Internet Engineering Task Force (IETF)
Request for Comments: 6243
Category: Standards Track
ISSN: 2070-1721

A. Bierman
Brocade
B. Lengyel
Ericsson

June 2011

With-defaults Capability for NETCONF

Abstract

The Network Configuration Protocol (NETCONF) defines ways to read and edit configuration data from a NETCONF server. In some cases, part of this data may not be set by the NETCONF client, but rather a default value known to the server is used instead. In many situations the NETCONF client has a priori knowledge about default data, so the NETCONF server does not need to save it in a NETCONF configuration datastore or send it to the client in a retrieval operation reply. In other situations the NETCONF client will need this data from the server. Not all server implementations treat this default data the same way. This document defines a capability-based extension to the NETCONF protocol that allows the NETCONF client to identify how defaults are processed by the server, and also defines new mechanisms for client control of server processing of default data.

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 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6243.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://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 3
1.1. Terminology 3
1.2. Default-Handling Behavior 5
1.3. Client Controlled Retrieval of Default Data 5
2. Default-Handling Basic Modes 6
2.1. 'report-all' Basic Mode 6
2.1.1. 'report-all' Basic Mode Retrieval 6
2.1.2. 'report-all' <with-defaults> Retrieval 6
2.1.3. 'report-all' <edit-config> and <copy-config> Behavior 6
2.2. 'trim' Basic Mode 7
2.2.1. 'trim' Basic Mode Retrieval 7
2.2.2. 'trim' <with-defaults> Retrieval 7
2.2.3. 'trim' <edit-config> and <copy-config> Behavior . . . 7
2.3. 'explicit' Basic Mode 8
2.3.1. 'explicit' Basic Mode Retrieval 8
2.3.2. 'explicit' <with-defaults> Retrieval 8
2.3.3. 'explicit' <edit-config> and <copy-config> Behavior . 8
3. Retrieval of Default Data 9
3.1. 'report-all' Retrieval Mode 9
3.2. 'trim' Retrieval Mode 9
3.3. 'explicit' Retrieval Mode 9
3.4. 'report-all-tagged' Retrieval Mode 9
4. With-defaults Capability 10
4.1. Overview 10
4.2. Dependencies 10
4.3. Capability Identifier 10
4.4. New Operations 11
4.5. Modifications to Existing Operations 11
4.5.1. <get>, <get-config>, and <copy-config> Operations . . 11
4.5.2. <edit-config> Operation 12

4.5.3. Other Operations 13
4.6. Interactions with Other Capabilities 13
5. YANG Module for the <with-defaults> Parameter 13
6. XSD for the 'default' Attribute 17
7. IANA Considerations 18
8. Security Considerations 18
9. Acknowledgements 19
10. Normative References 19
Appendix A. Usage Examples 20
A.1. Example YANG Module 20
A.2. Example Data Set 21
A.3. Protocol Operation Examples 22
A.3.1. <with-defaults> = 'report-all' 22
A.3.2. <with-defaults> = 'report-all-tagged' 23
A.3.3. <with-defaults> = 'trim' 24
A.3.4. <with-defaults> = 'explicit' 25

1. Introduction

The NETCONF protocol [RFC6241] defines ways to read configuration and state data from a NETCONF server. Part of the configuration data may not be set by the NETCONF client, but rather by a default value from the data model. In many situations the NETCONF client has a priori knowledge about default data, so the NETCONF server does not need to send it to the client. A priori knowledge can be obtained, e.g., from a document formally describing the data models supported by the NETCONF server.

It can be important for a client to know exactly how a server implementation will handle default data. There are subtle differences in some protocol operations where the default-handling behavior of the server will affect the outcome of the operation.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Data model schema: A document or set of documents describing the data models supported by the NETCONF server.

Management application: A computer program running outside the NETCONF server that configures or supervises the NETCONF server. A management application can reach the device, e.g., via NETCONF, command line interface (CLI), or the Simple Network Management Protocol (SNMP).

Schema default data: Data specified in the data model schema as default, that is, set or used by the device whenever the NETCONF client or other management application/user does not provide a specific value for the relevant data node. Schema default data may or may not be stored as part of a configuration datastore, depending on the basic mode used by a particular server.

