the change use the header defined by the node 'enable-notification-envelope', i.e. encoded as Section 3.3.1 when enabled and as defined in [RFC5277] if disabled.

3.2. Discovering the support of this model

A client can discover the support of 'notification-envelope' model through the capabilities model defined in [RFC9196]. This documents extends the 'ietf-notification-capabilities' model with:

* A container containing a leaf 'notification-envelope', stating that the YANG notification can be encoded following the notification-envelope model.

* A container 'metadata' containing all the supported extensions to this header. Extensions are defined in Section 3.4.

The YANG models defined in Section 5 augments the 'ietf-notification-capabilities' model with the leaf and container listed above:

```
  augment /sysc:system-capabilities/notc:subscription-capabilities:
    +--ro notification-metadata
       +--ro notification-envelope?   boolean
       +--ro metadata
```

This model can be retrieved via a NETCONF <get> RPC.

3.3. Notification Envelope

This section defines how YANG notifications are structured when the notification envelope is enabled on YANG-Push subscriptions. The following sections define how this model is encoded in XML, JSON and CBOR.

3.3.1. Base Notification Model

When a YANG-Push publisher uses the notification model defined in this document, the notification is structured as follows:

* The notification is encapsulated in a root "envelope" container.

* The header of the notification contains the notification metadata that is enabled during the configuration of the subscription as a child nodes of the root "notification-envelope" container.

* The content of the notification defined by the ’notification’ statement is encoded in the ’contents’ leaf.

* The ’contents’ element MUST be located at the end of the notification envelop structure.

The following YANG tree [RFC8340] illustrates the notification envelope supporting only the mandatory metadata ’event-time’. See Section 3.4 for more extensions to this header.

```
  structure envelope:
    +-- event-time         yang:date-and-time
    +-- contents?          <anydata>
```

3.3.2. Encoding of the Notification Model

The YANG notification can be encoded using XML [W3C.REC-xml-20001006][RFC7951], JSON [RFC7951] and CBOR [RFC9254].

3.3.2.1. XML encoding

A YANG notification encoded in XML is structured as a root "envelope" container. The namespace of this container is the namespace defined in the YANG module "ietf-yp-notification":

```
urn:ietf:params:xml:ns:yang:ietf-yp-notification
```

Two mandatory child nodes within the "envelope" container are expected, representing the event time and the notification payload. The "event-time" node is defined within the same XML namespace as the "envelope" container. The "event-time" node MUST be compliant with [RFC3339]. Other notification metadata within the YANG module defined in Section 5.1.2 MUST use the same XML namespace. See Section 3.4 for more details.

When other notification metadata is enabled through configuration, the supplementary nodes are encoded at the same level as the mandatory "event-time" node and use the XML namespace defined in the YANG module.

The content of the notification that is defined by the ’notification’ statement is encoded in the "contents" node. The name and namespace of this payload element are determined by the YANG module containing the ’notification’ statement representing the notification message.

The following example shows a "push-update" notification defined in the YANG module of YANG-Push [RFC8641] encoded in XML:

```
<envelope xmlns="urn:ietf:params:xml:ns:yang:ietf-yp-notification">
  <event-time>2024-10-10T10:59:55.32Z</event-time>
  <contents>
    <push-update xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
      <id>1011</id>
      <datastore-contents>
        <interfaces
          xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
          <interface>
            <name>eth0</name>
            <type>iana-if-type:ethernetCsmacd</type>
            <if-index>1</if-index>
            <admin-status>down</admin-status>
            <oper-status>down</oper-status>
            <statistics>
              <discontinuity-time>
                2013-04-01T03:00:00+00:00
              </discontinuity-time>
            </statistics>
         </interface>
        </interfaces>
      </datastore-contents>
    </push-update>
  </contents>
</envelope>
```

Figure 1: XML-encoded notification

3.3.2.2. JSON Encoding

A YANG notification encoded in JSON is structured as a root "envelope" container. The namespace of this container is the name of the YANG module "ietf-yp-notification" defined in Section 5.1.2.

Two mandatory child nodes within the "ietf-notification:envelope" container are expected, representing the event time and the notification payload. The "event-time" node is defined within the same namespace as the "ietf-yp-notification:envelope" container and is compliant with [RFC3339]. Other metadata specified within the YANG module defined in Section 5.1.2 MUST use the same namespace "ietf-yp-notification".

When other notification metadata is enabled through configuration, the supplementary nodes are encoded at the same level as the mandatory ’event-time’ node and use the YANG module name as its namespace. See Section 3.4 for more details.

The content of the notification that is defined by the 'notification' statement is encoded in the "contents" node. The name and namespace of this payload element are determined by the YANG module containing the 'notification' statement representing the notification message.

The following example shows a "push-update" notification defined in the YANG module of YANG-Push [RFC8641] encoded in JSON:

```
{
  "ietf-yp-notification:envelope": {
    "event-time": "2024-10-10T08:00:11.22Z",
    "contents": {
      "ietf-yang-push:push-update": {
        "id": 1011,
        "datastore-contents": {
          "ietf-interfaces:interfaces": {
            "interface": [
              {
                "name": "eth0",
                "type": "iana-if-type:ethernetCsmacd",
                "if-index": 1,
                "admin-status": "up",
                "oper-status": "up",
                "statistics": {
                  "discontinuity-time": "2025-03-25T06:43:12Z"
                }
              }
            ]
          }
        }
      }
    }
  }
}
```

Figure 2: JSON-encoded notification

3.3.2.3. CBOR Encoding

YANG notifications can be encoded in CBOR using Names or SIDs in keys.

