Workgroup: NETCONF
Internet-Draft:
draft-ietf-netconf-notification-
capabilities-20
Published: 10 October 2021
Intended Status: Standards Track
Expires: 13 April 2022
Authors: B. Lengyel A. Clemm B. Claise
Ericsson Futurewei Huawei

YANG Modules describing Capabilities for Systems and Datastore Update Notifications

Abstract

This document defines two YANG modules,"ietf-system-capabilities" and "ietf-notification-capabilities".

The module "ietf-system-capabilities" provides a placeholder structure that can be used to discover YANG related system capabilities for servers. The module can be used to report capability information from the server at run time or at implementation time, by making use of the YANG Instance Data File Format.

The module "ietf-notification-capabilities" augments "ietf-system-capabilities" to specify capabilities related to Subscription to YANG Notifications for Datastore Updates.

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 13 April 2022.

Copyright Notice

Table of Contents

1. Introduction

Servers and/or publishers often have capabilities, values describing operational behavior, that need to be conveyed to clients, which is enabled by the YANG modules described in this document.

There is a need to publish this capability information as it is part of the API contract between the server and client. Examples include maximum size of data that can be stored or transferred, information about counters (whether a node supports "on-change" telemetry), etc. Such capabilities are often dependent on a vendor's implementation or the available resources at deployment. Many such capabilities are specific to the complete system, individual YANG datastores [RFC8342], specific parts of the YANG schema, or even individual

data nodes. It is a goal of this document to provide a common way of representing such capabilities in a format that is:

- vendor independent
- machine-readable
- available in an identical format both at implementation time and at run time.

Implementation-time information is needed by Network Management System (NMS) implementers. An NMS implementation that supports notifications needs the information about a system's capability so it can send "on-change" notifications. If the information is not documented in a way that is readily available to the NMS designer, but only as instance data from the network node once it is deployed, the NMS implementation will be delayed, because it has to wait for the network node to be ready. In addition, the assumption that all NMS implementers will have a correctly configured network node available to retrieve data from is an expensive proposition and may not always hold. (An NMS may need to be able to handle many dozens of network node types.) Often a fully functional NMS is a requirement for introducing a new network node type into a network, so delaying NMS readiness effectively also delays the time at which a new network node type can be introduced into the network.

Implementation-time information is needed by system integrators. When introducing a network node type into their network, operators often need to integrate the node type into their own management system. The NMS may have management functions that depend on "on-change" notifications. The network operators need to plan their management practices and NMS implementation before they decide to buy the specific network node type. Moreover, the decision to buy the node type sometimes depends on these management possibilities.

Run-time capability information is needed:

- for any "purely model driven" application, e.g., a NETCONF-browser. Such applications depend on reading models and capabilities at run time to support all the publisher's available functionality.
- in case the capability might change during run time e.g., due to licensing, HW constraints etc.
- to check that capability information provided earlier, at implementation time, is what the publisher has implemented.

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 terms "YANG-Push", "on-change subscription" and "periodic subscription" are used as defined in [RFC8641].

The terms "subscriber", "publisher" and "receiver" are used as defined in [RFC8639].

The term "server" is used as defined in [RFC8342].

The terms "YANG instance data file format", "instance data", and "instance data set" are used as defined in [I-D.ietf-netmod-yang-instance-file-format].

In addition, this document defines the following terms:

"Implementation-time information": Information about the server's behavior that is made available during the implementation of the server, available from a source other than a running server.

"Run-time information": Information about the server's behavior that is available from the running server via management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040].

2. Providing System Capability Information

Capability information is represented by instance-data based on one or more "capability defining YANG modules". This allows a user to discover capabilities both at implementation time and at run time.

* For the implementation-time use case: Capabilities SHOULD be provided by the implementer as YANG instance data files complying to [I-D.ietf-netmod-yang-instance-file-format]. When provided, the file MUST be available already at implementation time, retrievable in a way that does not depend on a live network node. E.g., download from product website.

* For the run-time use case: Capabilities SHOULD be available via NETCONF [RFC6241] or RESTCONF [RFC8040] from the live server (implementing the publisher) during run time. Implementations that support changing these capabilities at run time SHOULD support "on-change" notifications about the system-capabilities container.

The module "ietf-system-capabilities" provides a placeholder structure to be used to specify any YANG related system capability.

