NETCONF
Internet-Draft
Intended status: Standards Track
Expires: 23 April 2026

T. Zhou
G. Zheng
Huawei
E. Voit
Cisco Systems
T. Graf
Swisscom
P. Francois
INSA-Lyon

20 October 2025

Subscription to Notifications in a Distributed Architecture
draft-ietf-netconf-distributed-notif-16

Abstract

This document describes extensions to the YANG notifications subscription to allow metrics being published directly from processors on line cards to target receivers, while subscription is still maintained at the route processor in a distributed forwarding system.

Requirements Language

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

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 23 April 2026.

Table of Contents

1. Introduction

The mechanism to support a subscription of a continuous and customized stream of updates from a YANG datastore [RFC8342] is defined in [RFC8639] and [RFC8641]. Requirements for Subscription to YANG Datastores are defined in [RFC7923].

By streaming YANG-Push notifications from publishers to receivers, much better performance and fine-grained sampling can be achieved than with a polling-based mechanism. In a distributed forwarding system, the packet forwarding is delegated to multiple processors on line cards. To not to overwhelm the route processor resources, it is not uncommon that data records are published directly from processors on line cards to target receivers to further increase efficiency on the routing system.

This document complements the general subscription requirements defined in Section 4.2.1 of [RFC7923] by the paragraph: A Subscription Service MAY support the ability to export from multiple software processes on a single routing system and expose the information which software process produced which message.

2. Terminologies

The following terms are defined in [RFC8639] and are not redefined here:

Subscriber

Publisher

Receiver

Subscription

The following terms are defined in [RFC9196] and are not redefined here:

Capability

In addition, this document defines the following terms:

Global Subscription: The Subscription requested by the subscriber. It may be decomposed into multiple Component Subscriptions.

Component Subscription: The Subscription that defines a data source which is managed and controlled by a single Publisher.

Global Capability: The overall subscription capability that the group of Publishers can expose to the Subscriber as defined in [RFC9196] for the distributed system. This includes which YANG nodes can be subscribed at which minimum-update-period for periodical subscriptions respectively which minimum-dampening-period for on-change subscriptions.

Component Capability: The subscription capability that each Publisher exposes to the Publisher Master. This includes which YANG nodes with which minimum-update-period for periodical subscriptions respectively which minimum-dampening-period for on-change subscriptions can be subscribed on which Component.

Publisher Master: The component of a Publisher that interacts with the Subscriber to deal with the Global Subscription. It decomposes the Global Subscription to multiple Component Subscriptions and interacts with the Publisher Agents.

Publisher Agent: The component of a Publisher that interacts with the Publisher Master to deal with the Component Subscription and pushing the data to the Receiver.

Node: Is the network node which contains one or more Publishers that obtains the data from the YANG datastore and pushes it to the Receiver.

Message Publisher: The Publisher that pushes the message to the Receiver.

Message Publisher ID: A 32-bit identifier of the publishing process that is locally unique to the publisher Node. With this identifier the publishing process from where the message was published from can be uniquely identified. Receivers SHOULD use the transport session and the Publisher ID field to separate different publisher streams originating from the same network Node.

3. Motivation and Solution Overview

In distributed forwarding systems much YANG data is subscribed to processors on line cards and published from the route processor, creating an unnecessary overhead at the route processor and a potential bottleneck.

This document proposes that subscribed YANG data can be published from processors on line cards while for such Subscriptions the route processor is only involved in maintaining and decomposing the Subscription, which includes notifying the Receiver which Subscription is being published from which line card processor with subscription state change notifications.

To enable Receivers to detect notification message loss when publishers share a transport session, the Message Publisher ID in the transport message header of the YANG notification message is introduced. In case of UDP transport, this is described in Section 3.2 of [I-D.ietf-netconf-udp-notif]. With unique sequence-

numbers per publishing process described in Section 3.4.1 of [I-D.ietf-netconf-notif-envelope], each message can uniquely be identified, loss recognized and related to a transport session and publishing process.

4. Solution Detail

Figure 1 below shows the distributed data export framework.

A collector usually includes two components,

* the Subscriber generates the subscription instructions to express what and how the Receiver wants to receive the data;

* the Receiver is the target for the data publication.

For one subscription, there can be one or more Receivers. And the Subscriber does not necessarily share the same IP address as the Receivers.