Notifications encoded using names is similar to JSON encoding as defined in Section 3.3 of [RFC9254]. The key of the element can be the name of the element itself or be namespace-qualified. In the latter case, the namespace of the notification container uses the YANG module name "ietf-yp-notification", defined in Section 5.1.2.

Notification encoded using YANG-SIDs replaces the names of the keys of the CBOR encoded message for a 63-bit unsigned integer. In this case, the keys of the encoded data use the SID value as defined in Section 3.2 of [RFC9254]. A SID allocation process is needed beforehand as defined in [RFC9595].

In the notification, two mandatory child nodes within the "ietf-yp-notification:envelope" container are expected, representing the event time and the notification payload. The "event-time" node is defined within the same namespace as the "ietf-yp-notification:envelope" container and is compliant with [RFC3339].

When other notification metadata is enabled through configuration, the supplementary nodes are encoded at the same level as the mandatory "event-time" node and use the YANG module name as its namespace when they are encoded as names. When encoded using YANG SIDs, a SID value assigned to the metadata node is used. See Section 3.4 for more details.

The content of the notification that is defined by the ’notification’ statement is encoded in the "contents" node. The name and namespace of this payload element are determined by the YANG module containing the ’notification’ statement representing the notification message. Similarly, SIDs can be used as keys if they are well allocated.

Figure 3 shows a "push-update" notification defined in the YANG module of YANG-Push [RFC8641] encoded in CBOR using names as keys. The example uses the CBOR diagnostic notation as defined in section 3.1 of [RFC9254]:

```
{
  "ietf-yp-notification:envelope": {
    "event-time": "2024-10-10T08:00:11.22Z",
    "contents": {
      "ietf-yang-push:push-update": {
        "id": 1011,
        "datastore-contents": {
          "ietf-interfaces:interfaces": {
            "interface": [
              {
                "name": "eth0",
                "type": "iana-if-type:ethernetCsmacd",
                "if-index": 1,
                "admin-status": "up",
                "oper-status": "up",
                "statistics": {
                  "discontinuity-time": "2025-03-25T06:43:12Z"
                }
              }
            ]
          }
        }
      }
    }
  }
}
```

Figure 3: CBOR-encoded notification using diagnostic notation

And Figure 4 shows the same notification encoded using SIDs:

```
{
  2958: {
    2: "2024-10-10T08:00:11.22Z",
    1: {
      "ietf-yang-push:push-update": {
        "id": 1011,
        "datastore-contents": {
          "ietf-interfaces:interfaces": {
            "interface": [
              {
                "name": "eth0",
                "type": "iana-if-type:ethernetCsmacd",
                "if-index": 1,
                "admin-status": "up",
                "oper-status": "up",
                "statistics": {
                  "discontinuity-time": "2025-03-25T06:43:12Z"
                }
              }
            ]
          }
        }
      }
    }
  }
}
```

Figure 4: CBOR-encoded notification using YANG SIDs in CBOR diagnostic notation

3.4. Extensions for the Notification Envelope

This section describes two envelope header extensions. When the envelope is enabled via the "enable-notification-envelope" node, the publisher includes by default the "hostname" and "sequence-number" defined in Section 3.4.1. The client discovers the support of these two extensions with the mechanism defined in Section 3.2. When the extensions defined in this document are supported by the server, the client discovers the presence of new metadata with the following augmentations in the 'ietf-notification-capabilities':

```
module: ietf-yp-notification

  augment /sysc:system-capabilities/notc:subscription-capabilities:
    +--ro notification-metadata
       +--ro notification-envelope?   boolean
       +--ro metadata
          +--ro hostname-sequence-number?   boolean

module: ietf-yp-observation

  augment /sysc:system-capabilities/notc:subscription-capabilities:
    +--ro yang-push-observation-supported?   boolean
            {yang-push-observation-timestamp}?
```

This document defines the following notification metadata as shown in the following YANG tree [RFC8340]. It also defines an extension to the YANG-Push header. See the following sections for more details.

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

3.4.1. Support of Hostname and Sequencing

When YANG-Push notification messages are forwarded from a receiver to another system, such as a message broker or a time series database, the transport context is lost since it is not part of the notification metadata of the notification container. Therefore, the downstream system is unable to associate the message with the publishing process (the exporting network node), nor able to detect message loss or reordering.

To correlate network data among different Network Telemetry planes as described in Section 3.1 of [RFC9232] or among different YANG-Push subscription types as defined in Section 3.1 of [RFC8641], a reference to the node streaming the data is needed. This is essential for understanding the timely relationship among these different planes and YANG-Push subscription types.

Today, network operators work around this impediment by preserving the transport source IP address and sequence numbers of the publishing process. However, this implies encoding this information in the YANG-Push notification messages which impact the semantic readability of the message in the downstream system.

On top of that, the transport source IP address might not represent the management IP address by which the YANG-Push publisher should be known. In other terms, the "source-host" [RFC6470], which is the "Address of the remote host for the session" might not be the management IP address.

To overcome these issues, this document proposes a notification container extension with a hostname and a sequence number. This allows the downstream system to not only be able to identify from which network node, subscription, and time the message was published but also, the order of the published messages.

hostname: Describes the node’s hostname according to the ’sysName’ object definition in [RFC1213] from where the message was published from. This value is usually configured on the node by the administrator to identify the node in the network uniquely.

sequence-number: Generates a unique sequence number for each published message by the publisher process. The number counts up at every published notification message as described in [RFC9187].