Default data: Conceptual data containing a default value. Default data is not kept in a datastore. Not all servers use the same criteria to decide if a data node is actually instantiated in a datastore. If a data node is not present in a datastore, and a schema default definition is in use by the server instead, then it is considered to be a default data node.

Default value: A default value is a value for a data node instance that is conceptually in use by the server, when the data node instance does not exist.

Explicitly set data: Data that is set to any value by a NETCONF client or other management application by the way of an explicit management operation, including any data model schema default value. Any value set by the NETCONF server that is not the schema defined default value is also considered explicitly set data.

<with-defaults> retrieval: Refers to a protocol operation that includes the <with-default> parameter to control the handling of default data.

:with-defaults: The shorthand notation for the with-defaults capability identifier.

The following terms are defined in [RFC6241]:

- client
- datastore
- operation
- server

The following term is defined in [RFC6020]:

- data node

1.2. Default-Handling Behavior

The default-handling behavior used by a server will impact NETCONF protocol operations in two ways:

1. Data retrieval: A server is normally allowed to exclude data nodes that it considers to contain the default value. The actual nodes omitted depend on the default-handling behavior used by the server.

2. Create and delete operations: The <edit-config> 'operation' attribute can be used to create and/or delete specific data nodes. These operations depend on whether or not the target node currently exists. The server's default-handling behavior will determine whether or not the requested node currently exists in the configuration datastore.

1.3. Client Controlled Retrieval of Default Data

A networking device may have a large number of default values. Often the default values are specifically defined with a reasonable value, documented and well-known, so that the management user does not need to handle them. For these reasons, it is quite common for networking devices to suppress the output of parameters having the default value.

However, there are use-cases when a NETCONF client will need the default data from the server:

- The management application often needs a single, definitive, and complete set of configuration values that determine how the networking device works.

- Documentation about default values can be unreliable or unavailable.

- Some management applications might not have the capabilities to correctly parse and interpret formal data models.

- Human users might want to understand the received data without consultation of the documentation.

In all these cases, the NETCONF client will need a mechanism to retrieve default data from a NETCONF server.

This document defines a NETCONF protocol capability to identify the server's default-handling behavior, an XML [W3C.REC-xmlschema-0-20041028] attribute to identify default data,

and a YANG module extension to the NETCONF protocol that allows the NETCONF client to control whether default data is returned by the server.

2. Default-Handling Basic Modes

Not all server implementations treat default data in the same way. Instead of forcing a single implementation strategy, this document allows a server to advertise a particular style of default-handling, and the client can adjust accordingly. Client implementations are expected to be powerful enough to support all three of the server basic default-handling modes.

NETCONF servers report default data in different ways. This document specifies three standard default-handling basic modes that a server implementer may choose from:

- report-all
- trim
- explicit

A server MUST select one of the three basic modes defined in this section for handling default data.

2.1. 'report-all' Basic Mode

A server that uses the 'report-all' basic mode does not consider any data node to be default data, even schema default data.

2.1.1. 'report-all' Basic Mode Retrieval

When data is retrieved from a server using the 'report-all' basic mode, and the <with-defaults> parameter is not present, all data nodes MUST be reported.

2.1.2. 'report-all' <with-defaults> Retrieval

If the 'report-all' basic mode is used by the server, then the server MUST support the <with-defaults> parameter with a value equal to 'report-all', as specified in Section 3.1.

2.1.3. 'report-all' <edit-config> and <copy-config> Behavior

The server MUST consider every data node to exist, even those containing a schema default value. A valid 'create' operation attribute for a data node that contains its schema default value MUST

fail with a 'data-exists' error-tag. A valid 'delete' operation attribute for a data node that contains its schema default value MUST succeed, even though the data node is immediately replaced by the server with the default value.

A server that uses the 'report-all' basic mode has no concept of a default node, so the 'report-all-tagged' <with-defaults> retrieval mode is not relevant. There will never be any tagged nodes, since there are no nodes that are omitted in a basic-mode retrieval operation. If the 'default' attribute is present in any configuration data, the server MUST return an <rpc-error> response with an 'unknown-attribute' error-tag.

2.2. 'trim' Basic Mode

A server that uses the 'trim' basic mode MUST consider any data node set to its schema default value to be default data.