In this framework, the Publisher pushes messages to the Receiver according to the subscription. The Publisher is either in the Master or Agent role. The Publisher Master knows all the capabilities that his Agents can provide and exposes the Global Capability to the collector. The Subscriber maintains the Global Subscription at the Publisher Master and disassembles the Global Subscription to multiple Component Subscriptions, depending which source data is needed. The Component Subscriptions are then distributed to the corresponding Publisher Agents on route and processors on line cards.

Publisher Agents collects metrics according to the Component Subscription, add its metadata, encapsulates, and pushes messages. Messages may require segmentation depending on the amount of YANG data subscribed and maximum transmission unit of the interface where the message is published from. The Receiver then reassembles packets and decapsulates notifications accordingly.

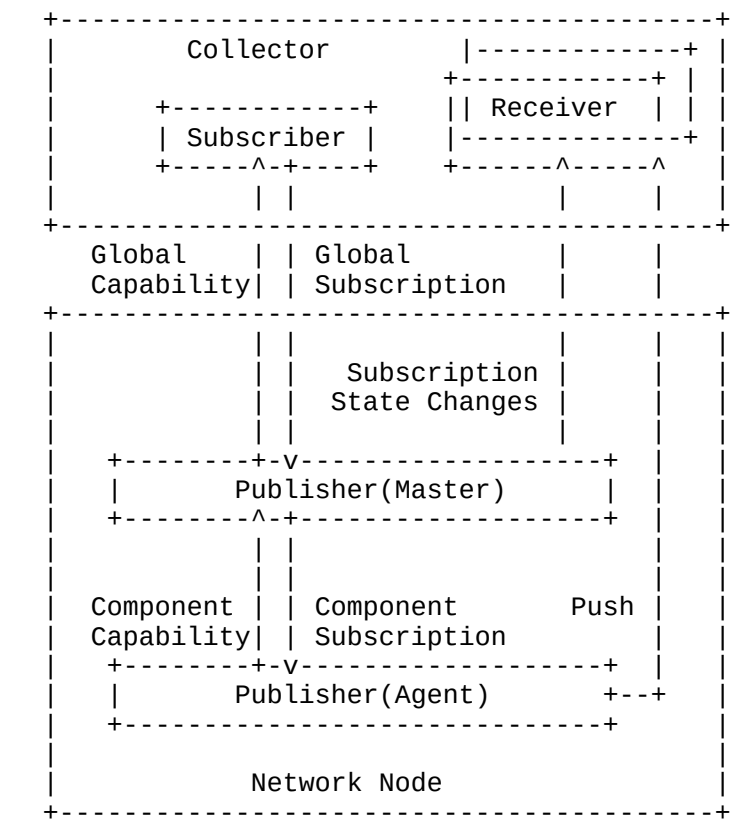

Figure 1: The Distributed Data Export Framework

Publisher Master and Publisher Agents interact with each other in several ways:

* Publisher Agents need to register at the Master at the beginning of their process life cycle.

* Contracts are created and maintained throughout the subscription lifecycle between the Publisher Master and each Agent on the Component Capability, and the format for notification data structure.

* The Publisher Master relays the component subscriptions to the Publisher Agents.

* The Publisher Agents announce the status of their Component Subscriptions to the Publisher Master. The status of the overall subscription is maintained by the Publisher Master. The Publisher Master is responsible for notifying the subscriber in case of problems with the Component Subscriptions.

The technical mechanisms or protocols used for the coordination of operational information between Publisher Master and Agent is out of scope of this document.

5. Subscription Decomposition

The Collector can only subscribe to the Master. This requires the Publisher Master to:

1. expose the Global Capability that can be served by multiple Publisher Agents;

2. disassemble the Global Subscription to multiple Component Subscriptions, and distribute them to the Publisher Agents of the corresponding metric sources from the YANG schema tree so that they do not overlap; How the subscription is being distributed is implementation specific and not part of this document.

3. notify changes related to the existing subscriptions to the different Publisher Agents.

And the Publisher Agent to:

* Inherit the Global Subscription properties from Publisher Master for its Component Subscription;

* share the same life-cycle as the Global Subscription;

* share the same Subscription ID as the Global Subscription.

6. Publication Composition

The Publisher Agent collects data and encapsulates the packets per Component Subscription. The format and structure of the data records are defined by the YANG schema, so that the decomposition at the Receiver can benefit from the structured and hierarchical data records.

The Receiver can associate the YANG data records with Subscription ID [RFC8639] to the subscribed subscription. Additionally, it can use the Message Publisher ID to determine the corresponding publisher process.

