NETMOD Working Group Q. Wu
Internet-Draft R. Ranade
Intended status: Standards Track Huawei
Expires: March 18, 2019 September 14, 2018

NMDA Base Notification for Applied Intended Configuration
draft-wu-netmod-base-notification-nmda-00

Abstract

The Network Configuration Protocol (NETCONF)and RESTCONF provides mechanisms to manipulate configuration datastores. NMDA introduces additional datastores for systems that support more advanced processing chains converting configuration to operational state. However, client applications are not able to be aware of common events in these additional datstores of the management system, such as a applied configuration state change in NETCONF server or RESTCONF server, that may impact management applications. This document define a YANG module that allows a client to receive additional notifications for some common system events pertaining to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

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 March 18, 2019.

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] and RESTCONF [RFC8040] provides mechanisms to manipulate configuration datastores. NMDA introduces additional datastores (e.g., <intended>, <operational>) for systems that support more advanced processing chains converting configuration to operational state. However, client applications are not able to be aware of common events in these additional datastores of the management system, e.g., there are many background activities (e.g.,NMDA datastore consistency checking [NMDA-DIFF]) that happen during the time that configuration is committed to <running> to the time that the configuration is actually applied to <operational>. It is possible that some configuration could not be applied to <operational> due to either validation issues, or missing resource etc. There is a need for user to know the applying result of <intended> data-store and the reason why the configuration were not applied.

This document define a YANG module that allows a NMS client to receive additional notifications for some common system events pertaining to the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. These notifications are designed to support the monitoring of the base system events within the server and not specific to any network management protocols such as NETCONF and RESTCONF.

The solution presented in this document is backwards compatible with [RFC6470].

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 following terms are defined in [RFC8342] and are not redefined here:

- operational state datastore
- running configuration datastore
- intended configuration datastore

2. NMDA Base Notifications for applied intended configuration

2.1. Overview

The YANG module in NETCONF Base Notifications [RFC6470] specifies the following 5 event notifications for the 'NETCONF' stream to notify a client application that the NETCONF server state has changed:

- netconf-config-change
- netconf-capability-change
- netconf-session-start
- netconf-session-end
- netconf-confirmed-commit

These event notifications used within the 'NETCONF' stream are accessible to clients via the subscription mechanism described in [RFC5277].

This document introduces NMDA specific extension which allows a client to receive 3 notifications for additional common system events as follows:

intended-apply-start: Generated when a server with network management protocol support detects that configuration applied