Figure 5 provides an example of a JSON encoded, [RFC8259], "push-update" notification message with hostname and sequence-number as extension.

```
{
    "ietf-yp-notification:envelope": {
    "event-time": "2023-03-25T08:30:11.22Z",
    "hostname": "example-router",
    "sequence-number": 1,
    "contents": {
      "ietf-yang-push:push-update": {
        "id": 6666,
        "datastore-contents": {
          "ietf-interfaces:interfaces": {
            "interface": [
              {
                "name": "eth0",
                "type": "iana-if-type:ethernetCsmacd",
                "if-index": 1,
                "admin-status": "up",
                "oper-status": "up",
                "statistics": {
                  "discontinuity-time": "2025-03-25T06:43:12Z"
                }
              }
            ]
          }
        }
      }
    }
  }
```

Figure 5: JSON Example for a ’push-update’ notification message

3.5. Extensions for the YANG-Push Header

This section described two optional YANG ’push-update’ and ’push-change-update’ notification header extensions which are enabled by default. The client discovers the support of these two extension headers with the mechanism defined in Section 3.2.

This document defines the following notification metadata as shown in the following YANG tree [RFC8340]. See the following sections for more details.

```
module: ietf-yp-observation

  augment /yp:push-update:
    +--ro timestamp?       yang:date-and-time
    +--ro point-in-time?   enumeration
  augment /yp:push-change-update:
    +--ro timestamp?       yang:date-and-time
    +--ro point-in-time?   enumeration
```

3.5.1. Support of Observation Timestamp

To correlate network data among different Network Telemetry planes, as described in Section 3.1 of [RFC9232], or among different YANG-Push subscription types, as defined in Section 3.1 of [RFC8641], a receiver needs a timestamp reference to align all the metrics and events. The observation timestamp defined in this document characterizes the moment the state change was observed or the moment when the data was measured, so that a receiver can correctly align the collected data.

The delay between the YANG-Push export process and the reception of the message at the receiver instance can be measured using the node 'event-time' defined in Section 3.3.1. However, as the 'event-time' node only establishes the moment when the YANG-Push message was crafted and sent, the moment when such metrics were collected or the state changes were observed cannot be measured using this timestamp. The observation timestamp defined in this section characterizes the moment when the metrics were observed, which enable aligning the received metrics to the actual moment they were measured.

When the time bucket length in a time series database and the periodical YANG-Push subscription time are configured with the same values, the 'event-time' of the NETCONF notification message header can be used for indexing the data in the time series database. There is a variable delay between the observation timestamp, the 'event-time', and the "anchor-time" as described in Section 4.2 of [RFC8641]. When these timestamps are close to the time bucket boundaries, a time bucket may experience data collection discrepancies, e.g. 0 measurements are aggregated into one time bucket while the next time bucket contains 2 measurements. This leads to inconsistent accounting errors in the time series database. This problem is resolved using the observation timestamp instead of the 'event-time' for time series database indexation.

By extending YANG-Push Notifications with the observation timestamp and a 'point-in-time' node, the data collection process can always ensure it has the best available time for indexing the data. It can therefore use unconditionally the observation timestamp node in the

data processing chain to correctly align the metrics and events. At the same time, the 'point-in-time' node ensures that the semantics associated with the timestamp are not lost throughout the data collection chain.

Besides the Subscription ID as described in Section 3.7 of [RFC8641], the following network observation time metadata objects are part of "push-update" and "push-change-update" notifications.

timestamp: States the measurement observation time for the "push-update" notification in "periodical" subscriptions and for the "push-change-update" notification in "on-change" subscriptions.

By comparing the observation timestamp of two "push-update" notifications in a periodical subscription, the collector can deduce the actual cadence of the measurements, and compare it with the subscription configuration. In case of an "on-change" subscription it states the time when the network state change was observed.

point-in-time: The enumeration states at which point in time the value of the observation timestamp was observed. Choices are:

'current-accounting' states the point in time where the metrics are polled and observed in "periodical" subscriptions.

'initial-state' states the initial point in time when the subscription was established and the state was observed for "on-change sync on start" subscriptions.

'state-changed' states the point in time when the state change was observed after the subscription was established for "on-change" and "on-change sync on start" subscriptions.

3.5.1.1. Usage Example

This section illustrates the usage of the "point-in-time" node in two different subscriptions. Section 3.5.1.1.1 showcases a YANG-Push subscription monitoring the state of an interface using an 'on-change sync on start' subscription. Section 3.5.1.1.2 illustrates the usage of the 'point-in-time' node within periodical subscriptions.

3.5.1.1.1. On-Change Subscriptions

Figure 6 illustrates the set of events that lead to the generation of 'on-change' YANG-Push notifications. This timeline depicts the following states and events:

* T1: At first, the operational state of the interface is "Up".

* T2: After configuring an ’on-change’ subscription supporting ’sync on start’, the publisher sends the initial state of the interface.

* T3: This is the moment the interface changes its operational status to "Down".

* T4: After the interface state changes, the publisher generates the ’on-change’ notification alerting the receiver.