For the dynamic subscription, the output of the "establish-subscription" RPC defined in [RFC8639] MUST include a list of Message Publisher IDs to indicate how the Global Subscription is decomposed into several Component Subscriptions.

The "subscription-started" and "subscription-modified" notification defined in [RFC8639] and "push-update" and "push-change-update" notification defined in [RFC8641] MUST also include a list of Message Publisher IDs to notify the current Publishers for the corresponding Global Subscription.

7. Subscription State Change Notifications

In addition to sending event records to Receivers, the Master MUST also send subscription state change notifications [RFC8639] when events related to subscription management have occurred. All the subscription state change notifications MUST be delivered by the Master.

When the subscription decomposition result changed, the "subscription-modified" notification MUST be sent to indicate the new list of Publisher Agents.

8. Publisher Configurations

This document assumes that all Publisher Agents are preconfigured to push data. The actual working Publisher Agents are selected based on the subscription decomposition result.

All Publisher Agents share the same source IP address for data export. For connectionless data transport such as UDP based transport [I-D.ietf-netconf-udp-notif] the same Layer 4 source port for data export can be used. For connection based data transport such as HTTPS based transport [I-D.ietf-netconf-https-notif], each Publisher Agent MUST be able to acknowledge packet retrieval from Receivers, and therefore requires a dedicated Layer 4 source port per software process.

The specific configuration on transports is described in the responsible documents.

9. YANG Tree

```
module: ietf-distributed-notif

  augment /sn:subscriptions/sn:subscription:
    +--ro message-publisher-id*   uint32
  augment /sn:subscription-started:
    +--ro message-publisher-id*   uint32
  augment /sn:subscription-modified:
    +--ro message-publisher-id*   uint32
  augment /sn:establish-subscription/sn:output:
    +--ro message-publisher-id*   uint32
  augment /yp:push-update:
    +--ro message-publisher-id?   uint32
  augment /yp:push-change-update:
    +--ro message-publisher-id?   uint32
```

Figure 2: YANG tree diagram for 'ietf-distributed-notif' module.

10. YANG Module

```
<CODE BEGINS> file "ietf-distributed-notif@2025-04-12.yang"
module ietf-distributed-notif {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-distributed-notif";
  prefix dn;

  import ietf-subscribed-notifications {
    prefix sn;
  }
  import ietf-yang-push {
    prefix yp;
    reference
      "RFC 8641: Subscription to YANG Notifications for Datastore
      Updates";
  }

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

     Authors:  Guangying Zheng
               <mailto:zhengguangying@huawei.com>
               Tianran Zhou
               <mailto:zhoutianran@huawei.com>
               Thomas Graf
               <mailto:thomas.graf@swisscom.com>
               Pierre Francois
```

```
                <mailto:pierre.francois@insa-lyon.fr>
                Eric Voit
                <mailto:evoit@cisco.com>";
    description
      "Defines augmentation for ietf-subscribed-notifications to
       enable the distributed publication with single subscription.

       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).

       All revisions of IETF and IANA published modules can be found
       at the YANG Parameters registry
       (https://www.iana.org/assignments/yang-parameters).

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

    revision 2025-04-12 {
      description
        "Initial version";
      reference
        "RFC XXXX: Subscription to Distributed Notifications";
    }

    grouping message-publisher-id {
      description
        "Provides a reusable leaf of the message-publisher-id.";
      leaf message-publisher-id {
        type uint32;
        config false;
        description
          "Identifies the software process which publishes the
           message (e.g., processor 1 on line card 1). This field
           is used to notify the receiver which publisher process
           published which message.";
      }
    }

    grouping message-publisher-ids {
      description
        "Provides a reusable leaf-list of message-publisher-id-list.";
```

```
    leaf-list message-publisher-id {
      type uint32;
      config false;
      description
        "Identifies the software process which publishes the
         message (e.g., processor 1 on line card 1). This field
         is used to notify the receiver which publisher processes
         are going to publish.";
    }
  }

  augment "/sn:subscriptions/sn:subscription" {
    description
      "This augmentation allows the Message
       Publisher ID to be exposed for a subscription.";
    uses message-publisher-ids;
  }

  augment "/sn:subscription-started" {
    description
      "This augmentation adds the Message Publisher ID to the
       subscription-started subscription change notifications.";
    uses message-publisher-ids;
  }

  augment "/sn:subscription-modified" {
    description
      "This augmentation adds the Message Publisher ID to the
       subscription-modified subscription change notifications.";
    uses message-publisher-ids;
  }

  augment "/sn:establish-subscription/sn:output" {
    description
      "This augmentation adds the Message Publisher ID to the
       dynamic establish-subscription output.";
    uses message-publisher-ids;
  }

  augment "/yp:push-update" {
    description
      "This augmentation adds the Message Publisher ID in the
       push-update notification.";
    uses message-publisher-id;
  }

  augment "/yp:push-change-update" {
    description
```

```
        "This augmentation adds the Message Publisher ID in the
         push-change-update notification.";
      uses message-publisher-id;
    }
  }
  <CODE ENDS>
```