The module "ietf-notification-capabilities" is defined to allow a publisher to specify capabilities related to "Subscription to YANG Notifications for Datastore Updates" [RFC8641], also known as YANG-Push, augmenting "ietf-system-capabilities".

The YANG data models in this document conform to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

3. Providing YANG-Push Notification Capabilities Information

A specific case is the need to specify capabilities in the YANG-Push functionality. As defined in [RFC8641] a publisher may allow subscribers to subscribe to updates from a datastore and will subsequently push such update notifications to the receiver. Notifications may be sent periodically or "on-change" (more or less immediately after each change).

A publisher supporting YANG-Push has a number of capabilities defined in [RFC8641] that are often determined during the implementation of the publisher. These include:

* Supported (reporting) periods for "periodic" subscriptions

* Maximum number of objects that can be sent in an update

* The set of datastores or data nodes for which "periodic" notification is supported.

Additional capabilities if the optional "on-change" feature is supported include:

* Supported dampening periods for "on-change" subscriptions

* The set of datastores or data nodes for which "on-change" notification is supported

Publishers might have some capabilities (or limitations) to document. For example, how many update notifications and how many datastore node updates they can send out in a certain time-period. Other publishers might not support "periodic" subscriptions to all datastores. In some cases, a publisher supporting "on-change" notifications will not be able to push updates for some object types "on-change". Reasons for this might be that the value of the datastore node changes frequently (e.g., in-octets counter), that small object changes are frequent and irrelevant to the receiver (e.g., a temperature gauge changing 0.1 degrees within a predetermined and acceptable range), or that the implementation is

not capable of on-change notification for a particular object. In all those cases, it will be important for subscriber applications to have a way to identify which objects "on-change" notifications are supported and for which ones not.

Faced with the reality that support for "on-change" notification does not mean that such notifications will be sent for any specific data node, subscriber/management applications can not rely on the "on-change" functionality unless the subscriber has some means to identify which objects "on-change" notifications are supported for. YANG models are meant to be used as an interface contract. Without identification of the data nodes actually supporting "on-change", this contract would be incomplete.

Clients of a server (and subscribers to a publisher, as subscribers are also clients) need a method to gather capability information.

4. System Capabilities Model

The module "ietf-system-capabilities" is defined to provide a structure that can be used to discover (as read-only operational state) any YANG related system capability.

This module itself does not contain any capabilities; it provides augmentation points for capabilities to be defined in subsequent YANG modules. The ietf-system-capabilies is used by other modules to augment in specific capability information. Every set of such capabilities MUST be wrapped in a container under the augment statement to cleanly separate different groups of capabilities. These "wrapper containers" SHALL be augmented in at /sysc:system-capabilities and /sysc:system-capabilities/sysc:datastore-capabilities/sysc:per-node-capabilities.

Capability values can be specified on system level, datastore level (by selecting all nodes in the datastore) or for specific data nodes of a specific datastore (and their contained sub-trees). Capability values specified for a specific datastore or node-set override values specified on the system level.

Note: The solution is usable for both NMDA and non-NMDA systems. For non-NMDA servers "config false" data is considered as if it were part of the running datastore.

4.1. Tree Diagram

The following tree diagram [RFC8340] provides an overview of the data model.

```
module: ietf-system-capabilities
  +--ro system-capabilities
     +--ro datastore-capabilities* [datastore]
        +--ro datastore         -> /yanglib:yang-library/datastore/name
        +--ro per-node-capabilities* []
           +--ro (node-selection)?
              +--:(node-selector)
                 +--ro node-selector?   nacm:node-instance-identifier
```