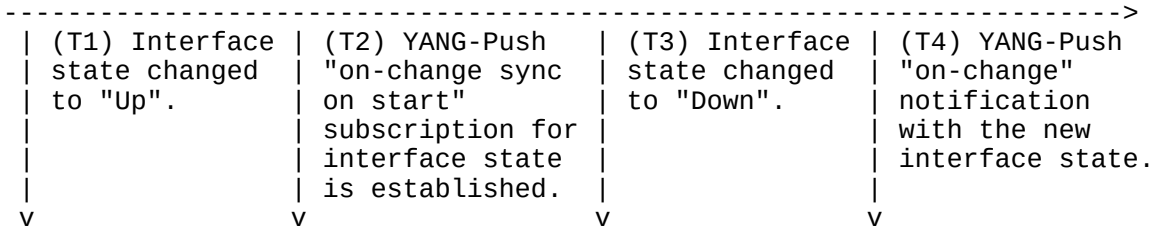

Figure 6: Example timeline for On-Change Sync on Start Subscription

At T2, after configuring the subscription, the publisher triggers a ’push-update’ notification as depicted in Figure 7. The ’event-time’ is the moment the YANG-Push process triggered the notification while the ’ietf-yp-observation:timestamp’ node characterizes the timestamp T1, the latest moment this state was observed. In this case, the ’point-in-time’ node is set to ’initial-state’.

```
{
  "ietf-yp-notification:envelope": {
    "event-time": "2025-03-25T08:30:11.22Z",
    "hostname": "example-router",
    "sequence-number": 1,
    "contents": {
      "ietf-yang-push:push-update": {
        "id": 6666,
        "ietf-yp-observation:timestamp": "2025-03-25T08:29:30.22Z",
        "ietf-yp-observation:point-in-time": "initial-state",
        "datastore-contents": {
          "ietf-interfaces:interfaces": {
            "interface": [{
              "name": "eth0",
              "type": "iana-if-type:ethernetCsmacd",
              "if-index": 1,
              "admin-status": "up",
              "oper-status": "up",
              "statistics": {
                "discontinuity-time": "2025-03-25T06:43:12Z"
              }
            }]
          }
        }
      }
    }
  }
}
```


Figure 7: Example of 'push-update' notification sent after the subscription is established.

After T3, the publisher triggers a 'push-change-update' notification announcing an interface status change to the receiver as depicted in Figure 8. In this case, the 'ietf-yp-observation:timestamp' node characterizes the moment the interface changed its status at T3 while the value of the 'event-time' node characterizes the moment the YANG-Push process generated the message at T4.

```
{
  "ietf-yp-notification:envelope": {
    "event-time": "2025-03-25T08:35:12.22Z",
    "hostname": "example-router",
    "sequence-number": 1,
    "contents": {
      "ietf-yang-push:push-change-update": {
        "id": 2222,
        "ietf-yp-observation:timestamp": "2025-03-25T08:34:05.22Z",
        "ietf-yp-observation:point-in-time": "state-changed",
        "datastore-contents": {
          "yang-patch": {
            "patch-id": "52",
            "edit": {
              "edit-id": "edit_example_1",
              "operation": "replace",
              "target": "/ietf-interfaces:interfaces",
              "value": {
                "ietf-interfaces:interfaces": {
                  "interface": [{
                    "name": "eth0",
                    "type": "iana-if-type:ethernetCsmacd",
                    "if-index": 1,
                    "admin-status": "up",
                    "oper-status": "down",
                    "statistics": {
                      "discontinuity-time": "2025-03-25T06:43:12Z"
                    }
                  }]
                }
              }
            }
          }
        }
      }
    }
  }
}
```

Figure 8: JSON Push Example for a push-change-update notification message

3.5.1.1.2. Periodical Subscriptions

In periodical subscription, the observation time characterizes the time when the metrics were polled from the datastore before it generates the actual YANG-Push message.

Figure 9 illustrates the delays associated with the generation of the YANG-Push message. The timeline shows the following states:

* T1: First, the periodical subscription is configured to push the operational status of the interface every N interval.

* T2: This is the moment the YANG-Push process polls the data state data and metrics from the datastore.

* T3: After polling the metrics, the YANG-Push process generates the 'push-update' notification.

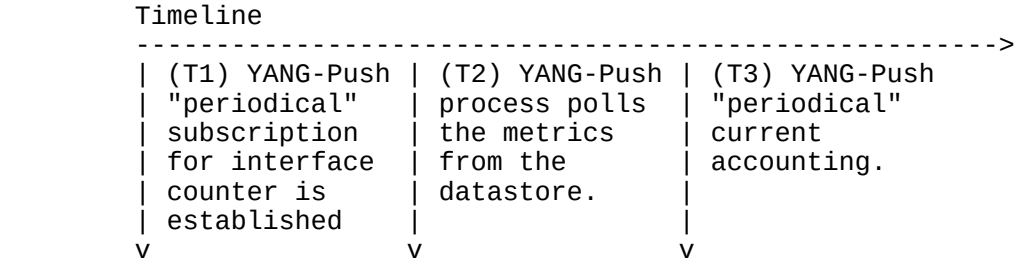

Figure 9: Example timeline for Periodical Subscription

An example of a 'push-update' notification sent at T3 is illustrated in Figure 10. In this case, the node 'event-time' characterizes the moment the YANG-Push process generated the message while the node 'ietf-yp-observation:timestamp' establishes the moment when the metrics were polled from the datastore. For these periodical subscriptions, the 'point-in-time' node must be set to 'current-accounting'.

```
{
  "ietf-yp-notification:envelope": {
    "event-time": "2023-03-25T08:30:12.25Z",
    "hostname": "example-router",
    "sequence-number": 1,
    "contents": {
      "ietf-yang-push:push-update": {
        "id": 6666,
        "ietf-yp-observation:timestamp": "2023-03-25T08:30:00.00Z",
        "ietf-yp-observation:point-in-time": "current-accounting",
        "datastore-contents": {
          "ietf-interfaces:interfaces": {
            "interface": [{
              "name": "eth0",
              "type": "iana-if-type:ethernetCsmacd",
              "if-index": 1,
              "admin-status": "up",
              "oper-status": "up",
              "statistics": {
                "discontinuity-time": "2023-03-25T07:43:12Z",
                "in-octets": 200,
                "in-octets": 250
              }
            }]
          }
        }
      }
    }
  }
}
```

Figure 10: JSON Push Example for a push-update notification message

4. Operational Considerations

The header defined by NETCONF Event Notifications [RFC5277] and the notification envelope defined in this document may coexist in a network. An operator deploying the header defined in this document should take the appropriate actions when both headers are used within the same network. It is RECOMMENDED to deploy one receiver for each header.

