

NETCONF
Internet-Draft
Intended status: Standards Track
Expires: April 6, 2021

T. Zhou
G. Zheng
Huawei
E. Voit
Cisco Systems
T. Graf
Swisscom
P. Francois
INSA-Lyon
October 3, 2020

Subscription to Distributed Notifications
draft-ietf-netconf-distributed-notif-00

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", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

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 April 6, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
2.	Terminologies	3
3.	Solution Overview	4
4.	Subscription Decomposition	6
5.	Publication Composition	6
6.	Subscription State Change Notifications	7
7.	Publisher Configurations	7
8.	YANG Tree	7
9.	YANG Module	7
10.	IANA Considerations	9
11.	Security Considerations	10
12.	Contributors	11
13.	Acknowledgements	11
14.	References	11
14.1.	Normative References	11
14.2.	Informative References	12
Appendix A.	Examples	12
A.1.	Dynamic Subscription	12
A.2.	Configured Subscription	16
	Authors' Addresses	18

[1.](#) Introduction

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

By streaming data from publishers to receivers, much better performance and fine-grained sampling can be achieved than with polling. 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 to maintain data integrity.

2. Terminologies

The following terms are defined in [\[RFC8639\]](#) and are not redefined here:

Subscriber

Publisher

Receiver

Subscription

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

Global Capability: is the overall subscription capability that the group of Publishers can expose to the Subscriber.

Component Capability: is the subscription capability that each Publisher can expose to the Subscriber.

Master: is the 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 Agents.

Agent: is the Publisher that interacts with the Master to deal with the Component Subscription and pushing the data to the collector.

3. Solution Overview

Figure 2 below shows the distributed data export framework.

A collector usually includes two components,

- o the Subscriber generates the subscription instructions to express what and how the collector want to receive the data;
- o the Receiver is the target for the data publication.

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

In this framework, the Publisher pushes data to the Receiver according to the subscription. The Publisher is either in the Master or Agent role. The Master knows all the capabilities that his Agents are able to provide and exposes the Global Capability to the collector. The Subscriber maintains the Global Subscription at the Master and disassembles the Global Subscription to multiple Component Subscriptions, depending from 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, encapsulate and pushes data to the Receiver where packets are reassembled and decapsulated.

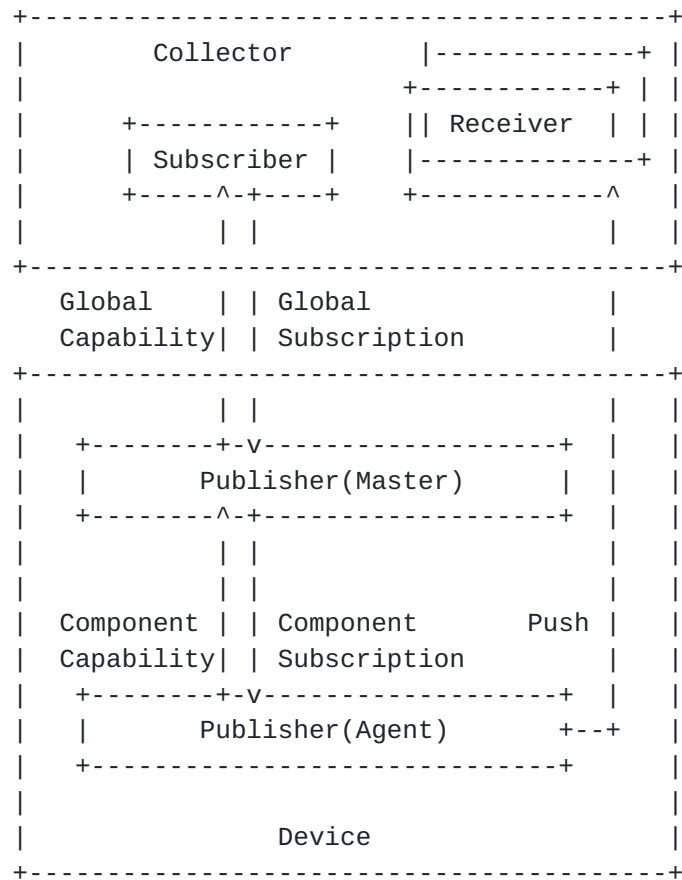


Fig. 2 The Distributed Data Export Framework

Master and Agents interact with each other in several ways:

- o Agents need to register at the Master at the beginning of their process life-cycle
- o Contracts are created between the Master and each Agent on the Component Capability, and the format for streaming data structure.
- o The Master relays the component subscriptions to the Agents.
- o The Agents announce the status of their Component Subscriptions to the Master. The status of the overall subscription is maintained by the Master. The 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 Master and Agent is out-of-scope of this document.