4.2. YANG Module

This YANG module imports typedefs from [RFC8341] and a reference path from [RFC8525].

```
<CODE BEGINS> file "ietf-system-capabilities@2021-04-02.yang"


module ietf-system-capabilities {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-system-capabilities";
  prefix sysc;

  import ietf-netconf-acm {
    prefix nacm;
    reference
      "RFC 8341: Network Configuration Access Control Model";
  }
  import ietf-yang-library {
    prefix yanglib;
    description
      "This module requires ietf-yang-library to be implemented.
       Revision 2019-01-04 or a revision derived from it
       is REQUIRED.";
    reference
      "RFC8525: YANG Library";
  }

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

     Editor:   Balazs Lengyel
               <mailto:balazs.lengyel@ericsson.com>";
  description
    "This module specifies a structure to specify system
     capabilities for a server or a publisher. System capabilities
     may include capabilities of a NETCONF or RESTCONF server or a
     notification publisher.

     This module does not contain any specific capabilities, it only
     provides a structure where containers containing the actual
     capabilities are augmented in.

     Capability values can be specified on system level,
     datastore level (by selecting all nodes in the datastore) or
     for specific data nodes of a specific datastore (and their
     contained sub-trees).
     Capability values specified for a specific datastore or
     node-set override values specified on the system/publisher level.

         The same grouping MUST be used to define hierarchical capabilities
     supported both at the system level and datastore/data node level.
```

```
      To find a capability value for a specific data node in a
      specific datastore the user SHALL:

      1) search for a datastore-capabilities list entry for
      the specific datastore. When stating a specific capability, the
      relative path for any specific capability must be the same
      under the system-capabilities container and under the
      per-node-capabilities list.

      2) If the datastore entry is found within that entry, process all
      per-node-capabilities entries in the order they appear in the list.
      The first entry that specifies the specific capability and has a
      node-selector selecting the specific data node defines the
      capability value.

      3) If the capability value is not found above and the specific
      capability is specified under the system-capabilities container
      (outside the datastore-capabilities list), this value shall be
      used.

      4) If no values are found in the previous steps, the
      system/publisher is not capable of providing a value. Possible
      reasons are: it is unknown, the capability is changing for some
      reason, there is no specified limit, etc. In this case the
      system's behavior is unspecified.

      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 (RFC 2119)
      (RFC 8174) when, and only when, they appear in all
      capitals, as shown here.

      Copyright (c) 2021 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
      (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
      for full legal notices.";

  // RFC Ed.: replace XXXX with actual RFC number and remove this
  // note.
```

```
revision 2021-04-02 {
  description
    "Initial version
     NOTE TO RFC EDITOR:
     (1)Please replace the above revision date to
       the date of RFC publication when published.
     (2) Please replace the date in the file name
       (ietf-system-capabilities@2021-04-02.yang)
       to the date of RFC publication.
     (3) Please replace the following reference
       with RFC number when published
       (i.e. RFC xxxx).";
  reference
    "RFC XXXX: YANG Modules describing Capabilities for Systems
     and Datastore Update Notifications";
}

container system-capabilities {
  config false;
  description
    "System capabilities.
     Capability values specified here at the system level
     are valid for all datastores and are used when the
     capability is not specified on the datastore level
     or for specific data nodes.";

  /*
   * "Augmentation point for system level capabilities."
   */

  list datastore-capabilities {
    key "datastore";
    description
      "Capabilities values per datastore.

       For non-NMDA servers/publishers 'config false' data is
       considered as if it was part of the running datastore.";
    leaf datastore {
      type leafref {
        path
          "/yanglib:yang-library/yanglib:datastore/yanglib:name";
      }
      description
        "The datastore for which capabilities are defined.
         Only one specific datastore can be specified
         e.g., ds:conventional, which represents a set of
         configuration datastores, must not be used.";
    }
```

```
      list per-node-capabilities {
        description
          "Each list entry specifies capabilities for the selected
           data nodes. The same capabilities apply for the data nodes
           in the subtree below the selected nodes.

           The system SHALL order the entries according to their
           precedence. The order of the entries MUST NOT change unless
           the underlying capabilities also change.

           Note that the longest patch matching can be achieved
           by ordering more specific matches before less
           specific ones.";
        choice node-selection {
          description
            "A method to select some or all nodes within a datastore.";
          leaf node-selector {
            type nacm:node-instance-identifier;
            description
              "Selects the data nodes for which capabilities are
               specified. The special value '/' denotes all data nodes
               in the datastore, consistent with the path leaf node on
               page 41 [RFC8341].";
            reference
              "RFC 8341: Network Configuration Access Control Model";
          }
        }
        /*
         * "Augmentation point for datastore or data node level
         * capabilities."
         */
      }
    }
  }
}

<CODE ENDS>
```


```
   <CODE ENDS>
```


5. Notification Capabilities Model

The YANG module "ietf-notification-capabilities" provides YANG-Push related capability information.