5. YANG Modules

5.1. The 'ietf-yp-notification' Module

The following sections show the YANG tree and YANG module for the 'ietf-yp-notification' module.

5.1.1. YANG ietf-yp-notification Tree Diagram

This YANG module extends "ietf-subscribed-notifications" [RFC8641] and "ietf-notification-capabilities" [RFC9196] as shown in the following YANG tree [RFC8340]:

```
module: ietf-yp-notification

  augment /sn:subscriptions:
    +--rw enable-notification-envelope?   boolean
    +--rw metadata
  augment /sysc:system-capabilities/notc:subscription-capabilities:
    +--ro notification-metadata
       +--ro notification-envelope?   boolean
       +--ro metadata
          +--ro hostname-sequence-number?   boolean

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

5.1.2. YANG ietf-yp-notification Module

The YANG module augments the module "ietf-subscribed-notifications" [RFC8641], augments the module "ietf-notification-capabilities" [RFC9196] and uses "ietf-yang-types" module [RFC6991] and "ietf-yang-structure-ext" module [RFC8791].

```
<CODE BEGINS> file "ietf-yp-notification@2025-06-04.yang"
module ietf-yp-notification {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yp-notification";
  prefix iypn;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-subscribed-notifications {
    prefix sn;
```

```
    reference
      "RFC 8639: Subscription to YANG Notifications";
  }
  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-structure-ext {
    prefix sx;
    reference
      "RFC 8791: YANG Data Structure Extensions";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/group/netconf/>
     WG List:  <mailto:netconf@ietf.org>

     Authors:  Alex Huang Feng
               <mailto:alex.huang-feng@insa-lyon.fr>
               Pierre Francois
               <mailto:pierre.francois@insa-lyon.fr>
               Thomas Graf
               <mailto:thomas.graf@swisscom.com>
               Benoit Claise
               <mailto:benoit.claise@huawei.com>";
  description
    "Defines a notification header for Subscribed Notifications
     [RFC8639] and YANG-Push [RFC8641]. When this notification header
     is enabled through configuration, the root container of the
     notification is encoded as defined in RFCXXX.

     This module can be used to validate XML-encoded notifications
     [RFC7950], JSON-encoded messages [RFC7951], and CBOR-encoded
     messages [RFC9254]. Refer to Section Y of RFC XXXX for more
     details.

     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
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.";

    revision 2025-06-04 {
      description
        "First revision";
      reference
        "RFC XXXX: YANG-Push Notification Envelope";
    }

    identity notif-envelope-error {
      description
        "Base identity for errors found while attempting to
         change the value of the leaf 'enable-notification-envelope'.";
    }

    grouping notif-env-capabilities {
      description
        "This grouping defines the capabilities for
         the notification-envelope defined in RFC XXXX
         and the different supported metadata.";
      leaf notification-envelope {
        type boolean;
        default "false";
        description
          "Supports YANG-Push to use the notification-envelope
           defined in RFC XXXX.";
      }
      container metadata {
        description
          "Container with the supported optional metadata by the
           YANG-Push publisher.";
        leaf hostname-sequence-number {
          type boolean;
          default "false";
          description
            "Supports hostname and sequence-number
             in the YANG-Push notifications as defined in the
             YANG-Push notification-envelope in RFC XXXX.";
        }
```

```
    }
  }

  sx:structure envelope {
    leaf event-time {
      type yang:date-and-time;
      mandatory true;
      description
        "The date and time the event was generated by the event
         source. This parameter is of type dateTime and compliant
         to [RFC3339].";
    }
    leaf hostname {
      type inet:host;
      description
        "The hostname of the network node according to
         [RFC1213]. This value is usually configured on the node
         by the administrator to identify the node in
         the network uniquely.";
    }
    leaf sequence-number {
      type yang:counter32;
      description
        "Unique sequence number as described in [RFC9187] for each
         published message.";
    }
    anydata contents {
      description
        "This contains the values defined by the 'notification'
         statement unchanged.";
    }
  }

  // Subscription container
  augment "/sn:subscriptions" {
    description
      "This augmentation adds the configuration switches for
       enabling the notification envelope and metadata.";
    leaf enable-notification-envelope {
      type boolean;
      default "false";
      description
        "Enables YANG-Push to use the notification-envelope
         defined in RFC XXXX.";
    }
    container metadata {
      description
        "Container for configuring optional metadata.";
```

```
      }
    }

    // YANG-Push Capabilities extension
    augment "/sysc:system-capabilities"
          + "/notc:subscription-capabilities" {
      description
        "Extension to the subscription-capabilities model to enable
         clients to learn whether the publisher supports the
         notification-envelope";
      container notification-metadata {
        description
          "Adds the notification metadata capabilities to subscription
           capabilities.";
        uses notif-env-capabilities;
      }
    }
  }
  <CODE ENDS>
```

5.2. The 'ietf-yp-observation' Module

The following sections show the YANG tree and YANG module for the 'ietf-yp-observation' module.

5.2.1. YANG ietf-yp-observation Tree Diagram

This YANG module extends "ietf-yang-push" [RFC8641] and "ietf-notification-capabilities" [RFC9196] as shown in the following YANG tree [RFC8340]:

```
module: ietf-yp-observation

  augment /yp:push-update:
    +--ro timestamp?       yang:date-and-time
    +--ro point-in-time?   enumeration
  augment /yp:push-change-update:
    +--ro timestamp?       yang:date-and-time
    +--ro point-in-time?   enumeration
  augment /sysc:system-capabilities/notc:subscription-capabilities:
    +--ro yang-push-observation-supported?   boolean
            {yang-push-observation-timestamp}?
```

5.2.2. YANG ietf-yp-observation Module

The YANG module augments the module "ietf-yang-push" [RFC8641], augments the module "ietf-system-capabilities" [RFC9196].

```
<CODE BEGINS> file "ietf-yp-observation@2025-06-04.yang"
module ietf-yp-observation {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yp-observation";
  prefix ypot;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-yang-push {
    prefix yp;
    reference
      "RFC 8641: Subscription to YANG Notifications for Datastore Updates";
  }
  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";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";
  contact
    "WG Web:   <http:/tools.ietf.org/wg/netconf/>
     WG List:  <mailto:netconf@ietf.org>

     Authors:  Thomas Graf
               <mailto:thomas.graf@swisscom.com>
               Benoit Claise
               <mailto:benoit.claise@huawei.com>
               Alex Huang Feng
               <mailto:alex.huang-feng@insa-lyon.fr>";
  description
    "Defines YANG-Push event notification header with the observation
     time in streaming update notifications.

     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 2025-06-04 {
      description
        "First revision";
      reference
        "RFC XXXX: Extensible YANG Model for Network Telemetry Notifications";
    }

    feature yang-push-observation-timestamp {
      description
        "This feature indicates the YANG-push Notifications support
         the observation timestamps in streaming update notifications.";
    }

    grouping yang-push-observation {
      description
        "This grouping adds the observation timestamp for the observed metrics.";
      leaf timestamp {
        type yang:date-and-time;
        description
          "This is the time when metrics were observed.";
      }
      leaf point-in-time {
        type enumeration {
          enum current-accounting {
            description
              "For periodical subscriptions, the point-in-time
               where the metrics are being polled and observed.";
          }
          enum initial-state {
            description
              "For 'on-change sync on start' subscriptions, the
               initial point in time when the subscription was established
               and the state was observed.";
          }
          enum state-changed {
            description
              "For 'on-change sync on start' subscriptions, the
               point in time when the state change was observed after the
               subscription was established.";
```

```
          }
        }
        description
          "This describes at which point in time the metrics were observed";
      }
    }

    // Event notifications
    augment "/yp:push-update" {
      description
        "This augmentation adds the observation timestamp of the accounted
         metrics in the push-update notification.";
      uses ypot:yang-push-observation;
    }

    augment "/yp:push-change-update" {
      description
        "This augmentation adds the observation timestamp of the event
         in the push-change-update notification.";
      uses ypot:yang-push-observation;
    }