11. IANA Considerations

This document registers the following namespace URI in the IETF XML Registry [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-distributed-notif

Registrant Contact: The IESG.

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

This document registers the following YANG module in the YANG Module Names registry [RFC3688]:

Name: ietf-distributed-notif

Namespace: urn:ietf:params:xml:ns:yang:ietf-distributed-notif

Prefix: dn

Reference: RFC XXXX

12. Implementation Status

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

12.1. Huawei VRP

Huawei implemented the Subscription Decomposition described in this document for a YANG-Push publisher on UDP-based Transport for Configured Subscriptions [I-D.ietf-netconf-udp-notif] in their VRP platform.

12.2. 6WIND VSR

6WIND implemented the Subscription Decomposition described in this document for a YANG-Push publisher on UDP-based Transport for Configured Subscriptions [I-D.ietf-netconf-udp-notif] in their VSR platform.

13. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

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

The new data nodes introduced in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get-config or notification) to this data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

* /subscriptions/subscription/message-publisher-ids

The entries in the two lists above will show where subscribed resources might be located on the publishers. Access control MUST be set so that only someone with proper access permissions has the ability to access this resource.

Other Security Considerations is the same as those discussed in [RFC8639].

14. Contributors

Alexander Clemm
Futurewai
2330 Central Expressway
Santa Clara
California
United States of America
Email: ludwig@clemm.org

15. Acknowledgements

We thank Kent Watsen, Mahesh Jethanandani, Martin Bjorklund, Tim Carey, Qin Wu, Robert Wilton, Benoit Claise, Alex Huang Feng Camilo Cardona and Juergen Schoenwaelder for their constructive suggestions for improving this document.

16. References

16.1. Normative References

[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>.

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

[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>.

[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>.

[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>.

[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>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[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>.

[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>.

[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>.

16.2. Informative References

[I-D.ietf-netconf-https-notif]
Jethanandani, M. and K. Watsen, "An HTTPS-based Transport for YANG Notifications", Work in Progress, Internet-Draft, draft-ietf-netconf-https-notif-15, 1 February 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-https-notif-15>.

[I-D.ietf-netconf-notif-envelope]
Feng, A. H., Francois, P., Graf, T., and B. Claise, "Extensible YANG Model for YANG-Push Notifications", Work in Progress, Internet-Draft, draft-ietf-netconf-notif-envelope-02, 17 June 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-notif-envelope-02>.

[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-23, 14 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-udp-notif-23>.

[RFC7923] Voit, E., Clemm, A., and A. Gonzalez Prieto, "Requirements for Subscription to YANG Datastores", RFC 7923, DOI 10.17487/RFC7923, June 2016, <https://www.rfc-editor.org/info/rfc7923>.

Appendix A. Examples

This appendix is non-normative.

A.1. Dynamic Subscription

Figure 3 shows a typical dynamic subscription to the network Node with distributed data export capability.