2.2.1. 'trim' Basic Mode Retrieval

When data is retrieved from a server using the 'trim' basic mode, and the <with-defaults> parameter is not present, data nodes MUST NOT be reported if they contain the schema default value. Non-configuration data nodes containing the schema default value MUST NOT be reported.

2.2.2. 'trim' <with-defaults> Retrieval

If the 'trim' basic mode is used by the server, then the server MUST support the <with-defaults> parameter with a value equal to 'trim', as specified in Section 3.2.

2.2.3. 'trim' <edit-config> and <copy-config> Behavior

The server MUST consider any data node that does not contain its schema default value to exist. A valid 'create' operation attribute for a data node that has a schema default value defined MUST succeed. A valid 'delete' operation attribute for a missing data node that has a schema default value MUST fail. The server MUST return an <rpc-error> response with a 'data-missing' error-tag.

If a client sets a data node to its schema default value, using any valid operation, it MUST succeed, although the data node MUST NOT be saved in the NETCONF configuration datastore. This has the same effect as removing the data node and treating it as default data.

If the server supports the 'report-all-tagged' value for the <with-defaults> parameter, then the 'default' attribute MUST be

accepted in configuration input, as described in Section 4.5.1 and Section 4.5.2.

2.3. 'explicit' Basic Mode

A server that uses the 'explicit' basic mode MUST consider any data node that is not explicitly set data to be default data.

2.3.1. 'explicit' Basic Mode Retrieval

When data is retrieved from a server using the 'explicit' basic mode, and the <with-defaults> parameter is not present, data nodes MUST be reported if explicitly set by the client, even if they contain the schema default value. Non-configuration data nodes containing the schema default value MUST be reported.

2.3.2. 'explicit' <with-defaults> Retrieval

If the 'explicit' basic mode is used by the server, the server MUST support the <with-defaults> parameter with a value equal to 'explicit', as specified in Section 3.3.

2.3.3. 'explicit' <edit-config> and <copy-config> Behavior

The server considers any data node that is explicitly set data to exist. A valid 'create' operation attribute for a data node that has been set by a client to its schema default value MUST fail with a 'data-exists' error-tag. A valid 'create' operation attribute for a data node that has been set by the server to its schema default value MUST succeed. A valid 'delete' operation attribute for a data node that has been set by a client to its schema default value MUST succeed. A valid 'delete' operation attribute for a data node that has been set by the server to its schema default value MUST fail with a 'data-missing' error-tag.

If the server supports the 'report-all-tagged' retrieval mode in its :with-defaults capability, then the 'default' attribute MUST be accepted in configuration input. If all NETCONF <edit-config> or <copy-config> parameters are valid, then the server will treat a tagged data node (i.e., the 'default' attribute set to 'true' or '1') as a request to return that node to default data. If this request is valid within the context of the requested NETCONF operation, then the data node is removed and returned to its default value. The data node within the NETCONF message MUST contain a value in this case, which MUST be equal to the schema default value. If not, the server MUST return an <rpc-error> response with an 'invalid-value' error-tag.

3. Retrieval of Default Data

This document defines a new parameter, called <with-defaults>, which can be added to specific NETCONF operation request messages to control how retrieval of default data is treated by the server.

A server that implements this specification MUST accept the <with-defaults> parameter containing the enumeration for any of the default-handling modes it supports. The <with-defaults> parameter contains one of the four enumerations defined in this section.

3.1. 'report-all' Retrieval Mode

When data is retrieved with a <with-defaults> parameter equal to 'report-all', all data nodes MUST be reported, including any data nodes considered to be default data by the server.

3.2. 'trim' Retrieval Mode

When data is retrieved with a <with-defaults> parameter equal to 'trim', data nodes MUST NOT be reported if they contain the schema default value. Non-configuration data nodes containing the schema default value MUST NOT be reported.

3.3. 'explicit' Retrieval Mode

When data is retrieved with a <with-defaults> parameter equal to 'explicit', a data node that was set by a client to its schema default value MUST be reported. A conceptual data node that would be set by the server to the schema default value MUST NOT be reported. Non-configuration data nodes containing the schema default value MUST be reported.