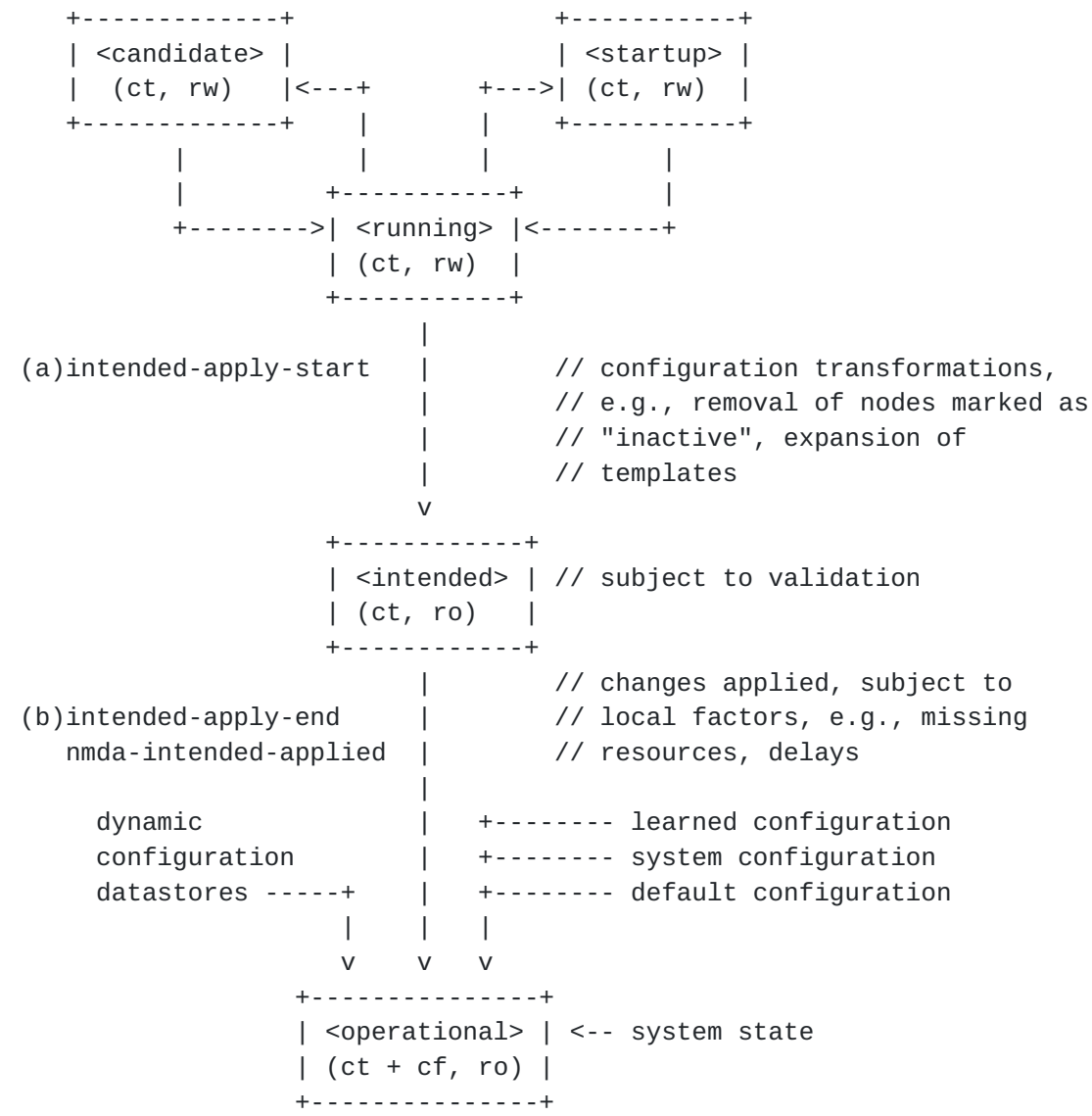
from intended has started. The applied-intended-start starts at the time when configuration is succesfully committed to <running>, i.e., the confirmed-commit procedure has been completed.A server MAY optionally generate this event for both NETCONF session and non-NETCONF management sessions. In case two commits are detected, if the second commit is confirmed commit, the commit and confirmed commit should be treated as the same commit, i.e., intended-apply-start commit should not be sent for persistent confirmed commit. if the the second commit is not confirmed commit, intended-apply-start for the second commit should not be sent until intended-apply-end or nmda-intended-applied for the first commit has been sent.

intended-apply-end: Generated when a server with network management protocol support detects that all or a set of configurations are successfully applied or none of them are applied. Note that configuration applying process of the learned configuration, system-provided configuration, and default values defined by data models is not relevant to the events defined in this document. Unlike the learned configuration, system-provided configuration, the intended configuration is not applied frequently and may be fixed after the configuration applying process. A server MAY optionally generate this event for both NETCONF session and non-NETCONF management sessions.

nmda-intended-applied: Generated when a server with network management protocol support detects that all or a set of configurations are successfully applied or none of them are applied. Occurs at the same as intended-apply-end and Indicates the event and the current state of the intended configuration applying. If the applied-event of the mmda-intended-applied is set to start or compete, the nmda-intended-applied can also used to indicate the start time and end time of the intended configuration applying procedure, i.e., intended-apply-start and intended-apply-end are not needed being sent to the client. In addition, NMDA datastore compare [NMDA-DIFF] should be used to check which part of intended configuration data is applied or which part of intended configuration data is not applied. A server MAY report events for non- NETCONF management sessions (such as RESTCONF,gPRC), using the 'session-id' value of zero.

