NETCONF
Internet-Draft
Intended status: Standards Track
Expires: August 12, 2018

A. Gonzalez Prieto
VMware
E. Voit
Cisco Systems
A. Clemm
Huawei
E. Nilsen-Nygaard
A. Tripathy
Cisco Systems

February 8, 2018

NETCONF Support for Event Notifications
draft-ietf-netconf-netconf-event-notifications-07

Abstract

This document provides a NETCONF binding for [I-D.draft-ietf-netconf-subscribed-notifications] and [I-D.ietf-netconf-yang-push]. Included are:

- transport mappings for subscription RPCs, state change notifications, and notification messages,
- functional requirements, and
- examples

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document defines a binding for events streamed over the NETCONF protocol [RFC6241] as per [I-D.draft-ietf-netconf-subscribed-notifications]. In addition, as [I-D.ietf-netconf-yang-push] is itself built upon [I-D.draft-ietf-netconf-subscribed-notifications], this document enables a NETCONF client to request and receive updates from a YANG datastore located on a NETCONF server.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

The following terms are defined in [I-D.draft-ietf-netconf-subscribed-notifications]: notification message, stream, publisher, receiver, subscriber, subscription, configured subscription.

3. Interleave Capability

To support multiple subscriptions on a single session, a NETCONF publisher MUST support the :interleave capability as defined in [RFC5277]. Such support MUST be indicated by the following capability: "urn:ietf:params:netconf:capability:interleave:1.0". Advertisement of this capability along with support [I-D.draft-ietf-netconf-subscribed-notifications] will indicate that a NETCONF publisher is able to receive, process, and respond to NETCONF requests and [I-D.draft-ietf-netconf-subscribed-notifications] subscription operations on a session with active subscriptions.

4. Mandatory XML, stream and datastore support

A NETCONF publisher MUST support XML encoding of RPCs and Notifications.

A NETCONF publisher supporting [I-D.draft-ietf-netconf-subscribed-notifications] MUST support the "NETCONF" event stream identified in that draft.

A NETCONF publisher supporting [I-D.ietf-netconf-yang-push] MUST support the operational state datastore as defined by [I.D.draft-ietf-netmod-revised-datastores].

5. Transport connectivity

5.1. Dynamic Subscriptions

For dynamic subscriptions, if the NETCONF session involved with the "establish-subscription" terminates, the subscription MUST be deleted.

5.2. Configured Subscriptions

For a configured subscription, there is no guarantee a transport session is currently in place with each associated receiver. In cases where a configured subscription has a receiver in the connecting state and the protocol configured as NETCONF, but no NETCONF transport session exists to that receiver, the publisher MUST initiate a transport session via NETCONF call home [RFC8071], section 4.1 to that receiver. Until NETCONF connectivity is established and a subscription-started state change notification is successfully sent, that receiver MUST remain in a status of either "connecting" or "timeout".

If the call home fails because the publisher receives receiver credentials which are subsequently declined per [RFC8071], Section 4.1, step S5 authentication, then that receiver MUST be assigned a "timeout" status.

If the call home fails to establish for any other reason, the publisher MUST NOT progress the receiver to the "active" state. Additionally, the publisher SHOULD place the receiver into a "timeout" status after a predetermined number of either failed call home attempts or NETCONF sessions remotely terminated by the receiver.

NETCONF Transport session connectivity SHOULD be verified via Section 4.1, step S7.

If an active NETCONF session is disconnected but the stop-time of a subscription has not been reached, the publisher MUST restart the call home process and return the receiver to the "connecting" state.

6. Notification Messages

Notification messages transported over NETCONF will be identical in format and content to those encoded using one-way operations defined within [RFC5277], section 4.

7. Dynamic Subscriptions and RPC Error Responses

Management of dynamic subscriptions occurs via RPCs as defined in [I-D.ietf-netconf-yang-push] and [I-D.draft-ietf-netconf-subscribed-notifications]. When an RPC error occurs, the NETCONF RPC reply MUST include an "rpc-error" element per [RFC6241] with the error information populated as follows:

- "error-type" of "application".
- "error-tag" of "operation-failed".