3.4. 'report-all-tagged' Retrieval Mode

In addition to the basic modes, a special variant of the 'report-all' basic mode is available called 'report-all-tagged'. This mode MUST be supported on a server if the 'also-supported' parameter in the :with-defaults capability contains the 'report-all-tagged' option. Refer to Section 4 for encoding details for this capability.

In this mode the server returns all data nodes, just like the 'report-all' mode, except a data node that is considered by the server to contain default data will include an XML attribute to indicate this condition. This is useful for an application to determine which nodes are considered to contain default data by the server, within a single retrieval operation.

A server that supports 'report-all-tagged' MUST also accept the 'default' XML attribute within configuration input to the <edit-config> or <copy-config> operations. Refer to Section 6 for XML encoding details of the 'default' XML attribute.

4. With-defaults Capability

4.1. Overview

The :with-defaults capability indicates which default-handling basic mode is supported by the server. It may also indicate support for additional defaults retrieval modes. These retrieval modes allow a NETCONF client to control whether default data is returned by the server. The capability affects both configuration and state data (while acknowledging that the usage of default values for state data is less prevalent). Sending of default data is controlled for each individual operation separately.

A NETCONF server implementing the :with-defaults capability:

- MUST indicate its basic mode behavior by including the 'basic-mode' parameter in the capability URI, as defined in Section 4.3.
- MUST support the YANG module defined in Section 5 for the default-handling mode indicated by the 'basic-mode' parameter.
- SHOULD support the YANG module in Section 5 for the default-handling mode identified by the 'report-all' or 'report-all-tagged' enumeration value.
- If the 'report-all-tagged' default-handling mode is supported, then the 'default' attribute MUST be supported.
- MAY support the YANG module in Section 5 for additional default-handling modes.

4.2. Dependencies

None.

4.3. Capability Identifier

urn:ietf:params:netconf:capability:with-defaults:1.0

The identifier MUST have a parameter: "basic-mode". This indicates how the server will treat default data, as defined in Section 2. The allowed values of this parameter are 'report-all', 'trim', and 'explicit', as defined in Section 2.

The identifier MAY have another parameter: "also-supported". This parameter indicates which additional enumeration values (besides the basic-mode enumeration) the server will accept for the <with-defaults> parameter in Section 5. The value of the parameter is a comma-separated list of one or more modes that are supported besides the mode indicated in the 'basic-mode' parameter. Possible modes are 'report-all', 'report-all-tagged', 'trim', and 'explicit', as defined in Section 3.

Note that this protocol capability URI is separate from the YANG module capability URI for the YANG module in Section 5. A server that implements this module MUST also advertise a YANG module capability URI according to the rules specified in [RFC6020].

Examples:

urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=explicit

urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=explicit&also-supported=report-all,report-all-tagged

4.4. New Operations

None.

4.5. Modifications to Existing Operations

4.5.1. <get>, <get-config>, and <copy-config> Operations

A new <with-defaults> XML element is added to the input for the <get>, <get-config>, and <copy-config> operations. If the <with-defaults> element is present, it controls the reporting of default data. The server MUST return default data in the NETCONF <rpc-reply> messages according to the value of this element, if the server supports the specified retrieval mode.

This parameter only controls these specified retrieval operations, and does not impact any other operations or the non-volatile storage of configuration data.

The <with-defaults> element is defined in the XML namespace for the ietf-netconf-with-defaults.yang module in Section 5, not the XML namespace for the <get>, <get-config>, and <copy-config> operations.

Allowed values of the with-defaults element are taken from the 'with-defaults-type' typedef in Section 5. The allowed values for a particular server are restricted to the values that the server

indicates it supports within the :with-defaults capability, in the 'basic-mode' and 'also-supported' parameters.

If an unsupported value is used, the NETCONF server MUST return an <rpc-error> response with an 'invalid-value' error-tag.

If the <with-defaults> element is not present, the server MUST follow its basic mode behavior as indicated by the :with-defaults capability identifier's 'basic-mode' parameter, defined in Section 4.3.

The <get> and <get-config> operations support a separate filtering mechanism, using the <filter> parameter. The defaults filtering is conceptually done before the <filter> parameter is processed. For example, if the <with-defaults> parameter is equal to 'report-all', then the <filter> parameter is conceptually applied to all data nodes and all default data.