5.1. Tree Diagram

The following tree diagram [RFC8340] provides an overview of the data model.

```
module: ietf-notification-capabilities
  augment /sysc:system-capabilities:
    +--ro subscription-capabilities
       +--ro max-nodes-per-update?               uint32
       +--ro periodic-notifications-supported?   notification-support
       +--ro (update-period)?
       |  +--:(minimum-update-period)
       |  |  +--ro minimum-update-period?        uint32
       |  +--:(supported-update-period)
       |     +--ro supported-update-period*      uint32
       +--ro on-change-supported?                notification-support
       |       {yp:on-change}?
       +--ro minimum-dampening-period?           uint32
       |       {yp:on-change}?
       +--ro supported-excluded-change-type*     union
               {yp:on-change}?
  augment /sysc:system-capabilities/sysc:datastore-capabilities
            /sysc:per-node-capabilities:
    +--ro subscription-capabilities
       +--ro max-nodes-per-update?               uint32
       +--ro periodic-notifications-supported?   notification-support
       +--ro (update-period)?
       |  +--:(minimum-update-period)
       |  |  +--ro minimum-update-period?        uint32
       |  +--:(supported-update-period)
       |     +--ro supported-update-period*      uint32
       +--ro on-change-supported?                notification-support
       |       {yp:on-change}?
       +--ro minimum-dampening-period?           uint32
       |       {yp:on-change}?
       +--ro supported-excluded-change-type*     union
               {yp:on-change}?
```