- Optionally, an "error-severity" of "error" (this MAY but does not have to be included).
- "error-app-tag" with the value being a string that corresponds to an identity associated with the error, as defined in [I-D.draft-ietf-netconf-subscribed-notifications] section 2.4.6 for general subscriptions, and [I-D.ietf-netconf-yang-push] Appendix A.1, for datastore (YANG-Push) subscriptions. The tag to use depends on the RPC for which the error occurred. Applicable are identities with a base identity of "establish-subscription-error" (for error responses to an establish-subscription request), "modify- subscription-error" (for error responses to a modify-subscription request), "delete-subscription-error" (for error responses to a delete-subscription request), "resynch-subscription-error" (for error responses to resynch-subscription request), or "kill- subscription-error" (for error responses to a kill-subscription request), respectively.
- In case of error responses to an establish-subscription or modify-subscription request: optionally, "error-info" containing XML-encoded data with hints regarding parameter settings that might lead to successful requests in the future, per yang-data definitions "establish-subscription-error-datastore" (for error responses to an establish-subscription request) or "modify-subscription-error-datastore (for error responses to a modify-subscription request), respectively.

  These yang-data that is included in "error-info" SHOULD NOT include the optional leaf "error-reason", as such a leaf would be redundant with the information that is already placed within the error-app-tag.

  In case of an rpc error as a result of a delete-subscription, or a kill-subscription, or a resynch-subscription request, no error-info needs to be included, as the subscription-id is the only RPC input parameter and no hints regarding RPC input parameters need to be provided.

Note that "error-path" does not need to be included with the "rpc-error" element, as subscription errors are generally not associated with nodes in the datastore but with the choice of RPC input parameters.

8. Security Considerations

Notification messages (including state change notifications) are never sent before the NETCONF capabilities exchange has completed.

If a malicious or buggy NETCONF subscriber sends a number of "establish-subscription" requests, then these subscriptions

accumulate and may use up system resources. In such a situation, subscriptions MAY be terminated by terminating the underlying NETCONF session. The publisher MAY also suspend or terminate a subset of the active subscriptions on that NETCONF session.

The NETCONF Authorization Control Model [RFC6536] SHOULD be used to control and restrict authorization of subscription configuration.

9. Acknowledgments

We wish to acknowledge the helpful contributions, comments, and suggestions that were received from: Andy Bierman, Yan Gang, Sharon Chisholm, Hector Trevino, Peipei Guo, Susan Hares, Tim Jenkins, Balazs Lengyel, Kent Watsen, and Guangying Zheng.

10. Normative References

[I-D.draft-ietf-netconf-subscribed-notifications]
Voit, E., Clemm, A., Gonzalez Prieto, A., Tripathy, A., and E. Nilsen-Nygaard, "Custom Subscription to Event Streams", draft-ietf-netconf-subscribed-notifications-10 (work in progress), January 2018.

[I-D.ietf-netconf-yang-push]
Clemm, Alexander., Voit, Eric., Gonzalez Prieto, Alberto., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "YANG Datastore Subscription", February 2018, <https://datatracker.ietf.org/doc/draft-ietf-netconf-yang-push/>.

[I.D.draft-ietf-netmod-revised-datastores]
Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-10 (work in progress), January 2018.

[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>.

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

[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>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[RFC8071] Watsen, K., "NETCONF Call Home and RESTCONF Call Home", RFC 8071, DOI 10.17487/RFC8071, February 2017, <https://www.rfc-editor.org/info/rfc8071>.

Appendix A. Examples

A.1. Event Stream Discovery

As defined in [I-D.draft-ietf-netconf-subscribed-notifications] an event stream exposes a continuous set of events available for subscription. A NETCONF client can retrieve the list of available event streams from a NETCONF publisher using the "get" operation against the top-level container "/streams" defined in [I-D.draft-ietf-netconf-subscribed-notifications].

The following example illustrates the retrieval of the list of available event streams using the "get" operation.

```
<rpc message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <streams
     xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"/>
    </filter>
  </get>
</rpc>
```

Figure 1: Get streams request

After such a request, the NETCONF publisher returns a list of event streams available.

A.2. Dynamic Subscriptions

A.2.1. Establishing Dynamic Subscriptions

The following figure shows two successful "establish-subscription" RPC requests as per [I-D.draft-ietf-netconf-subscribed-notifications]. The first request is given a subscription identifier of 22, the second, an identifier of 23.

```
      +------------+                 +-----------+
      | Subscriber |                 | Publisher |
      +------------+                 +-----------+
            |                              |
            |    Capability Exchange       |
            |<---------------------------->|
            |                              |
            |                              |
            |    establish-subscription    |
            |----------------------------->|  (a)
            | RPC Reply: OK, id = 22       |
            |<-----------------------------|  (b)
            |                              |
            | notification message (for 22)|
            |<-----------------------------|
            |                              |
            |                              |
            |    establish-subscription    |
            |----------------------------->|
            | RPC Reply: OK, id = 23       |
            |<-----------------------------|
            |                              |
            |                              |
            | notification message (for 22)|
            |<-----------------------------|
            | notification message (for 23)|
            |<-----------------------------|
            |                              |
```