The <copy-config> operation is only affected by the <with-defaults> parameter if the target of the operation is specified with the <url> parameter. If the target is a NETCONF configuration datastore (i.e., running, candidate, or startup), the <with-defaults> parameter has no effect. The server MUST use its basic mode when copying data to a NETCONF configuration datastore. If the <with-defaults> parameter is present in this case, it MUST be silently ignored by the server.

If the server supports the 'report-all-tagged' mode, then the 'default' attribute defined in Section 6 also impacts the <copy-config> operation. If the 'default' attribute is present and set to 'true' or '1', then the server MUST treat the new data node as a request to return that node to its default value (i.e., remove it from the configuration datastore). The data node within the NETCONF message MUST contain a value in this case, which MUST be equal to the schema default value. If not, the server MUST return an <rpc-error> response with an 'invalid-value' error-tag.

4.5.2. <edit-config> Operation

The <edit-config> operation has several editing modes. The 'create' and 'delete' editing operations are affected by the default-handling basic mode. The other enumeration values for the NETCONF operation attribute are not affected.

If the operation attribute contains the value 'create', and the data node already exists in the target configuration datastore, then the server MUST return an <rpc-error> response with an 'invalid-value' error-tag.

If the client sets a data node to its schema default value, the server MUST accept the request if it is valid. The server MUST keep or discard the new value based on its default-handling basic mode. For the 'trim' basic mode, all schema default values are discarded; otherwise, a client-provided schema default value is saved in a NETCONF configuration datastore.

If the server supports the 'report-all-tagged' mode, then the 'default' attribute defined in Section 6 also impacts the <edit-config> operation. If the 'default' attribute is present and set to 'true' or '1', then the server MUST treat the new data node as a request to return that node to its default value (i.e., remove it from the configuration datastore). The data node within the NETCONF message MUST contain a value in this case, which MUST be equal to the schema default value. If not, the server MUST return an <rpc-error> response with an 'invalid-value' error-tag.

If the 'default' attribute is present, then the effective operation for the target data node MUST be 'create', 'merge', or 'replace'. If not, then the server MUST return an <rpc-error> response with an 'invalid-value' error-tag. For example, if 'create' is the effective operation, then the create request must be valid on its own (e.g., current data node MUST NOT exist). The procedure for determining the effective operation is defined in [RFC6241]. It is derived from the 'default-operation' parameter and/or any operation attributes that are present in the data node or any of its ancestor nodes, within the <edit-config> request.

4.5.3. Other Operations

Other operations that return configuration data SHOULD also handle default data according to the rules set in this document, and explicitly state this in their documentation. If this is not specified in the document defining the respective operation, the default-handling rules described herein do not affect these operations.

4.6. Interactions with Other Capabilities

None.