5.2. YANG Module

This YANG module imports a feature and typedefs from [RFC8641] and also imports the "ietf-system-capabilities" specified in this document.

```
<CODE BEGINS> file "ietf-notification-capabilities@2021-04-02.yang"


module ietf-notification-capabilities {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-notification-capabilities";
  prefix notc;

  import ietf-yang-push {
    prefix yp;
    description
      "This module requires ietf-yang-push to be implemented.";
    reference
      "RFC 8641: Subscription to YANG Notifications for Datastore
        Updates";
  }
  import ietf-system-capabilities {
    prefix sysc;
    description
      "This module requires ietf-system-capabilities to be
       implemented.";
    reference
      "RFC XXXX: YANG Modules describing Capabilities for Systems
       and Datastore Update Notifications";
  }

  // RFC Ed.: replace the above XXXX with actual RFC number
  // and remove this note.

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

     Editor:   Balazs Lengyel
               <mailto:balazs.lengyel@ericsson.com>";
  description
    "This module specifies YANG-Push [RFC 8641] related publisher
     capabilities.

     The module contains:

     - specification of which data nodes support 'on-change' or
     'periodic' notifications.

     - capabilities related to the throughput of notification data
     that the publisher can support. (Note that for a specific
     subscription, the publisher MAY allow only longer periods
```

```
      or smaller updates depending on, e.g., actual load conditions.)

      Capability values can be specified on system/publisher level,
      datastore level or for specific data nodes of a specific
      datastore (and their contained sub-trees), as defined in the
      ietf-system-capabilities module.

      If different data nodes covered by a single subscription
      have different values for a specific capability, then using
      values that are only acceptable for some of these data nodes,
      but not for others, may result in the rejection of the
      subscription.

      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 (RFC 2119)
      (RFC 8174) when, and only when, they appear in all
      capitals, as shown here.

      Copyright (c) 2021 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
      (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
      for full legal notices.";

   // RFC Ed.: replace XXXX with actual RFC number and remove this
   // note.

   revision 2021-04-02 {
     description
       "Initial version";
     reference
       "RFC XXXX: YANG Modules describing Capabilities for Systems
        and Datastore Update Notifications";
   }

   // RFC Ed.: replace XXXX with actual RFC number and remove this
   // note.

   grouping subscription-capabilities {
     description
```

```
  "Capabilities related to YANG-Push subscriptions
   and notifications";
container subscription-capabilities {
  description
    "Capabilities related to YANG-Push subscriptions
     and notifications";
  typedef notification-support {
    type bits {
      bit config-changes {
        description
          "The publisher is capable of sending
           notifications for 'config true' nodes for the
           relevant scope and subscription type.";
      }
      bit state-changes {
        description
          "The publisher is capable of sending
           notifications for 'config false' nodes for the relevant
           scope and subscription type.";
      }
    }
    description
      "Type for defining whether 'on-change' or
       'periodic' notifications are supported for all data nodes,
       'config false' data nodes, 'config true' data nodes, or
       no data nodes.

       If the bit config-changes or state-changes is set
       for a datastore, or a set of nodes that does not contain
       nodes with the indicated config value, this has no
       effect, as if no support was declared. E.g. indicating
       support for state-changes for a candidate datastore has
       no effect.";
  }

  leaf max-nodes-per-update {
    type uint32 {
      range "1..max";
    }
    description
      "Maximum number of data nodes that can be sent
       in an update. The publisher MAY support more data nodes,
       but SHOULD support at least this number.

       May be used to avoid the 'update-too-big' error
       during subscription.";
    reference
      "RFC 8641: Subscription to YANG Notifications for Datastore
       Updates, the 'update-too-big' error/identity";
```

```
}
leaf periodic-notifications-supported {
  type notification-support;
  description
    "Specifies whether the publisher is capable of
     sending 'periodic' notifications for the selected
     data nodes including any subtrees that may exist
     below them.";
  reference
    "RFC 8641: Subscription to YANG Notifications for Datastore
     Updates, 'periodic' subscription concept";
}
choice update-period {
  description
    "Supported update period value or values for
     'periodic' subscriptions.";
  leaf minimum-update-period {
    type uint32;
    units "centiseconds";
    description
      "Indicates the minimal update period that is
       supported for a 'periodic' subscription.

       A subscription request to the selected data nodes with
       a smaller period than what this leaf specifies is
       likely to result in a 'period-unsupported' error.";
    reference
      "RFC 8641: Subscription to YANG Notifications for Datastore
       Updates, the period leaf in the ietf-yang-push YANG
       module";
  }
  leaf-list supported-update-period {
    type uint32;
    units "centiseconds";
    description
      "Supported update period values for a 'periodic'
       subscription.

       A subscription request to the selected data nodes with a
       period not included in the leaf-list will result in a
       'period-unsupported' error.";
    reference
      "RFC 8641: Subscription to YANG Notifications for Datastore
       Updates, the period leaf in the ietf-yang-push YANG
       module";
  }
}
leaf on-change-supported {
  if-feature "yp:on-change";
```

```
  type notification-support;
  description
    "Specifies whether the publisher is capable of
     sending 'on-change' notifications for the selected
     data nodes and the subtree below them.";
  reference
    "RFC 8641: Subscription to YANG Notifications for Datastore
     Updates, on-change concept";
}
leaf minimum-dampening-period {
  if-feature "yp:on-change";
  type uint32;
  units "centiseconds";
  description
    "The minimum dampening-period supported for 'on-change'
     subscriptions for the selected data nodes.

     If this value is present and greater than zero,
     that implies dampening is mandatory.";
  reference
    "RFC 8641: Subscription to YANG Notifications for
     Datastore Updates, the dampening-period leaf in the
     ietf-yang-push YANG module";
}
leaf-list supported-excluded-change-type {
  if-feature "yp:on-change";
  type union {
    type enumeration {
      enum none {
        value -2;
        description
          "None of the change types can be excluded.";
      }
      enum all {
        value -1;
        description
          "Any combination of change types can be excluded.";
      }
    }
    type yp:change-type;
  }
  description
    "The change types that can be excluded in
     YANG-Push subscriptions for the selected data nodes.";
  reference
    "RFC 8641: Subscription to YANG Notifications for Datastore
     Updates, the change-type typedef in the ietf-yang-push
     YANG module";
}
```

```
    }
  }

  augment "/sysc:system-capabilities" {
    description
      "Add system level capabilities";
    uses subscription-capabilities {
      refine
        "subscription-capabilities/supported-excluded-change-type" {
        default "none";
      }
    }
  }

  augment "/sysc:system-capabilities/sysc:datastore-capabilities"
        + "/sysc:per-node-capabilities" {
    description
      "Add datastore and node level capabilities";
    uses subscription-capabilities {
      refine
        "subscription-capabilities/supported-excluded-change-type" {
        default "none";
      }
    }
  }
}

<CODE ENDS>
```


<CODE ENDS>

6. Security Considerations

The YANG modules specified in this document define a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040] or as YANG instance data. 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 [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] 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.

All protocol-accessible data nodes in augmented modules are read-only and cannot be modified. The data in these modules is not

security sensitive. Access control may be configured, to avoid exposing the read-only data.