4. Subscription Decomposition

The Collector can only subscribe to the Master. This requires the 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 so that they not overlap;
3. notify on changes when portions of a subscription moving between different Publisher Agents over time.

And the Agent to:

- o Inherit the Global Subscription properties from Publisher Master for its Component Subscription;
- o share the same life-cycle as the Global Subscription;
- o share the same Subscription ID as the Global Subscription.

5. 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 is able to associate the YANG data records with Subscription ID [[RFC8639](#)] to the subscribed subscription and with Message Generator ID [[I-D.ietf-netconf-notification-messages](#)] to one of the Publisher Agents software processes to enable message integrity.

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

The "subscription-started" and "subscription-modified" notification defined in [[RFC8639](#)] MUST also include a list of Message Generator IDs to notify the current Publishers for the corresponding Global Subscription.

6. 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 Publishers.

7. 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.unyte-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.

8. YANG Tree

```
module: ietf-distributed-notifications
  augment /sn:subscriptions/sn:subscription:
    +--ro message-generator-id*   string
  augment /sn:subscription-started:
    +--ro message-generator-id*   string
  augment /sn:subscription-modified:
    +--ro message-generator-id*   string
  augment /sn:establish-subscription/sn:output:
    +--ro message-generator-id*   string
```

9. YANG Module

```
<CODE BEGINS> file "ietf-distributed-notifications@2020-05-09.yang"
module ietf-distributed-notif {
  yang-version 1.1;
  namespace
```



```
"urn:ietf:params:xml:ns:yang:ietf-distributed-notifications";
prefix mso;
import ietf-subscribed-notifications {
  prefix sn;
}

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

  Editor:    Tianran Zhou
             <mailto:zhoutianran@huawei.com>

  Editor:    Guangying Zheng
             <mailto:zhengguangying@huawei.com>";

description
  "Defines augmentation for ietf-subscribed-notifications to
  enable the distributed publication with single subscription.

  Copyright (c) 2018 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject to
  the license terms contained in, the Simplified BSD License set
  forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (https://trustee.ietf.org/license-info).

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

revision 2020-05-09 {
  description
    "Initial version";
  reference
    "RFC XXXX: Subscription to Distributed Notifications";
}

grouping message-generator-ids {
  description
    "Provides a reusable list of message-generator-ids.";

  leaf-list message-generator-id {
    type string;
```



```
    config false;
    ordered-by user;
    description
      "Software process which created the message (e.g.,
        processor 1 on linecard 1). This field is
        used to notify the collector the working originator.";
  }
}

augment "/sn:subscriptions/sn:subscription" {
  description
    "This augmentation allows the message generators to be
    exposed for a subscription.";

  uses message-generator-ids;
}

augment "/sn:subscription-started" {
  description
    "This augmentation allows MSO specific parameters to be
    exposed for a subscription.";

  uses message-generator-ids;
}

augment "/sn:subscription-modified" {
  description
    "This augmentation allows MSO specific parameters to be
    exposed for a subscription.";

  uses message-generator-ids;
}

augment "/sn:establish-subscription/sn:output" {
  description
    "This augmentation allows MSO specific parameters to be
    exposed for a subscription.";

  uses message-generator-ids;
}
}
<CODE ENDS>
```

10. IANA Considerations

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

URI: urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications

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-subscribed-notifications

Namespace: urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications

Prefix: mso

Reference: RFC XXXX

11. 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 RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [[RFC5246](#)].

The NETCONF Access Control Model (NACM) [[RFC6536](#)] 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.

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:

- o /subscriptions/subscription/message-generator-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 YANG-Push [[RFC8641](#)].