Editor-Note: If nmda-intended-applied can be used to indicate start time and end time of the intended configuration applying procedure, do we need intended-apply-start and intended-apply-end events?

The following figure shows event notification sequence defined in this document.



These notification messages are accessible to clients via either the subscription mechanism described in [RFC5277] or dynamic subscription mechanism and configured subscription mechanism described in [I-D.ietf-netconf-netconf-event-notifications].

2.2. Data Model Design

The data model is defined in the ietf-nmda-notifications YANG module. Its structure is shown in the following figure. The notation syntax follows [RFC8340].

```
   module: ietf-nmda-notifications
     notifications:
       +---n intended-apply-start
       |  +--ro username           string
       |  +--ro session-id         session-id-or-zero-type
       |  +--ro source-host?       inet:ip-address
       |  +--ro commit-persist-id  string
       +---n intended-apply-end
       |  +--ro username           string
       |  +--ro session-id         session-id-or-zero-type
       |  +--ro source-host?       inet:ip-address
       |  +--ro commit-persist-id  string
       |  +--ro (complete-status)?
       |     +--:(global-errors)
       |     |  +--ro errors
       |     |     +--ro error*
       |     |        +--ro error-type       enumeration
       |     |        +--ro error-tag        string
       |     |        +--ro error-app-tag?   string
       |     |        +--ro error-path?      instance-identifier
       |     |        +--ro error-message?   string
       |     |        +--ro error-info?
       |     +--:(ok)
       |        +--ro ok?            empty
       +---n nmda-intended-applied
          +--ro username               string
          +--ro session-id             session-id-or-zero-type
          +--ro source-host?           inet:ip-address
       |  +--ro commit-persist-id      string
          +--ro applied-event          enumeration
          +--ro (applied-status)?
          |  +--:(global-errors)
          |  |  +--ro errors
          |  |     +--ro error*
          |  |        +--ro error-type       enumeration
          |  |        +--ro error-tag        string
          |  |        +--ro error-app-tag?   string
          |  |        +--ro error-path?      instance-identifier
          |  |        +--ro error-message?   string
          |  |        +--ro error-info?
          |  +--:(ok)
          |     +--ro ok?                    empty
          +--ro fail-applied-target*
             +--ro datastore?   identityref
             +--ro edit-id?     string
             +--ro target?      ypatch:target-resource-offset
             +--ro (applied-status-choice)?
                +--:(errors)
```

```
                  +--ro errors
                     +--ro error*
                        +--ro error-type       enumeration
                        +--ro error-tag        string
                        +--ro error-app-tag?   string
                        +--ro error-path?      instance-identifier
                        +--ro error-message?   string
                        +--ro error-info?
```

The following are examples of a nmda-intended-applied notification message:

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2017-06-16T16:30:59.137045+09:00</eventTime>
  <nmda-intended-applied xmlns="urn:ietf:params:xml:ns:yang:ietf-nmda-
notifications">
    <username>admin</username>
    <session-id>0</session-id>
    <source-host>10.251.93.83</source-host>
    <commit-persist-id>IQ,d4668</commit-persist-id>
    <applied-event>complete<applied-event>
    <errors>
      <error-type>protocol</error-type>
      <error-tag>mis-resource</error-tag>
      <error-path xmlns:ops="https://example.com/ns/ietf-interfaces">\
       \/if:interfaces-state\
      \</error-path>
      <error-message>refer to resources that are not \
      \available or otherwise not physically present.\
      \</error-message>
    </errors>
    <fail-applied-taget>
      <datastore>intended</datastore>
      <edit-id>1<edit-id>
      <target>/ietf-interfaces:interfaces-state</target>
    </fail-applied-target>
    <fail-applied-taget>
      <datastore>intended</datastore>
      <edit-id>1<edit-id>
      <target>/ietf-system:system</target>
    </fail-applied-target>
 </nmda-intended-applied>
</notification>
```

2.3. Relation with NMDA Datastore Compare

NMDA datastore compare [NMDA-DIFF]could be used to check which part of intended configuration data is applied or which part of intended

configuration data is not applied. On the other hand, the event

notification for applied-intended-start can be used to trigger NMDA datastore compare procedure to be started.

2.4. Definitions

This section presents the YANG module defined in this document. This module imports data types from the 'ietf-datastores' module defined in [RFC8342] and 'ietf-inet-types' module defined in [RFC6021].