5. YANG Module for the <with-defaults> Parameter

The following YANG module defines the addition of the with-defaults element to the <get>, <get-config>, and <copy-config> operations. The YANG language is defined in [RFC6020]. The above operations are defined in YANG in [RFC6241]. Every NETCONF server that supports the :with-defaults capability MUST implement this YANG module.

```
  <CODE BEGINS> file="ietf-netconf-with-defaults@2011-06-01.yang"

  module ietf-netconf-with-defaults {

     namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults";

     prefix ncwd;

     import ietf-netconf { prefix nc; }

     organization
      "IETF NETCONF (Network Configuration Protocol) Working Group";

     contact
      "WG Web:   <http://tools.ietf.org/wg/netconf/>

       WG List:  <netconf@ietf.org>

       WG Chair: Bert Wijnen
                 <bertietf@bwijnen.net>

       WG Chair: Mehmet Ersue
                 <mehmet.ersue@nsn.com>

       Editor: Andy Bierman
               <andy.bierman@brocade.com>

       Editor: Balazs Lengyel
               <balazs.lengyel@ericsson.com>";

     description
      "This module defines an extension to the NETCONF protocol
       that allows the NETCONF client to control how default
       values are handled by the server in particular NETCONF
       operations.

       Copyright (c) 2011 IETF Trust and the persons identified as
       the document authors.  All rights reserved.

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

       This version of this YANG module is part of RFC 6243; see
       the RFC itself for full legal notices.";
```

```
     revision 2011-06-01 {
       description
         "Initial version.";
       reference
        "RFC 6243: With-defaults Capability for NETCONF";
     }

     typedef with-defaults-mode {
        description
          "Possible modes to report default data.";
        reference
           "RFC 6243; Section 3.";
        type enumeration {
           enum report-all {
               description
                 "All default data is reported.";
               reference
                 "RFC 6243; Section 3.1";
           }
           enum report-all-tagged {
               description
                 "All default data is reported.
                  Any nodes considered to be default data
                  will contain a 'default' XML attribute,
                  set to 'true' or '1'.";
               reference
                 "RFC 6243; Section 3.4";
           }
           enum trim {
               description
                 "Values are not reported if they contain the default.";
               reference
                 "RFC 6243; Section 3.2";
           }
           enum explicit {
               description
                 "Report values that contain the definition of
                  explicitly set data.";
               reference
                 "RFC 6243; Section 3.3";
           }
       }
     }

     grouping with-defaults-parameters {
       description
         "Contains the <with-defaults> parameter for control
          of defaults in NETCONF retrieval operations.";
```

```
      leaf with-defaults {
        description
          "The explicit defaults processing mode requested.";
        reference
          "RFC 6243; Section 4.5.1";

        type with-defaults-mode;
      }
    }

    // extending the get-config operation
    augment /nc:get-config/nc:input {
        description
          "Adds the <with-defaults> parameter to the
           input of the NETCONF <get-config> operation.";
        reference
          "RFC 6243; Section 4.5.1";

        uses with-defaults-parameters;
    }

    // extending the get operation
    augment /nc:get/nc:input {
        description
          "Adds the <with-defaults> parameter to
           the input of the NETCONF <get> operation.";
        reference
          "RFC 6243; Section 4.5.1";

        uses with-defaults-parameters;
    }

    // extending the copy-config operation
    augment /nc:copy-config/nc:input {
        description
          "Adds the <with-defaults> parameter to
           the input of the NETCONF <copy-config> operation.";
        reference
          "RFC 6243; Section 4.5.1";

        uses with-defaults-parameters;
    }

  }

  <CODE ENDS>
```

6. XSD for the 'default' Attribute

The following XML Schema document [W3C.REC-xml-20081126] defines the 'default' attribute, described within this document. This XSD is only relevant if the server supports the 'report-all-tagged' defaults retrieval mode.

The 'default' attribute uses the XSD data type 'boolean'. In accordance with Section 3.2.2.1 of XML Schema Part 2: Datatypes, the allowable lexical representations for the xs:boolean datatype are the strings "0" and "false" for the concept of false and the strings "1" and "true" for the concept of true. Implementations MUST support both styles of lexical representation.

```
<CODE BEGINS> file="defaults.xsd"

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns="urn:ietf:params:xml:ns:netconf:default:1.0"
           targetNamespace="urn:ietf:params:xml:ns:netconf:default:1.0"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified"
           xml:lang="en">

  <xs:annotation>
    <xs:documentation>
      This schema defines the syntax for the 'default' attribute
      described within this document.
    </xs:documentation>
  </xs:annotation>

  <!--
    default attribute
    -->
  <xs:attribute name="default" type="xs:boolean" default="false">
    <xs:annotation>
      <xs:documentation>
        This attribute indicates whether the data node represented
        by the XML element containing this attribute is considered
        by the server to be default data.  If set to 'true' or '1', then
        the data node is default data.  If 'false' or '0', then the
        data node is not default data.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
```

```
</xs:schema>

<CODE ENDS>
```

7. IANA Considerations

This document registers the following capability identifier URN in the 'Network Configuration Protocol (NETCONF) Capability URNs' registry:

```
   urn:ietf:params:netconf:capability:with-defaults:1.0
```

This document registers two XML namespace URNs in the 'IETF XML registry', following the format defined in [RFC3688].

```
   URI: urn:ietf:params:xml:ns:netconf:default:1.0

   URI: urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults
```

Registrant Contact: The NETCONF WG of the IETF.

XML: N/A, the requested URIs are XML namespaces.

This document registers one module name in the 'YANG Module Names' registry, defined in [RFC6020] .

```
   name: ietf-netconf-with-defaults

   prefix: ncwd

   namespace: urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults

   RFC: 6243
```

8. Security Considerations

This document defines an extension to existing NETCONF protocol operations. It does not introduce any new or increased security risks into the management system.