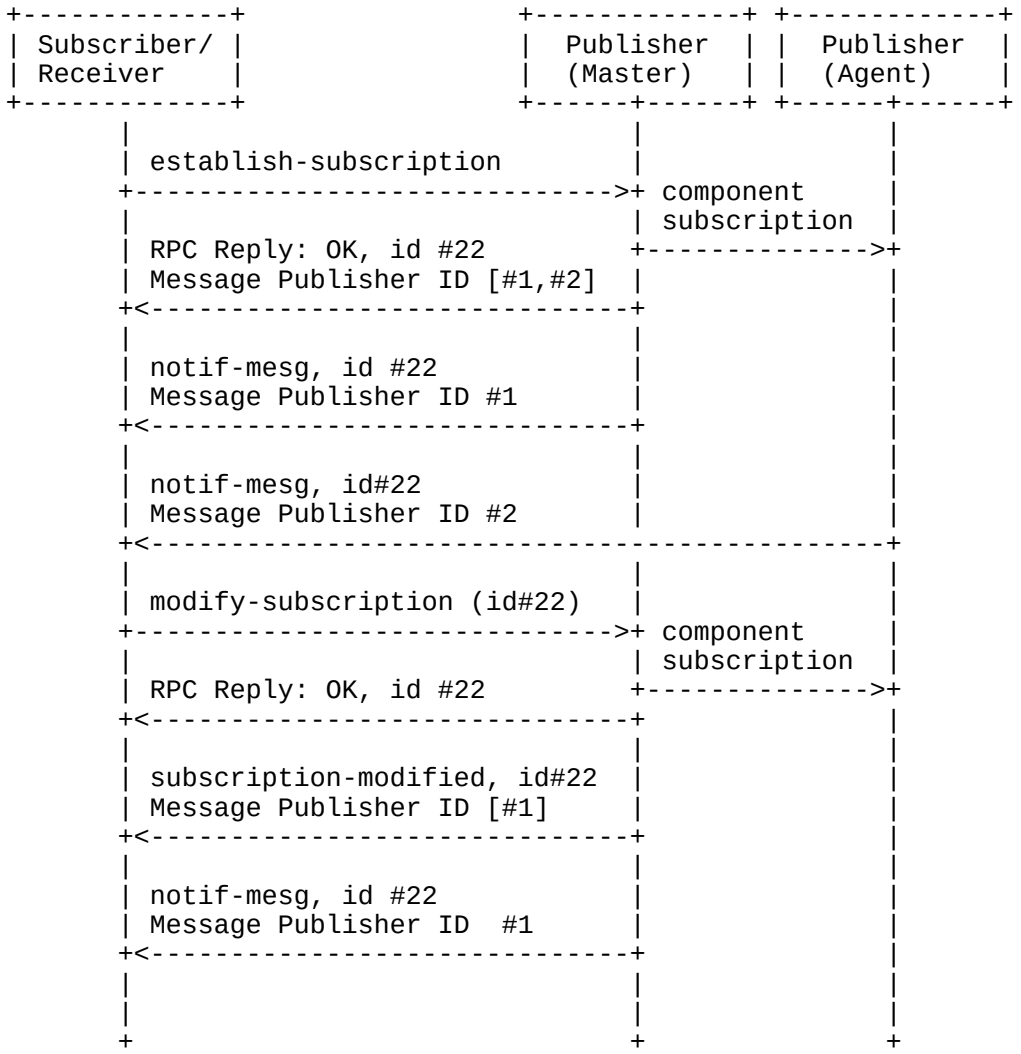

Figure 3: Call Flow for Dynamic Subscription

A "establish-subscription" RPC request as per [RFC8641] is sent to the Master with a successful response. An example of using NETCONF:

```
<netconf:rpc message-id="101"
   xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
 <establish-subscription
    xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"
    xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
   <yp:datastore
       xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
     ds:operational
   </yp:datastore>
   <yp:datastore-xpath-filter
      xmlns:ex="https://example.com/sample-data/1.0">
     /ex:foo
   </yp:datastore-xpath-filter>
   <yp:periodic>
     <yp:period>500</yp:period>
   </yp:periodic>
  </establish-subscription>
 </netconf:rpc>
```

Figure 4: "establish-subscription" Request

As the network Node is able to fully satisfy the request, the request is given a subscription ID of 22. The response as in Figure 5 indicates that the subscription is decomposed into two component subscriptions which will be published by two message Message Publisher ID: #1 and #2.

```
<rpc-reply message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <id
   xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
     22
  </id>
  <message-publisher-id
   xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
     1
  </message-publisher-id>
  <message-publisher-id
   xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
     2
  </message-publisher-id>
</rpc-reply>
```

Figure 5: "establish-subscription" Positive RPC Response

Then, both Publishers send notifications with the corresponding piece of data to the Receiver.

The subscriber may invoke the "modify-subscription" RPC for a subscription it previously established. The RPC has no difference to the single publisher case as in [RFC8641]. Figure 6 provides an example where a subscriber attempts to modify the period and datastore XPath filter of a subscription using NETCONF.

```
     <rpc message-id="102"
       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">
      <id>22</id>
      <yp:datastore
          xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
        ds:operational
      </yp:datastore>
      <yp:datastore-xpath-filter
          xmlns:ex="https://example.com/sample-data/1.0">
        /ex:bar
      </yp:datastore-xpath-filter>
      <yp:periodic>
        <yp:period>250</yp:period>
      </yp:periodic>
     </modify-subscription>
  </rpc>
```