Figure 2: Multiple subscriptions over a NETCONF session

To provide examples of the information being transported, example messages for interactions (a) and (b) in Figure 2 are detailed below:

```
  <rpc netconf:message-id="102"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <establish-subscription
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      <stream>
        <name>NETCONF</name>
        <xpath-filter xmlns:ex="http://example.com/events">
             /ex:foo
        </xpath-filter>
      </stream>
      <dscp>
         10
      </dscp>
    </establish-subscription>
  </rpc>
```

Figure 3: establish-subscription request (a)

As NETCONF publisher was able to fully satisfy the request (a), the publisher sends the subscription identifier of the accepted subscription within message (b):

```
  <rpc-reply message-id="102"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <identifier
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      22
    </identifier>
  </rpc-reply>
```

Figure 4: establish-subscription success (b)

If the NETCONF publisher had not been able to fully satisfy the request, or subscriber has no authorization to establish the subscription, the publisher would have sent an RPC error response. For instance, if the "dscp" value of 10 asserted by the subscriber in Figure 3 proved unacceptable, the publisher may have returned:

```
<rpc-reply message-id="102"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rpc-error>
   <error-type>application</error-type>
   <error-tag>operation-failed</error-tag>
   <error-severity>error</error-severity>
   <error-app-tag>
    dscp-unavailable
   </error-app-tag>
   <error-info>
    <establish-subscription-error-datastore
     xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
     <dscp>
      100
     </dscp>
   </error-info>
  </rpc-error>
</rpc-reply>
```

Figure 5: an unsuccessful establish subscription

The subscriber can use this information in future attempts to establish a subscription.

A.2.2. Modifying Dynamic Subscriptions

An existing subscription may be modified. The following exchange shows a negotiation of such a modification via several exchanges between a subscriber and a publisher. This negotiation consists of a failed RPC modification request/response, followed by a successful one.

```
      +------------+                 +-----------+
      | Subscriber |                 | Publisher |
      +------------+                 +-----------+
            |                              |
            | notification message (for 23)|
            |<-----------------------------|
            |                              |
            | modify-subscription (id = 23)|
            |----------------------------->|  (c)
            | RPC error (with hint)        |
            |<-----------------------------|  (d)
            |                              |
            | modify-subscription (id = 23)|
            |----------------------------->|
            | RPC Reply: OK                |
            |<-----------------------------|
            |                              |
            | notification message (for 23)|
            |<-----------------------------|
            |                              |
```

Figure 6: Interaction model for successful subscription modification

If the subscription being modified in Figure 6 is a datastore subscription as per [I-D.ietf-netconf-yang-push], the modification request made in (c) may look like that shown in Figure 7. As can be seen, the modifications being attempted are the application of a new xpath filter as well as the setting of a new periodic time interval.

```
<rpc message-id="303"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <modify-subscription
       xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"
       xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <yp:datastore>
      <yp:xpath-filter xmlns="http://example.com/datastore">
        /interfaces-state/interface/oper-status
      </yp:xpath-filter>
      <yp:periodic>
        <yp:period>500</yp:period>
      </yp:periodic>
    </yp:datastore>
    <identifier>
      23
    </identifier>
  </modify-subscription>
</rpc>
```

Figure 7: Subscription modification request (c)

If the NETCONF publisher can satisfy both changes, the publisher sends a positive result for the RPC. If the NETCONF publisher cannot satisfy either of the proposed changes, the publisher sends an RPC error response (d). The following is an example RPC error response for (d) which includes a hint. This hint is an alternative time period value which might have resulted in a successful modification:

```
   <rpc-reply message-id="303"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
     <rpc-error>
       <error-type>application</error-type>
       <error-tag>operation-failed</error-tag>
       <error-severity>error</error-severity>
       <error-app-tag>
           period-unsupported
       </error-message>
       <error-info
          xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
          <modify-subscription-error-datastore>
            <period-hint>
                3000
            </period-hint>
          </modify-subscription-error-datastore>
       </error-info>
     </rpc-error>
   </rpc-reply>
```

Figure 8: Modify subscription failure with Hint (d)

A.2.3. Deleting Dynamic Subscriptions

The following demonstrates deleting a subscription.

```
  <rpc message-id="103"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <delete-subscription
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      <identifier>22</identifier>
    </delete-subscription>
  </rpc>
```

Figure 9: Delete subscription

If the NETCONF publisher can satisfy the request, the publisher replies with success to the RPC request.