12. Contributors

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

13. Acknowledgements

We thank Kent Watsen, Mahesh Jethanandani, Martin Bjorklund, Tim Carey and Qin Wu for their constructive suggestions for improving this document.

14. References

14.1. 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>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [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>>.
- [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>>.

- [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>>.
- [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>>.
- [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>>.

[14.2. Informative References](#)

- [I-D.ietf-netconf-https-notif]
Jethanandani, M. and K. Watsen, "An HTTPS-based Transport for Configured Subscriptions", [draft-ietf-netconf-https-notif-04](#) (work in progress), July 2020.
- [I-D.ietf-netconf-notification-messages]
Voit, E., Jenkins, T., Birkholz, H., Bierman, A., and A. Clemm, "Notification Message Headers and Bundles", [draft-ietf-netconf-notification-messages-08](#) (work in progress), November 2019.
- [I-D.unyte-netconf-udp-notif]
Zhou, T., Zheng, G., Lucente, P., Graf, T., and P. Francois, "UDP-based Transport for Configured Subscriptions", [draft-unyte-netconf-udp-notif-00](#) (work in progress), July 2020.

[Appendix A. Examples](#)

This appendix is non-normative.

[A.1. Dynamic Subscription](#)

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

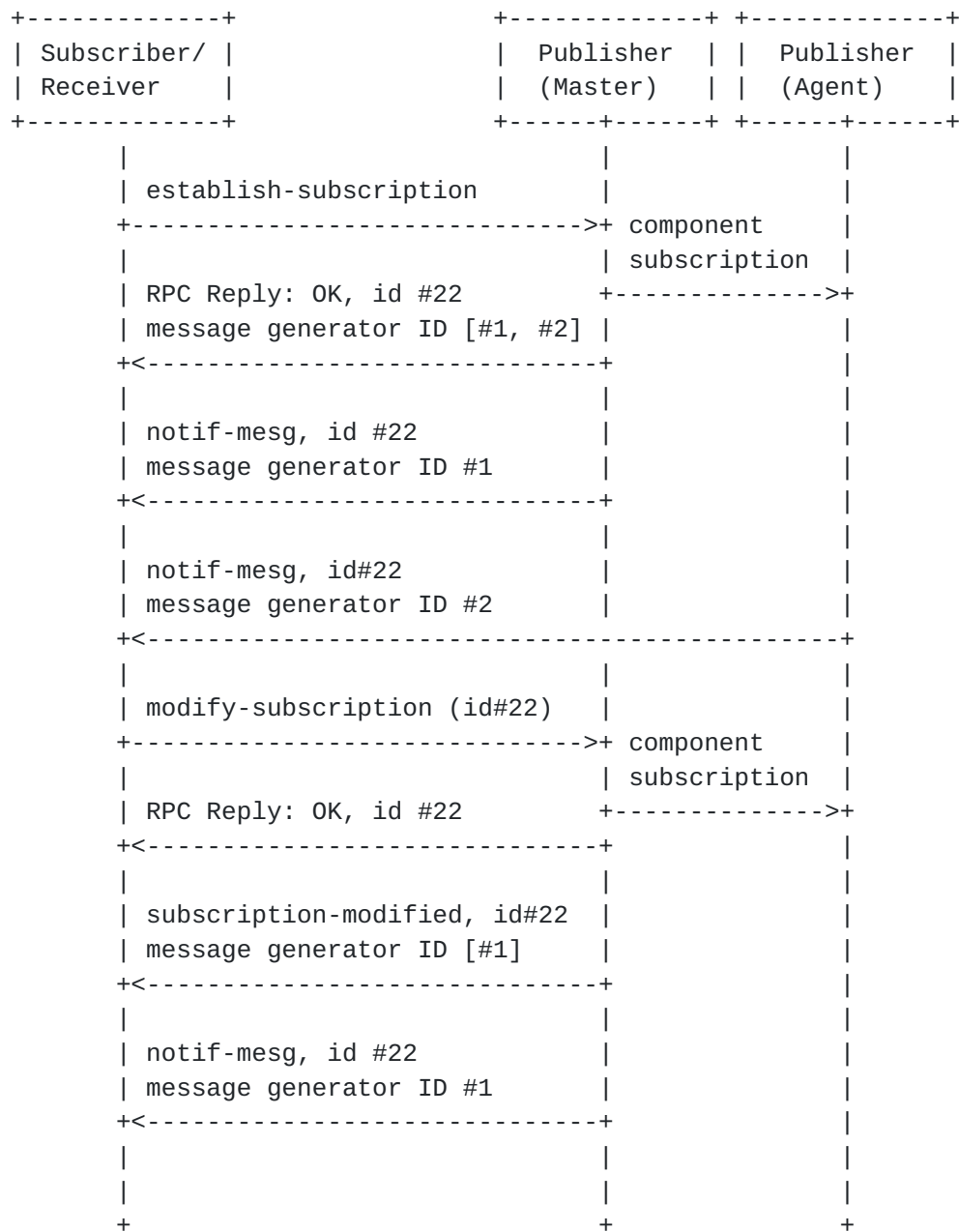


Fig. 3 Call Flow for Dynamic Subscription

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>