The 'with-defaults' capability gives clients control over the retrieval of default data from a NETCONF datastore. The security consideration of [RFC6241] applies to this document as well.

9. Acknowledgements

Thanks to Martin Bjorklund, Sharon Chisholm, Phil Shafer, Juergen Schoenwaelder, Kent Watsen, Washam Fan, and many other members of the NETCONF WG for providing important input to this document.

10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[W3C.REC-xml-20081126]
Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", World Wide Web Consortium Recommendation REC-xml-20081126, November 2008, <http://www.w3.org/TR/2008/REC-xml-20081126>.

[W3C.REC-xmlschema-0-20041028]
Fallside, D. and P. Walmsley, "XML Schema Part 0: Primer Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-0-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-0-20041028>.

Appendix A. Usage Examples

A.1. Example YANG Module

The following YANG module defines an example interfaces table to demonstrate how the <with-defaults> parameter behaves for a specific data model.

Note that this is not a real module, and implementation of this module is not required for conformance to the :with-defaults capability, defined in Section 4. This module is not to be registered with IANA, and is not considered to be a code component. It is intentionally very terse, and includes few descriptive statements.

```
  module example {

  namespace "http://example.com/ns/interfaces";

  prefix exam;

  typedef status-type {
     description "Interface status";
     type enumeration {
       enum ok;
       enum 'waking up';
       enum 'not feeling so good';
       enum 'better check it out';
       enum 'better call for help';
     }
     default ok;
  }

  container interfaces {
      description "Example interfaces group";

      list interface {
        description "Example interface entry";
        key name;

        leaf name {
          description
            "The administrative name of the interface.
             This is an identifier that is only unique
             within the scope of this list, and only
             within a specific server.";
          type string {
```

```
            length "1 .. max";
          }
        }

        leaf mtu {
          description
            "The maximum transmission unit (MTU) value assigned to
             this interface.";
          type uint32;
          default 1500;
        }

        leaf status {
          description
            "The current status of this interface.";
          type status-type;
          config false;
        }
      }
    }
  }
```

A.2. Example Data Set

The following data element shows the conceptual contents of the example server for the protocol operation examples in the next section. This includes all the configuration data nodes, non-configuration data nodes, and default leafs.

```
    <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <interfaces xmlns="http://example.com/ns/interfaces">
        <interface>
          <name>eth0</name>
          <mtu>8192</mtu>
          <status>up</status>
        </interface>
        <interface>
          <name>eth1</name>
          <mtu>1500</mtu>
          <status>up</status>
        </interface>
        <interface>
          <name>eth2</name>
          <mtu>9000</mtu>
          <status>not feeling so good</status>
        </interface>
        <interface>
          <name>eth3</name>
```

```
            <mtu>1500</mtu>
            <status>waking up</status>
          </interface>
        </interfaces>
      </data>
```

In this example, the 'mtu' field for each interface entry is set in the following manner:

name	set by	mtu
eth0	client	8192
eth1	server	1500
eth2	client	9000
eth3	client	1500

A.3. Protocol Operation Examples

The following examples show some <get> operations using the 'with-defaults' element. The data model used for these examples is defined in Appendix A.1.

The client is retrieving all the data nodes within the 'interfaces' object, filtered with the <with-defaults> parameter.

A.3.1. <with-defaults> = 'report-all'

The behavior of the <with-defaults> parameter handling for the value 'report-all' is demonstrated in this example.

```
 <rpc message-id="101"
      xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <get>
     <filter type="subtree">
       <interfaces xmlns="http://example.com/ns/interfaces"/>
     </filter>
     <with-defaults
      xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults">
       report-all
     </with-defaults>
   </get>
 </rpc>

 <rpc-reply message-id="101"
            xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <data>
```

```
        <interfaces xmlns="http://example.com/ns/interfaces">
          <interface>
            <name>eth0</name>
            <mtu>8192</mtu>
            <status>up</status>
          </interface>
          <interface>
            <name>eth1</name>
            <mtu>1500</mtu>
            <status>up</status>
          </interface>
          <interface>
            <name>eth2</name>
            <mtu>9000</mtu>
            <status>not feeling so good</status>
          </interface>
          <interface>
            <name>eth3</name>
            <mtu>1500</mtu>
            <status>waking up</status>
          </interface>
        </interfaces>
      </data>
    </rpc-reply>
```

A.3.2. <with-defaults> = 'report-all-tagged'

The behavior of the <with-defaults> parameter handling for the value 'report-all-tagged' is demonstrated in this example. A 'tagged' data node is an element that contains the 'default' XML attribute, set to 'true' or '1'.