    // Event capabilities
    augment "/sysc:system-capabilities"
          + "/notc:subscription-capabilities" {
      description
        "Add YANG-Push notification capabilities to system-level capability
         container.";
      leaf yang-push-observation-supported {
        if-feature "yang-push-observation-timestamp";
        type boolean;
        description
          "Specifies whether the publisher supports exporting
           observation-timestamp and point-in-time in notifications.";
        reference
          "RFC XXXX: Extensible YANG Model for YANG-Push Notifications";
      }
    }
  }
  <CODE ENDS>
```

6. Implementation Status

Note to the RFC-Editor: Please remove this section before publishing.

6.1. Huawei VRP

Huawei implemented in push-update and push-change-update notifications the timestamp and point-in-time extension as described in Section 3.5 for a YANG-Push publisher on UDP-based Transport for Configured Subscriptions [I-D.ietf-netconf-udp-notif] in their VRP platform.

6.2. 6WIND VSR

6WIND implemented in push-update and push-change-update notifications the timestamp and point-in-time extension as described in Section 3.5 for a YANG-Push publisher on UDP-based Transport for Configured Subscriptions [I-D.ietf-netconf-udp-notif] in their VSR platform.

6.3. Cisco IOS XR

Cisco implemented in push-update and push-change-update notifications the timestamp and point-in-time extension as described in Section 3.5 for a YANG-Push publisher on UDP-based Transport for Configured Subscriptions [I-D.ietf-netconf-udp-notif] in their IOS XR platform.

7. Security Considerations

This section uses the template described in Section 3.7 of [I-D.ietf-netmod-rfc8407bis].

The "ietf-yp-notification" and "ietf-yp-observation" YANG modules defines data models that are designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. These protocols have to use a secure transport layer (e.g., SSH [RFC6242], TLS [RFC8446], and QUIC [RFC9000]) and have to use mutual authentication.

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

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., "config true", which is the default). All writable data nodes are likely to be reasonably sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) and delete operations to these data nodes without proper protection or authentication can have a negative effect on network operations. The following subtrees and data nodes have particular sensitivities/vulnerabilities:

* /sn:subscriptions/iypn:enable-notification-envelope

The entries in the list above will show whether the mechanism defined in this document is enabled. Access control MUST be set so that only someone with proper access permissions has the ability to access and modify this resource.

8. IANA Considerations

This document describes the URI used for the IETF XML Registry and registers a new YANG module name.

8.1. URI

IANA is requested to add this document as a reference in the following URI's in the IETF XML Registry [RFC3688].

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

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

8.2. YANG module name

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

name: ietf-yp-notification
namespace: urn:ietf:params:xml:ns:yang:ietf-yp-notification
prefix: iypn
reference: RFC-to-be

name: ietf-yp-observation
namespace: urn:ietf:params:xml:ns:yang:ietf-yp-observation
prefix: ypot
reference: RFC-to-be

8.3. YANG SID-file

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

SID range entry point: TBD
SID range size: 50
YANG module name: ietf-yp-notification
reference: RFC-to-be

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

9. Acknowledgements

The authors would like to thank Per Anderson, Andy Bierman, Carsten Bormann, Mohamed Boucadair, Tom Petch, Jason Sterne, Kent Watsen and Rob Wilton for their review and valuable comments.

10. References

10.1. Normative References

[RFC1213] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, DOI 10.17487/RFC1213, March 1991, <https://www.rfc-editor.org/info/rfc1213>.

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

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

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

[RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event Notifications", RFC 5277, DOI 10.17487/RFC5277, July 2008, <https://www.rfc-editor.org/info/rfc5277>.

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

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

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

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

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

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

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

[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://www.rfc-editor.org/info/rfc8639>.

[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/info/rfc8641>.

[RFC8791] Bierman, A., Bj旦rklund, M., and K. Watsen, "YANG Data Structure Extensions", RFC 8791, DOI 10.17487/RFC8791, June 2020, <https://www.rfc-editor.org/info/rfc8791>.

[RFC9187] Touch, J., "Sequence Number Extension for Windowed Protocols", RFC 9187, DOI 10.17487/RFC9187, January 2022, <https://www.rfc-editor.org/info/rfc9187>.

[RFC9196] Lengyel, B., Clemm, A., and B. Claise, "YANG Modules Describing Capabilities for Systems and Datastore Update Notifications", RFC 9196, DOI 10.17487/RFC9196, February 2022, <https://www.rfc-editor.org/info/rfc9196>.

[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/info/rfc9254>.

[W3C.REC-xml-20001006]
Bray, T., Paoli, J., Sperberg-McQueen, M., and E. Maler, "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C, October 2000, <https://www.w3.org/TR/2000/REC-xml-20001006>.

10.2. Informative References

[I-D.ietf-netconf-udp-notif]
Feng, A. H., Francois, P., Zhou, T., Graf, T., and P. Lucente, "UDP-based Transport for Configured Subscriptions", Work in Progress, Internet-Draft, draft-ietf-netconf-udp-notif-21, 14 May 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-udp-notif-21>.

[I-D.ietf-netmod-rfc8407bis]
Bierman, A., Boucadair, M., and Q. Wu, "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", Work in Progress, Internet-Draft, draft-ietf-netmod-rfc8407bis-28, 5 June 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-rfc8407bis-28>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6470] Bierman, A., "Network Configuration Protocol (NETCONF) Base Notifications", RFC 6470, DOI 10.17487/RFC6470, February 2012, <https://www.rfc-editor.org/info/rfc6470>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

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

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

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

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

[RFC9232] Song, H., Qin, F., Martinez-Julia, P., Ciavaglia, L., and A. Wang, "Network Telemetry Framework", RFC 9232, DOI 10.17487/RFC9232, May 2022, <https://www.rfc-editor.org/info/rfc9232>.

[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/info/rfc9595>.

Appendix A. .sid file

Note to the RFC-Editor: Please remove this section before publishing.

For CBOR encoding using YANG-SIDs identifiers, a ".sid" file is requested to IANA in Section 8.3.

<CODE BEGINS> file "ietf-yp-notification@2025-06-04.sid"

```
{
  "ietf-sid-file:sid-file": {
    "module-name": "ietf-yp-notification",
    "module-revision": "2025-06-04",
    "sid-file-status":"unpublished",
    "description": "draft-ietf-netconf-notif-envelope-02: Extensible YANG Model for YA
NG-Push Notifications",
    "dependency-revision": [
      {
        "module-name":"ietf-datastores",
        "module-revision":"2018-02-14"
      },
      {
        "module-name":"ietf-interfaces",
        "module-revision":"2018-02-20"
      },
      {
        "module-name":"ietf-ip",
        "module-revision":"2018-02-22"
      },
      {
        "module-name":"ietf-netconf-acm",
```

```
          "module-revision":"2018-02-14"
        },
        {
          "module-name":"ietf-network-instance",
          "module-revision":"2019-01-21"
        },
        {
          "module-name":"ietf-restconf",
          "module-revision":"2017-01-26"
        },
        {
          "module-name":"ietf-yang-library",
          "module-revision":"2019-01-04"
        },
        {
          "module-name":"ietf-yang-patch",
          "module-revision":"2017-02-22"
        },
        {
          "module-name":"ietf-yang-schema-mount",
          "module-revision":"2019-01-14"
        },
        {
          "module-name": "ietf-yang-types",
          "module-revision": "2013-07-15"
        },
        {
          "module-name": "ietf-inet-types",
          "module-revision": "2021-02-22"
        },
        {
          "module-name": "ietf-subscribed-notifications",
          "module-revision": "2019-09-09"
        },
        {
          "module-name": "ietf-system-capabilities",
          "module-revision": "2022-02-17"
        },
        {
          "module-name": "ietf-notification-capabilities",
          "module-revision": "2022-02-17"
        },
        {
          "module-name": "ietf-yang-structure-ext",
          "module-revision": "2020-06-17"
        }
      ],
      "assignment-range": [
```