If the NETCONF publisher cannot satisfy the request, the publisher sends an error-rpc element indicating the modification didn't work. Figure 10 shows a valid response for existing valid subscription identifier, but that subscription identifier was created on a different NETCONF transport session:

```
   <rpc-reply message-id="103"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
     <rpc-error>
       <error-type>application</error-type>
       <error-tag>operation-failed</error-tag>
       <error-severity>error</error-severity>
       <error-app-tag>
           no-such-subscription
       </error-app-tag>
     </rpc-error>
   </rpc-reply>
```

Figure 10: Unsuccessful delete subscription

A.3. Configured Subscriptions

Configured subscriptions may be established, modified, and deleted using configuration operations against the top-level subtree of [I-D.draft-ietf-netconf-subscribed-notifications] or [I-D.ietf-netconf-yang-push].

In this section, we present examples of how to manage the configuration subscriptions using a NETCONF client.

A.3.1. Creating Configured Subscriptions

For subscription creation, a NETCONF client may send:

```
<rpc message-id="201"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <subscriptions
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      <subscription>
        <identifier>22</identifier>
        <encoding>encode-xml</encoding>
        <stream>
          <name>NETCONF</name>
          <receiver>
            <address>1.2.3.4</address>
            <port>1234</port>
          </receiver>
        </stream>
      </subscription>
    </subscriptions>
  </edit-config>
</rpc>
```

Figure 11: Create a configured subscription

If the request is accepted, the publisher will indicate this. If the request is not accepted because the publisher cannot serve it, no configuration is changed. In this case the publisher may reply:

```
<rpc-reply message-id="201"
           xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <rpc-error>
        <error-type>application</error-type>
        <error-tag>resource-denied</error-tag>
        <error-severity>error</error-severity>
        <error-message xml:lang="en">
            Temporarily the publisher cannot serve this
            subscription due to the current workload.
        </error-message>
    </rpc-error>
</rpc-reply>
```

Figure 12: Response to a failed configured subscription establishment

After a subscription has been created, NETCONF connectivity to each receiver's IP address and port will be established if it does not already exist. This will be accomplished via [RFC8071].

The following figure shows the interaction model for the successful creation of a configured subscription.

```
    +----------+                 +-----------+     +---------+
    |Config Ops|                 | Publisher |     | 1.2.3.4 |
    +----------+                 +-----------+     +---------+
         |                            |                |
         |    Capability Exchange     |                |
         |<-------------------------->|                |
         |                            |                |
         |                            |                |
         |        Edit-config         |                |
         |--------------------------->|                |
         |       RPC Reply: OK        |                |
         |<---------------------------|                |
         |                            |   Call Home    |
         |                            |<-------------->|
         |                            |                |
         |                            |  subscription- |
         |                            |  started       |
         |                            |--------------->|
         |                            |                |
         |                            |  notification  |
         |                            |  message       |
         |                            |--------------->|
```

Figure 13: Interaction model for configured subscription establishment

A.3.2. Modifying Configured Subscriptions

Configured subscriptions can be modified using configuration operations against the top-level container "/subscriptions".

For example, the subscription established in the previous section could be modified as follows, here a adding a second receiver:

```
<rpc message-id="202"
       xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
       xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <subscriptions
   xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      <subscription>
        <identifier>
          1922
        </identifier>
        <receiver>
          <address>
            1.2.3.5
          </address>
          <port>
            1234
          </port>
        </receiver>
      </subscription>
    </subscriptions>
  </edit-config>
</rpc>
```

Figure 14: Modify configured subscription

If the request is accepted, the publisher will indicate success. The result is that the interaction model described in Figure 13 may be extended as follows.

```
 +----------+                 +-----------+     +---------+  +---------+
 |Config Ops|                 | Publisher |     | 1.2.3.4 |  | 1.2.3.5 |
 +----------+                 +-----------+     +---------+  +---------+
       |                            |  notification  |            |
       |                            |  message       |            |
       |                            |--------------->|            |
       |        Edit-config         |                |            |
       |--------------------------->|                |            |
       |       RPC Reply: OK        |                |            |
       |<---------------------------|                |            |
       |                            |  subscription- |            |
       |                            |  started       |            |
       |                            |----------------------------->|
       |                            |                |            |
       |                            |  notification  |            |
       |                            |  message       |            |
       |                            |--------------->|            |
       |                            |----------------------------->|
       |                            |                |            |
```

Figure 15: Interaction model for configured subscription modification

Note in the above that in the specific example above, modifying a configured subscription actually resulted in "subscription-started" notification. And because of an existing NETCONF session, no additional call home was needed. Also note that if the edit of the configuration had impacted the filter, a separate modify-subscription would have been required for the original receiver.