```

Fig. 4 "establish-subscription" Request

As the device 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 generators: #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-generator-id
    xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
    1
  </message-generator-id>
  <message-generator-id
    xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
    2
  </message-generator-id>
</rpc-reply>
```

Fig. 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>
```

Fig. 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 generator #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-generator-id
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      1
    </message-generator-id>
  </subscription-modified>
</notification>
```

Fig. 7 "subscription-modified" Subscription State Notification

[A.2.](#) Configured Subscription

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

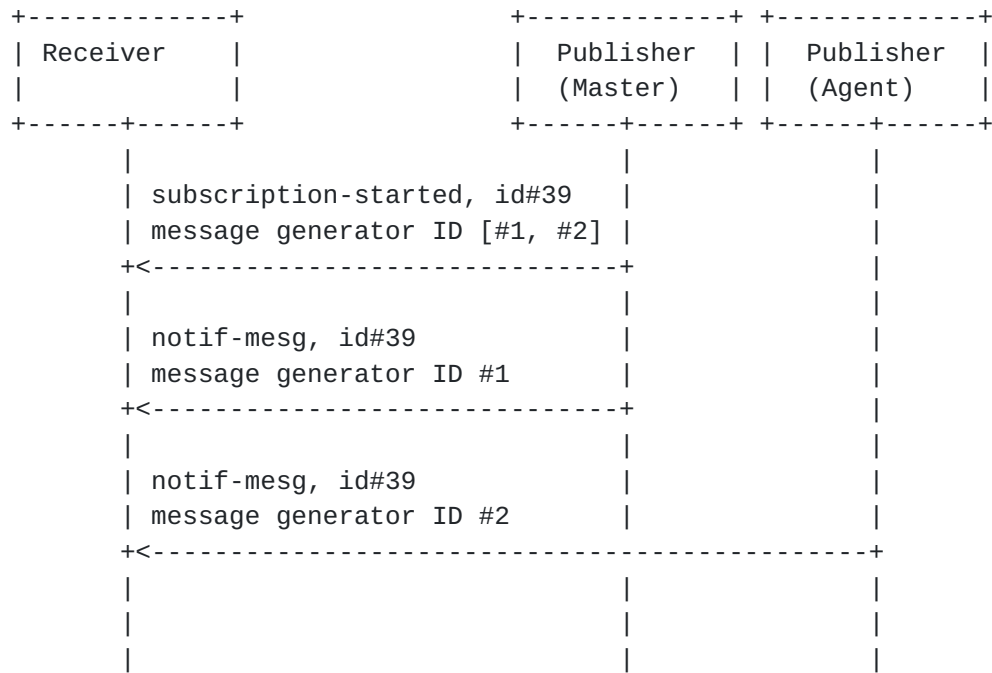


Fig. 8 Call Flow for Configured Subscription

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 generators: #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-generator-id
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      1
    </message-generator-id>
    <message-generator-id
      xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">
      2
    </message-generator-id>
  </subscription-started>
</notification>
```

Fig. 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
Zuerich 8045
Switzerland

Email: thomas.graf@swisscom.com

Pierre Francois
INSA-Lyon
Lyon
France

Email: pierre.francois@insa-lyon.fr