When that data is in file format, data should be protected against modification or unauthorized access using normal file handling mechanisms. The data in file format also inherits all the security considerations of [I-D.ietf-netmod-yang-instance-file-format] which has additional considerations about read protections; and distinguishes between data at rest and in motion.

7. IANA Considerations

7.1. The IETF XML Registry

This document registers two URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested:

URI: urn:ietf:params:xml:ns:yang:ietf-system-capabilities
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-notification-capabilities
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

7.2. The YANG Module Names Registry

This document registers two YANG modules in the YANG Module Names registry [RFC6020]. Following the format in [RFC6020], the following registrations are requested:

```
name:       ietf-system-capabilities
namespace:  urn:ietf:params:xml:ns:yang:ietf-system-capabilities
prefix:     sysc
reference:  RFC XXXX

name:       ietf-notification-capabilities
namespace:
     urn:ietf:params:xml:ns:yang:ietf-notification-capabilities
prefix:     notc
reference:  RFC XXXX
```

8. Acknowledgments

For their valuable comments, discussions, and feedback, we wish to acknowledge Andy Bierman, Juergen Schoenwaelder, Rob Wilton, Kent Watsen, Eric Voit, Joe Clarke, Martin Bjorklund, Ladislav Lhotka, Qin Wu, Mahesh Jethanandani, Ran Tao, Reshad Rahman and other members of the Netmod WG.

9. References

9.1. Normative References

[I-D.ietf-netmod-yang-instance-file-format] Lengyel, B. and B. Claise, "YANG Instance Data File Format", Work in Progress, Internet-Draft, draft-ietf-netmod-yang-instance-file-format-19, 16 September 2021, <https://www.ietf.org/archive/id/draft-ietf-netmod-yang-instance-file-format-19.txt>.

[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>.

[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>.

[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>.

[RFC8525] Bierman, A., Bjorklund, M., Schoenwaelder, J., Watsen, K., and R. Wilton, "YANG Library", RFC 8525, DOI 10.17487/RFC8525, March 2019, <https://www.rfc-editor.org/info/rfc8525>.

[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>.

9.2. Informative References

[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>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[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>.

[RFC8792] Watsen, K., Auerswald, E., Farrel, A., and Q. Wu, "Handling Long Lines in Content of Internet-Drafts and RFCs", RFC 8792, DOI 10.17487/RFC8792, June 2020, <https://www.rfc-editor.org/info/rfc8792>.

Appendix A. Instance data example #1

The following examples use artwork folding [RFC8792] for better formatting.