```
          {
            "entry-point": "2950",
            "size": "50"
          }
        ],
        "item": [
          {
            "status":"unstable",
            "namespace":"module",
            "identifier":"ietf-yp-notification",
            "sid":"2950"
          },
          {
            "status":"unstable",
            "namespace":"identity",
            "identifier":"notif-envelope-error",
            "sid":"2951"
          },
          {
            "status":"unstable",
            "namespace":"data",
            "identifier":"/ietf-subscribed-notifications:subscriptions/ietf-yp-notificatio
n:enable-notification-envelope",
            "sid":"2952"
          },
          {
            "status":"unstable",
            "namespace":"data",
            "identifier":"/ietf-subscribed-notifications:subscriptions/ietf-yp-notificatio
n:metadata",
            "sid":"2953"
          },
          {
            "status":"unstable",
            "namespace":"data",
            "identifier":"/ietf-system-capabilities:system-capabilities/ietf-notification-
capabilities:subscription-capabilities/ietf-yp-notification:notification-metadata",
            "sid":"2954"
          },
          {
            "status":"unstable",
            "namespace":"data",
            "identifier":"/ietf-system-capabilities:system-capabilities/ietf-notification-
capabilities:subscription-capabilities/ietf-yp-notification:notification-metadata/notific
ation-envelope",
            "sid":"2955"
          },
          {
            "status":"unstable",
            "namespace":"data",
            "identifier":"/ietf-system-capabilities:system-capabilities/ietf-notification-
capabilities:subscription-capabilities/ietf-yp-notification:notification-metadata/metadat
a",
            "sid":"2956"
          },
```

```
        {
          "status":"unstable",
          "namespace":"data",
          "identifier":"/ietf-system-capabilities:system-capabilities/ietf-notification-
capabilities:subscription-capabilities/ietf-yp-notification:notification-metadata/metadat
a/hostname-sequence-number",
          "sid":"2957"
        },
        {
          "status":"unstable",
          "namespace":"data",
          "identifier":"/ietf-yp-notification:envelope",
          "sid":"2958"
        },
        {
          "status":"unstable",
          "namespace":"data",
          "identifier":"/ietf-yp-notification:envelope/contents",
          "sid":"2959"
        },
        {
          "status":"unstable",
          "namespace":"data",
          "identifier":"/ietf-yp-notification:envelope/event-time",
          "sid":"2960"
        },
        {
          "status":"unstable",
          "namespace":"data",
          "identifier":"/ietf-yp-notification:envelope/hostname",
          "sid":"2961"
        },
        {
          "status":"unstable",
          "namespace":"data",
          "identifier":"/ietf-yp-notification:envelope/sequence-number",
          "sid":"2962"
        }
      ]
    }
  }
  <CODE ENDS>
```

Figure 11: .sid file for "ietf-yp-notification" module

Appendix B. Example extending the Notification Envelope

This appendix provides a non-normative example illustrating how an extension to the notification envelope should be implemented. The example does not define valid metadata, but serves to demonstrate how future extensions should be specified.

Specifically, the example extends the notification envelope with a metadata field named "foo". The YANG module illustrates the necessary augmentations to both the 'ietf-yp-notification' module and the 'ietf-notification-capabilities' module.

<CODE BEGINS> file "example-foo-extension.yang"

```
module example-foo-extension {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:example-foo-extension";
  prefix fooext;

  import ietf-subscribed-notifications {
    prefix sn;
  }
  import ietf-system-capabilities {
    prefix sysc;
  }
  import ietf-notification-capabilities {
    prefix notc;
  }
  import ietf-yang-structure-ext {
    prefix sx;
  }
  import ietf-yp-notification {
    prefix iypn;
  }

  description
    "Defines a new 'foo' metadata for the notification envelope.";

  // Extending the notification envelope header with a new
  // 'foo' metadata
  sx:augment-structure "/iypn:envelope" {
    leaf foo {
      type string;
      description
        "Description of the 'foo' extension.";
    }
  }

  // Extending the notifications capabilities so that clients can
```

```
     // learn whether this new extension is supported or not
     augment "/sysc:system-capabilities"
           + "/notc:subscription-capabilities"
           + "/iypn:notification-metadata/iypn:metadata" {
       description
         "Extension to the subscription-capabilities model to enable
          clients to learn whether the publisher supports the new
          'foo' metadata.";
       leaf foo {
         type boolean;
         default "false";
         description
           "Adds the 'foo' capability.";
       }
     }

     // (Optional) A user can optionally add knobs for enabling and
     // disabling specific metadata.
     augment "/sn:subscriptions/iypn:metadata" {
       description
         "An user can optionally, support a configuration knob for enabling
          or disabling a metadata.";
       leaf foo {
         type boolean;
         default "false";
         description
           "Configuration knob for enabling and disabling the 'foo' metadata.";
       }
     }
   }
   <CODE ENDS>

   The extensions result in the following YANG tree:

   module: example-foo-extension

     augment /sysc:system-capabilities/notc:subscription-capabilities
               /iypn:notification-metadata/iypn:metadata:
       +--ro foo?   boolean
     augment /sn:subscriptions/iypn:metadata:
       +--rw foo?   boolean

     augment-structure /iypn:envelope:
       +-- foo?   string
```

Authors' Addresses

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

Pierre Francois
INSA-Lyon
Lyon
France
Email: pierre.francois@insa-lyon.fr

Thomas Graf
Swisscom
Binzring 17
CH-8045 Zurich
Switzerland
Email: thomas.graf@swisscom.com

Benoit Claise
Huawei
Email: benoit.claise@huawei.com