

NETMOD Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 31, 2019

Q. Wu
R. Tao
R. Ranade
Huawei
June 29, 2019

NMDA Base Notification for Intent based configuration update
draft-wu-netmod-base-notification-nmda-03

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 datastores of the management system, such as an intended configuration state change in NETCONF server or RESTCONF server, that may impact management applications, especially when a server is managed by multiple clients or management applications. This document defines 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 December 31, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

Internet-Draft

NMDA Base Notification

June 2019

This document is subject to [BCP 78](https://trustee.ietf.org/license-info) 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
1.1.	Terminology	3
2.	NMDA Base Notifications for Intent based configuration Update	3
2.1.	Overview	3
2.2.	Data Model Design	5
2.3.	Relation with NMDA Datastore Compare	6
2.4.	Definitions	7
3.	Security Considerations	13
4.	IANA Considerations	13
5.	Acknowledgements	14
6.	Contributors	14
7.	Normative References	14
Appendix A.	Changes between revisions	15
	Authors' Addresses	15

[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 those additional datastores of the management system, e.g., there are many background system activities (e.g., system internal interactions with hardware, interaction with protocols or other devices) that happen during propagation of a configuration change to the software and hardware components of a system. It is possible that some configuration could not be applied to <operational> due to either remnant Configuration, or missing resource, etc. There is a need for user or an application (configuration) to know the origin of failed

configuration node and the reason why the configuration changes were not applied.

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 [\[RFC8342\]](#). 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:

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

[2.](#) NMDA Base Notifications for Intent based configuration Update

[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:

- o netconf-config-change

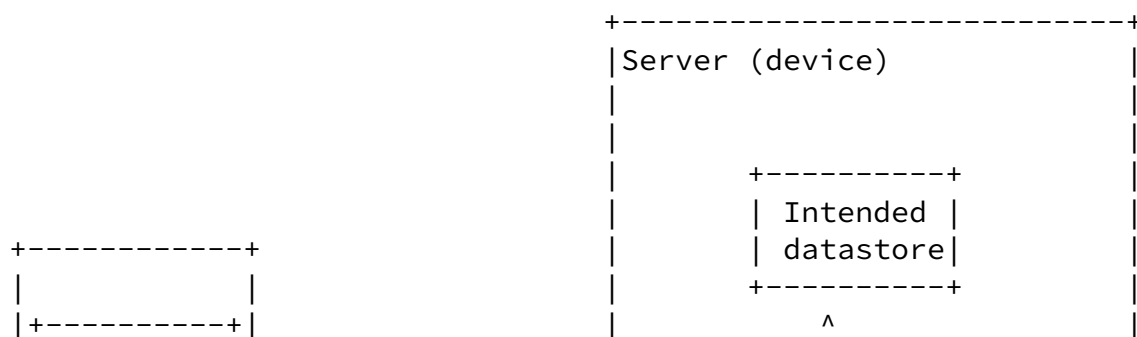
- o netconf-capability-change
- o netconf-session-start
- o netconf-session-end
- o netconf-confirmed-commit

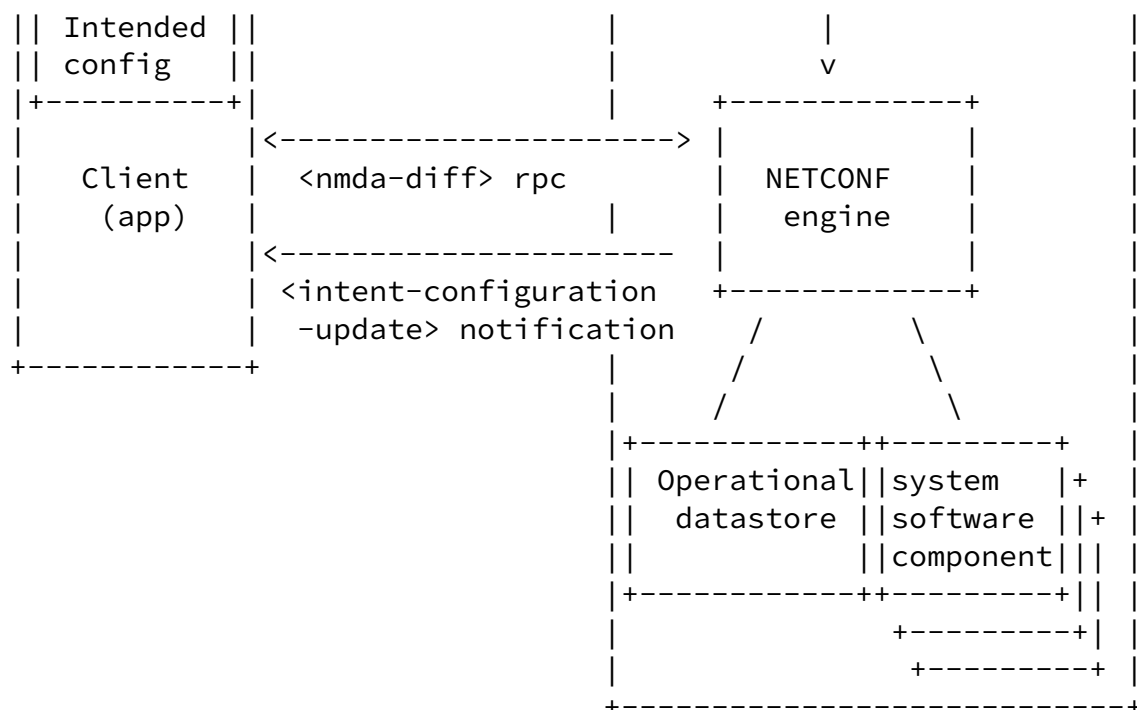
These event notifications used within the 'NETCONF' stream are accessible to clients via the subscription mechanism described in [\[RFC5277\]](#).