The following instance data example describes the notification capabilities of a hypothetical "acme-router". The router implements the running and operational datastores. Every change can be reported "on-change" from the running datastore, but only "config false" nodes and some "config false" data from the operational datastore. Interface statistics are not reported "on-change", only two important counters. Datastore subscription capabilities are not reported "on-change", as they never change on the acme-router during run time.

```
========== NOTE: '\' line wrapping per RFC 8792) ===========

<?xml version="1.0" encoding="UTF-8"?>
<instance-data-set xmlns=\
    "urn:ietf:params:xml:ns:yang:ietf-yang-instance-data">
  <name>acme-router-notification-capabilities</name>
  <content-schema>
    <module>ietf-system-capabilities@2021-04-02</module>
    <module>ietf-notification-capabilities@2021-04-02</module>
  </content-schema>
  <description>Defines the notification capabilities of an acme-router.
    The router only has running and operational datastores.
    Every change can be reported on-change from the running
    datastore, but only "config false" nodes and some "config
    false" data from the operational datastore. Statistics are
    not reported on-change except for two important counters,
    where a small dampening period is mandated.
  </description>
  <content-data>
    <system-capabilities \
     xmlns="urn:ietf:params:xml:ns:yang:ietf-system-capabilities" \
     xmlns:notc=\
       "urn:ietf:params:xml:ns:yang:ietf-notification-capabilities" \
     xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
     <notc:subscription-capabilities>
      <notc:minimum-update-period>500</notc:minimum-update-period>
      <notc:max-nodes-per-update>2000</notc:max-nodes-per-update>
      <notc:minimum-dampening-period>100</notc:minimum-dampening-period>
      <notc:periodic-notifications-supported>\
         config-changes state-changes\
      </notc:periodic-notifications-supported>
      <notc:on-change-supported>\
         config-changes state-changes\
      </notc:on-change-supported>
      <notc:supported-excluded-change-type>\
         all\
      </notc:supported-excluded-change-type>
     </notc:subscription-capabilities>
      <datastore-capabilities>
        <datastore>ds:operational</datastore>
        <per-node-capabilities>
          <node-selector>\
              /if:interfaces/if:interface[if:name='lo']\
          </node-selector>
          <notc:subscription-capabilities>
            <notc:on-change-supported/>
            <notc:periodic-notifications-supported/>
          </notc:subscription-capabilities>
        </per-node-capabilities>
```

```
        <per-node-capabilities>
          <node-selector>\
              /if:interfaces/if:interface/if:statistics/if:in-octets\
          </node-selector>
          <notc:subscription-capabilities>
            <notc:minimum-dampening-period>10
              </notc:minimum-dampening-period>
            <notc:on-change-supported>\
              state-changes\
            </notc:on-change-supported>
          </notc:subscription-capabilities>
        </per-node-capabilities>
        <per-node-capabilities>
          <node-selector>\
              /if:interfaces/if:interface/if:statistics/if:out-octets\
          </node-selector>
          <notc:subscription-capabilities>
            <notc:minimum-dampening-period>10
              </notc:minimum-dampening-period>
            <notc:on-change-supported>\
              state-changes\
            </notc:on-change-supported>
          </notc:subscription-capabilities>
        </per-node-capabilities>
        <per-node-capabilities>
          <node-selector>\
              /if:interfaces/if:interface/if:statistics\
          </node-selector>
          <notc:subscription-capabilities>
            <notc:on-change-supported/>
          </notc:subscription-capabilities>
        </per-node-capabilities>
      </datastore-capabilities>
    </system-capabilities>
  </content-data>
</instance-data-set>
```

Figure 1: Notification Capabilities with data node specific settings

Appendix B. Instance data example #2

The following examples use artwork folding [RFC8792] for better formatting.

The following instance data example describes the notification capabilities of a hypothetical "acme-switch". The switch implements the running, candidate and operational datastores. Every change can be reported "on-change" from the running datastore, nothing from the candidate datastore and all "config false" data from the operational datastore. "periodic" subscriptions are supported for running and operational, but not for candidate datastores.

```
========== NOTE: '\' line wrapping per RFC 8792) ===========

<?xml version="1.0" encoding="UTF-8"?>
<instance-data-set xmlns=\
    "urn:ietf:params:xml:ns:yang:ietf-yang-instance-data">
  <name>acme-switch-notification-capabilities</name>
  <content-schema>
    <module>ietf-system-capabilities@2021-04-02</module>
    <module>ietf-notification-capabilities@2021-04-02</module>
  </content-schema>
  <description>Notification capabilities of acme-switch.
    Acme-switch implements the running, candidate and operational
    datastores. Every change can be reported on-change from the
    running datastore, nothing from the candidate datastore and
    all "config false" data from the operational datastore. Periodic
    subscriptions are supported for running and operational, but not
    for candidate datastore.
  </description>
  <content-data>
    <system-capabilities \
     xmlns="urn:ietf:params:xml:ns:yang:ietf-system-capabilities" \
     xmlns:notc=\
       "urn:ietf:params:xml:ns:yang:ietf-notification-capabilities" \
     xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
     <notc:subscription-capabilities>
      <notc:minimum-update-period>500</notc:minimum-update-period>
      <notc:max-nodes-per-update>2000</notc:max-nodes-per-update>
      <notc:minimum-dampening-period>100</notc:minimum-dampening-period>
      <notc:periodic-notifications-supported>\
        config-changes state-changes\
      </notc:periodic-notifications-supported>
     </notc:subscription-capabilities>
     <datastore-capabilities>
       <datastore>ds:operational</datastore>
       <per-node-capabilities>
         <node-selector>/</node-selector>
         <notc:subscription-capabilities>
           <notc:on-change-supported>\
             state-changes\
           </notc:on-change-supported>
         </notc:subscription-capabilities>
       </per-node-capabilities>
     </datastore-capabilities>
     <datastore-capabilities>
       <datastore>ds:candidate</datastore>
       <per-node-capabilities>
         <node-selector>/</node-selector>
         <notc:subscription-capabilities>
           <notc:on-change-supported/>
```

```
            <notc:periodic-notifications-supported/>
          </notc:subscription-capabilities>
        </per-node-capabilities>
      </datastore-capabilities>
      <datastore-capabilities>
       <datastore>ds:running</datastore>
        <per-node-capabilities>
          <node-selector>/</node-selector>
          <notc:subscription-capabilities>
            <notc:on-change-supported>\
              config-changes\
            </notc:on-change-supported>
          </notc:subscription-capabilities>
        </per-node-capabilities>
      </datastore-capabilities>
    </system-capabilities>
  </content-data>
</instance-data-set>
```