The actual data nodes tagged by the server depend on the default-handling basic mode used by the server. Only the data nodes that are considered to be default data will be tagged.

In this example, the server's basic mode is equal to 'trim', so all data nodes that would contain the schema default value are tagged. If the server's basic mode is 'explicit', then only data nodes that are not explicitly set data are tagged. If the server's basic mode is 'report-all', then no data nodes are tagged.

```
    <rpc message-id="102"
         xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <get>
        <filter type="subtree">
          <interfaces xmlns="http://example.com/ns/interfaces"/>
        </filter>
```

```
        <with-defaults
         xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults">
          report-all-tagged
        </with-defaults>
      </get>
    </rpc>

    <rpc-reply message-id="102"
               xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
               xmlns:wd="urn:ietf:params:xml:ns:netconf:default:1.0">
      <data>
        <interfaces xmlns="http://example.com/ns/interfaces">
          <interface>
            <name>eth0</name>
            <mtu>8192</mtu>
            <status wd:default="true">up</status>
          </interface>
          <interface>
            <name>eth1</name>
            <mtu wd:default="true">1500</mtu>
            <status wd:default="true">up</status>
          </interface>
          <interface>
            <name>eth2</name>
            <mtu>9000</mtu>
            <status>not feeling so good</status>
          </interface>
          <interface>
            <name>eth3</name>
            <mtu wd:default="true">1500</mtu>
            <status>waking up</status>
          </interface>
        </interfaces>
      </data>
    </rpc-reply>
```

A.3.3. <with-defaults> = 'trim'

The behavior of the <with-defaults> parameter handling for the value 'trim' is demonstrated in this example.

```
    <rpc message-id="103"
         xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <get>
        <filter type="subtree">
          <interfaces xmlns="http://example.com/ns/interfaces"/>
        </filter>
        <with-defaults
```

```
        xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults">
        trim
      </with-defaults>
    </get>
  </rpc>

  <rpc-reply message-id="103"
             xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <data>
      <interfaces xmlns="http://example.com/ns/interfaces">
        <interface>
          <name>eth0</name>
          <mtu>8192</mtu>
        </interface>
        <interface>
          <name>eth1</name>
        </interface>
        <interface>
          <name>eth2</name>
          <mtu>9000</mtu>
          <status>not feeling so good</status>
        </interface>
        <interface>
          <name>eth3</name>
          <status>waking up</status>
        </interface>
      </interfaces>
    </data>
  </rpc-reply>
```

A.3.4. <with-defaults> = 'explicit'

The behavior of the <with-defaults> parameter handling for the value 'explicit' is demonstrated in this example.

```
  <rpc message-id="104"
       xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get>
      <filter type="subtree">
        <interfaces xmlns="http://example.com/ns/interfaces"/>
      </filter>
      <with-defaults
       xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-with-defaults">
        explicit
      </with-defaults>
    </get>
  </rpc>
```

```
    <rpc-reply message-id="104"
               xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <data>
        <interfaces xmlns="http://example.com/ns/interfaces">
          <interface>
            <name>eth0</name>
            <mtu>8192</mtu>
            <status>up</status>
          </interface>
          <interface>
            <name>eth1</name>
            <status>up</status>
          </interface>
          <interface>
            <name>eth2</name>
            <mtu>9000</mtu>
            <status>not feeling so good</status>
          </interface>
          <interface>
            <name>eth3</name>
            <mtu>1500</mtu>
            <status>waking up</status>
          </interface>
        </interfaces>
      </data>
    </rpc-reply>
```

Authors' Addresses

Andy Bierman
Brocade

EMail: andy.bierman@brocade.com

Balazs Lengyel
Ericsson
Budapest,
Hungary

EMail: balazs.lengyel@ericsson.com