Stream: Internet Engineering Task Force (IETF)
RFC: 9196
Category: Standards Track
Published: February 2022
ISSN: 2070-1721
Authors: B. Lengyel, *Ericsson*; A. Clemm, *Futurewei*; B. Claise, *Huawei*

RFC 9196
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 runtime 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" (RFC 8641).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9196.

Copyright Notice

Table of Contents

1. Introduction

Servers and/or publishers often have capabilities, which can be represented by values that designate operational behavior, that need to be conveyed to clients. The YANG modules that are defined in this document facilitate this.

There is a need to publish this capability information as it is part of the API contract between the server and client. Examples include the 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 to represent such capabilities in a format that is:

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

Implementation-time information is needed by Network Management System (NMS) implementers. An NMS implementation that supports notifications needs 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 from which to retrieve data 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.

Runtime capability information is needed:

- for any "purely model-driven" application, e.g., a NETCONF browser. Such applications depend on reading models and capabilities at runtime to support all the publisher's available functionality.
- in case the capability might change during runtime, e.g., due to licensing, hardware constraints, etc.

- to check that 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 [RFC9195].

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.

Runtime 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 runtime.

For the implementation-time use case: Capabilities **SHOULD** be provided by the implementer as YANG instance data files complying with [RFC9195]. When provided, the file **MUST** already be available at implementation time and retrievable in a way that does not depend on a live network node, e.g., downloading from a product website.

For the runtime use case: Capabilities **SHOULD** be available via NETCONF [RFC6241] or RESTCONF [RFC8040] from the live server (implementing the publisher) during runtime. Implementations that support changing these capabilities at runtime **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.
- the 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-octet 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 the objects for which "on-change" notifications are supported and the objects for which they are not.

Support for "on-change" notifications does not mean that such notifications will be sent for any specific data node, as the ability to do so may not be supported for every data node. Therefore, subscriber/management applications cannot rely on the "on-change" functionality unless the subscriber has some means to identify the objects for which "on-change" notifications are in fact supported. YANG data models are meant to be used as an interface contract. Without identification of the data nodes actually supporting "on-change" notifications, 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 a 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. "ietf-system-capabilities" 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 at /sysc:system-capabilities and /sysc:system-capabilities/sysc:datastore-capabilities/sysc:per-node-capabilities.

Capability values can be specified at the system level, at the datastore level (by selecting all nodes in the datastore), or for specific data nodes of a specific datastore (and their contained subtrees). Capability values specified for a specific datastore or node-set override values specified at 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@2022-01-21.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 at the system level, at the
     datastore level (by selecting all nodes in the datastore), or
     for specific data nodes of a specific datastore (and their
     contained subtrees).
     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 at the
     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 that 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) 2022 IETF Trust and the persons identified as
      authors of the code.  All rights reserved.

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

      This version of this YANG module is part of RFC 9196
      (https://www.rfc-editor.org/info/rfc9196); see the RFC itself
      for full legal notices.";

  revision 2022-01-21 {
    description
      "Initial version";
    reference
      "RFC 9196: 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 at 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 were 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 must not be used, as it
           represents a set of configuration datastores.";
      }
      list per-node-capabilities {
        description
          "Each list entry specifies capabilities for the selected
           data nodes.  The same capabilities apply to 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 of [RFC8341].";
            reference
              "RFC 8341: Network Configuration Access Control Model";
          }
        }
        /*
         * "Augmentation point for datastore- or data-node-level
         * capabilities."
         */
      }
    }
  }
}
```

```
<CODE ENDS>
```

5. Notification Capabilities Model

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

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@2022-01-21.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 9196: YANG Modules Describing Capabilities for Systems
       and Datastore Update Notifications";
  }

  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 publisher capabilities related to
     YANG-Push (RFC 8641).

     The module contains:

     - a specification of the data nodes that 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 at the system/publisher
     level, at the datastore level, or for specific data nodes of
     a specific datastore (and their contained subtrees), 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) 2022 IETF Trust and the persons identified as
      authors of the code.  All rights reserved.

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

      This version of this YANG module is part of RFC 9196
      (https://www.rfc-editor.org/info/rfc9196); see the RFC itself
      for full legal notices.";

    revision 2022-01-21 {
      description
        "Initial version";
      reference
        "RFC 9196: YANG Modules Describing Capabilities for Systems
         and Datastore Update Notifications";
    }

    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.

            The bits config-changes or state-changes have no effect
            when they are set for a datastore or for a set of nodes
            that does not contain nodes with the indicated config
            value.  In those cases, the effect is the same as if no
            support was declared.  One example of this is indicating
            support for state-changes for a candidate datastore that
            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.

This document outlines a framework for conveying system capability information that is inherently flexible and extensible. While the full set of use cases is not known now, they may range as wide as conveying the minimum update period for periodic subscription updates and

what protocols might be used for such notifications. Knowledge of this type of value might, for example, allow an attacker to gain insight into how long unauthorized configuration changes might be active prior to detection and what communications channels might be disrupted to extend the period of non-detection. Documents adding additional capabilities via augmenting this module are encouraged to document the security considerations of the new YANG nodes, according to the guidance in BCP 216 [RFC8407].

All protocol-accessible data nodes in augmented modules are read-only and cannot be modified. Access control may be configured to avoid exposing any read-only data that is defined by the augmenting module documentation as being security sensitive.

When that data is in file format, the 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 [RFC9195], which includes 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 the following URIs in the "IETF XML Registry" [RFC3688]:

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

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

7.2. The YANG Module Names Registry

This document registers the following YANG modules in the "YANG Module Names" registry [RFC6020]:

name: ietf-system-capabilities
namespace: urn:ietf:params:xml:ns:yang:ietf-system-capabilities
prefix: sysc
reference: RFC 9196

name: ietf-notification-capabilities
namespace: urn:ietf:params:xml:ns:yang:ietf-notification-capabilities
prefix: notc
reference: RFC 9196

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

[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>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

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

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[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>.

[RFC9195] Lengyel, B. and B. Claise, "A File Format for YANG Instance Data", RFC 9195, DOI 10.17487/RFC9195, February 2022, <https://www.rfc-editor.org/info/rfc9195>.

8.2. Informative References

[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>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[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 can be reported on-change from the operational datastore. Interface statistics are not reported "on-change"; only two important counters are. Datastore subscription capabilities are not reported "on-change", as they never change on the acme-router during runtime.

```
========== 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@2022-01-21</module>
    <module>ietf-notification-capabilities@2022-01-21</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 can be reported on-change 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 Settings Specific to the Data Node

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 can be reported on-change from the candidate datastore, and all "config false" data can be reported on-change 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@2022-01-21</module>
    <module>ietf-notification-capabilities@2022-01-21</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

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 Working Group.

Authors' Addresses

Balazs Lengyel
Ericsson
Budapest
Magyar Tudosok korutja 11
1117
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
George's Court Townsend Street
Dublin 2
Ireland
Email: benoit.claise@huawei.com