```
 <CODE BEGINS> file "ietf-nmda-notifications@2018-04-01.yang"
module ietf-nmda-notifications {
  namespace "urn:ietf:params:xml:ns:yang:ietf-nmda-notifications";
  prefix ndn;
  import ietf-datastores {
    prefix ds;
  }
  import ietf-inet-types { prefix inet; }
  import ietf-yang-patch {
    prefix ypatch;
  }
  import ietf-restconf { prefix rc;}
  organization
    "IETF NETMOD (Network Modeling) Working Group";
  contact
    "WG Web:   <http://tools.ietf.org/wg/netmod/>
     WG List:  <mailto:netmod@ietf.org>
     WG Chair: Kent Watsen
               <mailto:kwatsen@juniper.net>
     Editor:   Qin Wu
               <mailto:bill.wu@huawei.com>
     Editor:   Rohit R Ranade
               <mailto:rohitrranade@huawei.com>";
  description
    "This module defines a YANG data model for use with the
     NETCONF and RESTCONF protocol that allows the client to
     receive additional common event notifications related to NMDA.
     Copyright (c) 2012 IETF Trust and the persons identified as
     the document authors.  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
     (http://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 2018-04-01 {
```

```
    description
      "Initial version.";
    reference "RFC xxx: NETCONF Base Notifications for NMDA";
  }
  typedef session-id-or-zero-type {
     type uint32;
     description
       "NETCONF Session Id or Zero to indicate none";
   }

  grouping common-session-parms {
    description
      "Common session parameters to identify a
       management session.";

    leaf username {
      type string;
      mandatory true;
      description
        "Name of the user for the session.";
    }

    leaf session-id {
      type session-id-or-zero-type;
      mandatory true;
      description
        "Identifier of the session.
         A NETCONF session MUST be identified by a non-zero value.
         A non-NETCONF session MAY be identified by the value zero.";
    }

    leaf source-host {
      type inet:ip-address;
      description
        "Address of the remote host for the session.";
    }

   leaf commit-persist-id {
     type string;
     description
       "This parameter is used to identify each commit.
        The value of this parameter is a token that must be given
        in the 'persist-id' parameter of <commit> or
        <cancel-commit> operations in order to confirm or cancel
        the persistent confirmed commit.

        The token should be a random string.";
       reference "RFC 6241, Section 8.3.4.1";
```

```
   }
  }

   notification intended-apply-start {
       description
         " Generated when a server detects that applied configuration from
           intended has started. This event will be sent immediately upon
           any data is written to <running>. A server MAY generate this
           event for both NETCONF session and non-NETCONF management
           sessions.";
       uses common-session-parms;
     } // notification applied-intended-start

   notification intended-apply-end {
       description
         " Generated when a server detects that a applied configuration
          from intended has complete.  A server MAY optionally generate
          this event for both NETCONF session and non-NETCONF management
          sessions.";
       uses common-session-parms;
      choice complete-status {
           description
             "Report global errors or complete success.
              If there is no case selected, then errors
              are reported in the 'edit-status' container.";
           case global-errors {
             uses rc:errors;
             description
               "This container will be present if global errors that
                are unrelated to a specific edit occurred.";
           }
           leaf ok {
             type empty;
             description
               "This leaf will be present if the request succeeded
                and there are no errors reported in the 'edit-status'
                container.";
           }
         }
     } // notification applied-intended-complete

    notification nmda-intended-applied {
      description
        "Generated when a server detects that a
         intended configuration applied event has occurred. Indicates
         the event and the current state of the intended data applying
          procedure in progress.";
      reference "RFC 8342, Section 5";
```