A.3.3. Deleting Configured Subscriptions

Configured subscriptions can be deleted using configuration operations against the top-level container "/subscriptions". Deleting the subscription above would result in the following flow impacting all active receivers.

```
 +----------+                 +-----------+     +---------+  +---------+
 |Config Ops|                 | Publisher |     | 1.2.3.4 |  | 1.2.3.5 |
 +----------+                 +-----------+     +---------+  +---------+
       |                            |                |            |
       |                            |  notification  |            |
       |                            |  message       |            |
       |                            |--------------->|            |
       |                            |---------------------------->|
       |                            |                |            |
       |        Edit-config         |                |            |
       |--------------------------->|                |            |
       |       RPC Reply: OK        |                |            |
       |<---------------------------|                |            |
       |                            |  subscription- |            |
       |                            |  terminated    |            |
       |                            |--------------->|            |
       |                            |---------------------------->|
       |                            |                |            |
```

Figure 16: Interaction model for configured subscription deletion

A.4. Subscription State Notifications

A publisher will send subscription state notifications according to the definitions within [I-D.draft-ietf-netconf-subscribed-notifications]).

A.4.1. subscription-started and subscription-modified

A "subscription-started" over NETCONF encoded in XML would look like:

```
 <notification
   xmlns=" urn:ietf:params:xml:ns:netconf:notification:1.0">
   <eventTime>2007-09-01T10:00:00Z</eventTime>
   <subscription-started
     xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"/>
     <identifier>39</identifier>
     <encoding>encode-xml</encoding>
     <stream>
       <name>NETCONF</name>
       <xpath-filter xmlns:ex="http://example.com/events">
            /ex:foo
       </xpath-filter>
     </stream>
   </subscription-started>
 </notification>
```

Figure 17: subscription-started subscription state notification

The "subscription-modified" is identical to Figure 17, with just the word "started" being replaced by "modified".

A.4.2. subscription-completed, subscription-resumed, and replay-complete

A "subscription-completed" would look like:

```
  <notification
    xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
    <eventTime>2007-09-01T10:00:00Z</eventTime>
    <subscription-completed
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      <identifier>39</identifier>
    </subscription-completed>
  </notification>
```

Figure 18: subscription-completed notification in XML

The "subscription-resumed" and "replay-complete" are virtually identical, with "subscription-completed" simply being replaced by "subscription-resumed" and "replay-complete" in both encodings.

A.4.3. subscription-terminated and subscription-suspended

A "subscription-terminated" would look like:

```
<notification
  xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2007-09-01T10:00:00Z</eventTime>
  <subscription-terminated
    xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
    <identifier>39</identifier>
    <error-id>
       suspension-timeout
    </error-id>
  </subscription-terminated>
</notification>
```

Figure 19: subscription-terminated subscription state notification

The "subscription-suspended" is virtually identical, with "subscription-terminated" simply being replaced by "subscription-suspended".

Appendix B. Changes between revisions

(To be removed by RFC editor prior to publication)

B.1. v06 to v07

- XML encoding and operational datastore mandatory.
- Error mechanisms and examples updated.

B.2. v05 to v06

- Moved examples to appendices
- All examples rewritten based on namespace learnings
- Normative text consolidated in front
- Removed all mention of JSON
- Call home process detailed
- Note: this is a major revision attempting to cover those comments received from two week review.

B.3. v03 to v04

- Added additional detail to "configured subscriptions"
- Added interleave capability
- Adjusted terminology to that in draft-ietf-netconf-subscribed-notifications
- Corrected namespaces in examples

B.4. v01 to v03

- Text simplifications throughout
- v02 had no meaningful changes

B.5. v00 to v01

- Added Call Home in solution for configured subscriptions.
- Clarified support for multiple subscription on a single session. No need to support multiple create-subscription.
- Added mapping between terminology in yang-push and [RFC6241] (the one followed in this document).
- Editorial improvements.

Authors' Addresses

Alberto Gonzalez Prieto
VMware

Email: agonzalezpri@vmware.com

Eric Voit
Cisco Systems

Email: evoit@cisco.com

Alexander Clemm
Huawei

Email: ludwig@clemm.org

Einar Nilsen-Nygaard
Cisco Systems

Email: einarnn@cisco.com

Ambika Prasad Tripathy
Cisco Systems

Email: ambtripa@cisco.com