Figure 2: Notification Capabilities with datastore level settings

Appendix C. Changes between revisions

Note to RFC Editor (To be removed by RFC Editor)

v18 - v19

- *IESG review

v17 - v18

- *IESG review

v16 - v17

- *AD review comments addressed

v15 - v16

- *Two editorial comments from document shepherd

v14 - v15

- *Address the last comments from document shepherd

v13 - v14

- *Updated according to sheperds review
- *Added to import, which imported modules need to be implemented.
- *Added notes to the RFC editor
- *Re-arrange the sections, for a better reading flow
- *Many editorial changes
- *Replace YANG module prefix

v12 - v13

- *Rearranged order of notification capability leafs into 3 groups: generic, specific to periodic subscriptions, specific to on-change.
- *Introduced artwork folding in the examples
- *Updated to follow draft-ietf-netmod-yang-instance-file-format-10
- *Various editing changes

v11 - v12

* Updated max-nodes-per-update description

* Reformatted YANG models with pyang -f yang --keep-comments --yang-line-length 69

v10 - v11

* Updated examples

* Updated typedef notification-support

v09 - v10

* Removed description text from imports about the need for implementing the imported module.

* Changed notification-support to bits with shorter names

* Assigned enum values to supported-excluded-change-type

* Made node-selector a choice to allow for future alternative selection methods.

* Changed precedence of per-node-capabilities entries. Precedence is now according to the position of entries in the list.

v08 - v09

* Split the YANG module into two: ietf-system-capabilities and ietf-notification-capabilities. Restructured/updated the draft accordingly.

v07 - v08

* Prepared the YANG model to include other non-YANG-Push related capabilities.

* Renamed the top level container to system-capabilities

* Added a container subscription-capabilities to the grouping subscription-capabilities to contain all subscription related capabilities

* Updated examples according to draft-ietf-netmod-yang-instance-file-format-06.

v06 - v07

*Updated examples according to draft-ietf-netmod-yang-instance-file-format-05.

v05 - v06

*Providing the capability data is only a "SHOULD" recommendation. Some reviewers wanted MUST some wanted much less.

*The YANG module import statements now indicate the imported modules that must be implemented not just available as import as requested by the YangDoctors review.

v04 - v05

*Added new capabilities periodic-notifications-supported and supported-excluded-change-type.

*Restructured YANG module to make the node-selector's usage similar to how NACM uses it: "/" means the whole datastore.

*Small corrections, spelling, rewording

*Replaced the term server with the term publisher except in cases where we speak about datastores and functionality based on get, getconfig operations. In this latter case it is really the server functionality that is discussed.

v03 - v04

*Clarified recommended support for on-change notifications about the datastore-subscription-capabilities.

v02 - v03

*Allow throughput related capabilities to be defined on top, datastore or data node level. Described that specific capability values always override generic ones.

*Indicate that non-NMDA servers can also use this model.

*Updated according to draft-ietf-netmod-yang-instance-file-format-04

v01 - v02

*Added instance data examples

*On-change capability can be defined per datastore

*Added "if-feature yp:on-change" where relevant

*Unified units used

v00 - v01

*Add more capabilities: minimum period, supported period max-number of objects, min dampening period, dampening supported

Authors' Addresses

Balazs Lengyel
Ericsson
1117 Budapest
Magyar Tudosok korutja 11
Hungary

Email: balazs.lengyel@ericsson.com

Alexander Clemm
Futurewei
2330 Central Expressway
Santa Clara, CA 95050,
United States of America

Email: ludwig@clemm.org

Benoit Claise
Huawei

Email: benoit.claise@huawei.com