```
      uses common-session-parms;
      leaf applied-event {
        type enumeration {
          enum "start" {
            description
              "The intended data applying procedure has started.";
          }
         enum "transform" {
             description
             " <intended> MAY be updated independently of <running>
               due to configuration transformation.";
         }
         enum "synchronizing" {
             description
             "The data validation results within <intended> is in
              progress of synchronizing to <operatonal>.";
         }
          enum "complete" {
            description
              "The intended data applying procedure has been
               completed.";
          }
        }
        mandatory true;
        description
          "Indicates the event that caused the notification.";
      }
     choice applied-status {
           when "./applied-event = 'complete'";
           description
             "Report global errors or complete success.
              If there is no case selected, then errors
              are reported in the 'edit-status' container.";
           case global-errors {
             uses rc:errors;
             description
               "This container will be present if global errors that
                are unrelated to a specific edit occurred.";
           }
          leaf ok {
            type empty;
            description
              "This leaf will be present if the request succeeded
               and there are no errors reported in the 'edit-status'
               container.";
           }
         }
```

```
      list fail-applied-target {
        leaf datastore {
          type identityref {
            base ds:datastore;
          }
          default "ds:operational";
          description
            "Indicates which datastore has changed or which datastore is
             target of edit-data operation.";
        }
        leaf edit-id {
               type string;
                description
                  "Response status is for the 'edit' list entry
                   with this 'edit-id' value.";
             }
        leaf target {
          type ypatch:target-resource-offset;
          description
            "Topmost node associated with the configuration change.
             A server SHOULD set this object to the node within
             the datastore that is being altered.  A server MAY
             set this object to one of the ancestors of the actual
             node that was changed, or omit this object, if the
             exact node is not known.";
        }
        choice applied-status-choice {
               description
                 "A choice between different types of status
                  responses for each 'edit' entry.";
               case errors {
                 uses rc:errors;
                 description
                   "The server detected errors associated with the
                    edit identified by the same 'edit-id' value.";
               }
             }
        description
          "List for fail applied targets";
      }
    }
  }
```

<CODE ENDS>

3. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH, defined in [RFC6242].

Some of the readable data nodes 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, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/applied-intended-start:

Event type itself indicates that a server may start applying configuration from intended. It may be possible for an attacker to slow down intended validation or update intended independently of running by somehow taking advantage of configuration template transformation.

/applied-intended-complete:

Event type itself indicates that a server may be finished applying configuration from intended. This event could alert an attacker that a datastore may have been altered.

/nmda-data-applied/applied-event:

Indicates the specific applied intended applied event state change that occurred. A value of 'complete' probably indicates that intended data applying procedure has completed.

4. IANA Considerations

This document registers one XML namespace URN in the 'IETF XML registry', following the format defined in [RFC3688]:

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

Registrant Contact: The IESG.

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

This document registers one module name in the 'YANG Module Names' registry, defined in [RFC7950]:

name: ietf-nmda-notifications

prefix: ndn

namespace: urn:ietf:params:xml:ns:yang:ietf-nmda-notifications

RFC: xxxx

5. Acknowledgements

Thanks to Juergen Schoenwaelder, Alex Clemm,Carey Timothy and Andy Berman to review this draft and Thank Xiaojian Ding provide important input to the initial version of this document.

6. Normative References

[I-D.clemm-netmod-nmda-diff]
Clemm, A., Qu, Y., Tantsura, J., and A. Bierman, "Comparison of NMDA datastores", draft-clemm-netmod-nmda-diff-00 (work in progress), June 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>.

[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>.

[RFC6021] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6021, DOI 10.17487/RFC6021, October 2010, <https://www.rfc-editor.org/info/rfc6021>.

[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>.

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

[RFC8072] Bierman, A., Bjorklund, M., and K. Watsen, "YANG Patch Media Type", RFC 8072, DOI 10.17487/RFC8072, February 2017, <https://www.rfc-editor.org/info/rfc8072>.

[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>.

Authors' Addresses

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

Rohit R Ranade
Huawei
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

Email: rohitrranade@huawei.com