Figure 6: "modify-subscription" Request

If the modification is successfully accepted, the "subscription-modified" subscription state notification is sent to the subscriber by the Master. The notification, Figure 7 for example, indicates the modified subscription is decomposed into one component subscription which will be published by message Message Publisher ID #1.

```
 <notification
       xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
 <eventTime>2007-09-01T10:00:00Z</eventTime>
 <subscription-modified
       xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"
       xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
     <id>22</id>
     <yp:datastore
         xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
       ds:operational
     </yp:datastore>
     <yp:datastore-xpath-filter
         xmlns:ex="https://example.com/sample-data/1.0">
       /ex:bar
     </yp:datastore-xpath-filter>
     <yp:periodic>
         <yp:period>250</yp:period>
     </yp:periodic>
     <message-publisher-id
     xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
        1
     </message-publisher-id>
   </subscription-modified>
 </notification>
```

Figure 7: "subscription-modified" Subscription State Notification

A.2. Configured Subscription

Figure 8 shows a typical configured subscription to the network Node with distributed data export capability.

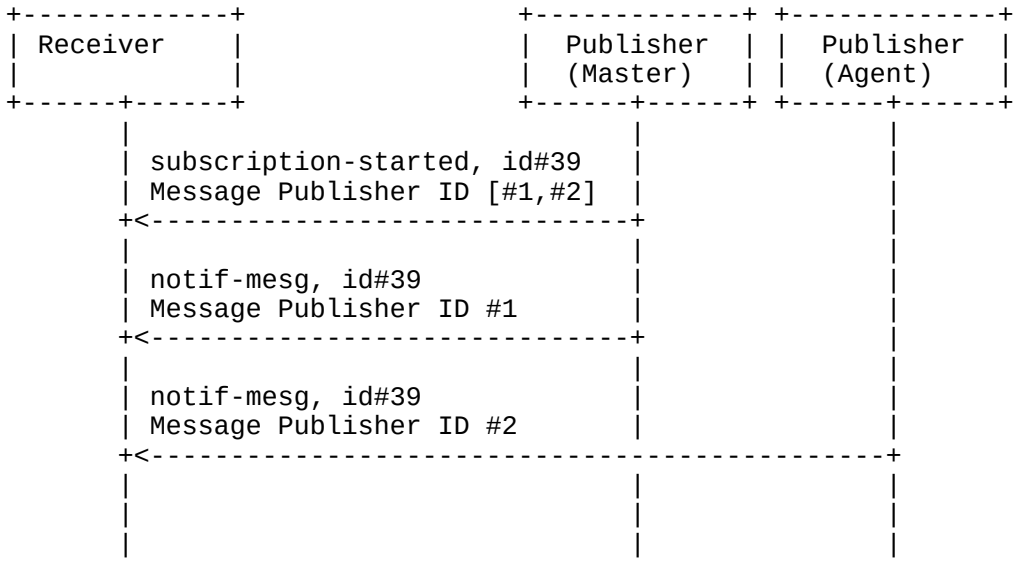

Figure 8: Call Flow for Configured Subscription

Before starting to push data, the "subscription-started" subscription state notification is sent to the Receiver. The following example assumes the NETCONF transport has already established. The notification indicates that the configured subscription is decomposed into two component subscriptions which will be published by two message Message Publisher IDs: #1 and #2.

```
 <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"
       xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
     <identifier>39</identifier>
     <yp:datastore
         xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
     ds:operational
     </yp:datastore>
     <yp:datastore-xpath-filter
         xmlns:ex="https://example.com/sample-data/1.0">
       /ex:foo
     </yp:datastore-xpath-filter>
     <yp:periodic>
         <yp:period>250</yp:period>
     </yp:periodic>
     <message-publisher-ids
     xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications>
        1
     </message-publisher-ids>
     <message-publisher-ids
     xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications>
        2
     </message-publisher-ids>
   </subscription-started>
 </notification>
```

Figure 9: "subscription-started" Subscription State Notification

Then, both Publishers send notifications with the corresponding data record to the Receiver.

Authors' Addresses

Tianran Zhou
Huawei
156 Beiqing Rd., Haidian District
Beijing
China
Email: zhoutianran@huawei.com

Guangying Zheng
Huawei
101 Yu-Hua-Tai Software Road
Nanjing
Jiangsu,
China
Email: zhengguangying@huawei.com

Eric Voit
Cisco Systems
United States of America
Email: evoit@cisco.com

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

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