This document introduces NMDA specific extension which allows a client to receive 1 notifications for additional common system events as follows:

apply-configuration-updated: Generated when a server with network management protocol support interacts with hardware and detects that a set of configurations are not applied or none of them are not applied. Indicates the event and the current state of the applied configuration or data inconsistency issue between intended data initiated from the client and operational data saved in the server. NMDA datastore compare [\[I-D.ietf-netmod-nmda-diff\]](#) can be used to trigger consistency data check, i.e., indicate the source of configuration node and check which part of configuration data is applied or which part of 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.

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].

```

notifications:
  +---n intent-configuration-update
    +--ro app-tag?          string
    +--ro src-ds?           identityref
    +--ro dst-ds?           identityref
    +--ro (filter-spec)?
      | +--:(subtree-filter)
      | | +--ro subtree-filter? <anydata>
      | +--:(xpath-filter)
      |   +--ro xpath-filter?  yang:xpath1.0 {nc:xpath}?
      --ro apply-result        enumeration

```

```

+--ro fail-applied-object* [edit-id]
  +--ro edit-id      string
  +--ro operation     enumeration
  +--ro object?      ypatch:target-resource-offset
  +--ro value?       <anydata>
  +--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?    <anydata>

```

The following are examples of a apply-configuration-updated notification message:

```

<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2017-06-16T16:30:59.137045+09:00</eventTime>
  <intent-configuration-update xmlns="urn:ietf:params:xml:ns:yang:ietf-nmda-n
    <app-tag>ds--module-a</app-tag>
    <datastore>intended</datastore>
    <datastore>operational</datastore>
    <fail-applied-object>
      <edit-id>1</edit-id>
      <operation>merge</operation>
      <target>/ietf-interfaces:interfaces-state</target>
      <value>
        <interfaces-state xmlns="http://foo.com/ietf-interfaces">
          <interface>
            <name>eth0</name>
            <oper-status>down</oper-status>

```

```

        </interface>
    </interfaces-state>
</value>
</fail-applied-object>
<fail-applied-object>
    <edit-id>2</edit-id>
    <target>/ietf-system:system</target>
    <errors>
        <error-type>protocol</error-type>
        <error-tag>mis-resource</error-tag>
        <error-path xmlns:ops="https://example.com/ns/ietf-system">\
            \if:interfaces-state\
            \</error-path>
        <error-message>refer to resources that are not \
            \available or otherwise not physically present.\
            \</error-message>
    </errors>
</fail-applied-object>
</intent-configuration-update>
</notification>

```

[2.3.](#) Relation with NMDA Datastore Compare

NMDA datastore compare [[I-D.ietf-netmod-nmda-diff](#)] could be used to check which part of configuration data is applied or which part of configuration data is not applied, e.g., If a client creates an interface "et-0/0/0" but the interface does not physically exist at this point, the interface will appear in <intended> but does not exist in the <operational>. By comparing configuration difference between <intended> and <operational>, the interface that is not applied can be sorted out. Unlike [[I-D.ietf-netmod-nmda-diff](#)], the notification message only focuses on the configuration data that is not applied and the reason why the configuration changes

were not applied. Also system internal interactions with hardware is needed within the server to make sure fail applied object is caused by mis-resource or remnant Configuration, etc.

[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@2019-06-19.yang"
module ietf-nmda-notifications {
  yang-version 1.1;
  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-types { prefix yang;}
  import ietf-yang-patch {
    prefix ypatch;
  }
  import ietf-netconf {
    prefix nc;
  }
  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>

    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
```

(<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 2019-06-19 {
  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";
}
feature error-info {
  description
    "This feature must
    also be enabled for that session if error-info
    can be advertised by the server. Otherwise,
    this feature must not be enabled.";
}
grouping common-session-parms {
  description
    "Common session parameters to identify a
    management session or internal interaction
    on a set of configuration data.";

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

  leaf session-id {
    type session-id-or-zero-type;
    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 app-tag {
    type string;
```

```
    description
      "The application tag used to identify the managment session
      or internal interaction on a set of configuration data.";
  }
}

notification intent-configuration-updated {
  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 src-ds {
    type identityref {
      base ds:datastore;
    }
    description
      "Indicates which datastore is
      source of edit-data operation.";
  }
  leaf dst-ds {
    type identityref {
      base ds:datastore;
    }
    description
      "Indicates which datastore is
      target of edit-data operation.";
  }
  choice filter-spec {
    description
      "The content filter specification for this request.";
    anydata subtree-filter {
      description
        "This parameter identifies the portions of the
        target datastore to retrieve.";
      reference
        "RFC 6241: Network Configuration Protocol, Section 6.";
    }
    leaf xpath-filter {
      if-feature nc:xpath;
      type yang:xpath1.0;
      description
        "This parameter contains an XPath expression identifying
```

the portions of the target datastore to retrieve.

If the expression returns a node-set, all nodes in the node-set are selected by the filter. Otherwise, if the expression does not return a node-set, then the get-data operation fails.

The expression is evaluated in the following XPath context:

- o The set of namespace declarations are those in scope on the 'xpath-filter' leaf element.
- o The set of variable bindings is empty.
- o The function library is the core function library, and the XPath functions defined in [section 10 in RFC 7950](#).
- o The context node is the root node of the target datastore.";

```
    }
  }
  leaf apply-result {
    type enumeration {
      enum "partial-fail" {
        description
          "A set of configuration data is not applied.";
      }
      enum "fail" {
        description
          "None of configuration data is applied.";
      }
      enum "sucess" {
        description
          "All configuration data is applied.";
      }
    }
  }
  description
    "Configuration data apply result.";
}
```

```

list fail-applied-object {
  when "../apply-result = 'partial-fail'";
  key edit-id;
  ordered-by user;
  leaf edit-id {
    type string;
    description
      "Response status is for the 'edit' list entry
       with this 'edit-id' value.";
  }
}

```

```

}
leaf operation {
  type enumeration {
    enum create {
      description
        "The target data node is created using the supplied
         value, only if it does not already exist. The
         'target' leaf identifies the data node to be
         created, not the parent data node.";
    }
    enum delete {
      description
        "Delete the target node, only if the data resource
         currently exists; otherwise, return an error.";
    }
    enum insert {
      description
        "Insert the supplied value into a user-ordered
         list or leaf-list entry. The target node must
         represent a new data resource. If the 'where'
         parameter is set to 'before' or 'after', then
         the 'point' parameter identifies the insertion
         point for the target node.";
    }
    enum merge {
      description
        "The supplied value is merged with the target data
         node.";
    }
    enum move {
      description

```

```

    "Move the target node. Reorder a user-ordered
    list or leaf-list. The target node must represent
    an existing data resource. If the 'where' parameter
    is set to 'before' or 'after', then the 'point'
    parameter identifies the insertion point to move
    the target node.";
}
enum replace {
    description
        "The supplied value is used to replace the target
        data node.";
}
enum remove {
    description
        "Delete the target node if it currently exists.";
}

```

```

    }
    mandatory true;
    description
        "The datastore operation requested for the associated
        'edit' entry.";
}
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.";
}
anydata value {
    description
        "Value used for this edit operation. The anydata 'value'
        contains the target resource associated with the
        'target' leaf.

        For example, suppose the target node is a YANG container
        named foo:

```

```

    container foo {
        leaf a { type string; }
        leaf b { type int32; }
    }

```

The 'value' node contains one instance of foo:

```

    <value>
        <foo xmlns='example-foo-namespace'>
            <a>some value</a>
            <b>42</b>
        </foo>
    </value>
";
}
uses rc:errors {if-feature error-info;}
description
    "List for fail applied objects that is not applied. ";
}
}
}
<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:

/intent-configuration-update:

Event type itself indicates that intent based configuration has been updated. This event could alert an attacker that a datastore may have been altered.

/intent-configuration-updated/apply-result:

Indicates the specific intent based configuration update event state change that occurred. A value of 'success' probably indicates that intent based configuration has been applied successfully.

[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

Wu, et al.

Expires December 31, 2019

[Page 13]

Internet-Draft

NMDA Base Notification

June 2019

[5.](#) Acknowledgements

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

[6.](#) Contributors

Chong Feng
Huawei
Email: frank.fengchong@huawei.com

7. Normative References

- [I-D.ietf-netmod-nmda-diff]
Clemm, A., Qu, Y., Tantsura, J., and A. Bierman,
"Comparison of NMDA datastores", [draft-ietf-netmod-nmda-diff-01](#) (work in progress), May 2019.
- [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>>.

Appendix A. Changes between revisions

v01 - v03

- o Change notification name into intent-configuration update.
- o Change title into NMDA Base event for intent based configuration update.
- o Clarify the usage of NMDA base event and relation with NMDA diff work.

v01 - v00

- o Add application tag support and use additional parameters to identify management session.
- o Remove apply-intended-start and apply-intended-end two notifications since they are not needed based on discussion.

Authors' Addresses

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

Email: bill.wu@huawei.com

Ran Tao
Huawei

Email: taoran20@huawei.com

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

Email: rohitrranade@huawei.com

