Internet Engineering Task Force (IETF)
Request for Comments: 6022
Category: Standards Track
ISSN: 2070-1721

M. Scott
Ericsson
M. Bjorklund
Tail-f Systems

October 2010

YANG Module for NETCONF Monitoring

Abstract

This document defines a Network Configuration Protocol (NETCONF) data model to be used to monitor the NETCONF protocol. The monitoring data model includes information about NETCONF datastores, sessions, locks, and statistics. This data facilitates the management of a NETCONF server. This document also defines methods for NETCONF clients to discover data models supported by a NETCONF server and defines a new NETCONF <get-schema> operation to retrieve them.

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/rfc6022.

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] model to be used to monitor the NETCONF protocol. It provides information about NETCONF sessions and supported schema as defined in [RFC4741].

Considerations such as different schema formats, feature optionality, and access controls can all impact the applicability and level of detail the NETCONF server sends to a client during session setup. The methods defined in this document address the need for further means to query and retrieve schema and NETCONF state information from a NETCONF server. These are provided to complement existing base NETCONF capabilities and operations and in no way affect existing behaviour.

A new <get-schema> operation is also defined to support explicit schema retrieval via NETCONF.

1.1. Definition of Terms

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].

2. Data Model to Monitor NETCONF

The NETCONF monitoring data model defined in this document provides operational information on the NETCONF server. This includes details specific to the NETCONF protocol (e.g., protocol-specific counters such as 'in-sessions') as well as data related to schema retrieval (e.g., schema list).

A server that implements the data model defined in this document ("urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring") MUST advertise the capability URI as described in [RFC6020].

This section presents an overview of the monitoring data model. For detailed descriptions, refer to the normative YANG module provided in this document (see Section 5).

2.1. The /netconf-state Subtree

The netconf-state container is the root of the monitoring data model.

```
netconf-state
    /capabilities
    /datastores
    /schemas
    /sessions
    /statistics

capabilities
   List of NETCONF capabilities supported by the server.

datastores
   List of NETCONF configuration datastores (e.g., running, startup,
   candidate) supported on this device and related information.

schemas
   List of schemas supported on the server.  Includes all the
   information required to identify the schemas and to support their
   retrieval.

sessions
   List of all active NETCONF sessions on the device.  Includes per-
   session counters for all NETCONF sessions.

statistics
   Includes global counters for the NETCONF server.
```

2.1.1. The /netconf-state/capabilities Subtree

The /netconf-state/capabilities subtree contains the capabilities supported by the NETCONF server. The list MUST include all capabilities exchanged during session setup still applicable at the time of the request.

2.1.2. The /netconf-state/datastores Subtree

The /netconf-state/datastores subtree contains the list of available datastores for the NETCONF server and includes information on their lock state.

```
datastore
    /name
    /locks

name (leaf, netconf-datastore-type)
   Enumeration of supported datastores; candidate, running, startup.
```

locks (grouping, lock-info)
   List of locks for the datastore. Information is provided for both global and partial locks [RFC5717]. For partial locks, the list of locked nodes and the select expressions originally used to request the lock are returned.

2.1.3. The /netconf-state/schemas Subtree

The list of supported schema for the NETCONF server.

```
schema
    /identifier   (key)
    /version      (key)
    /format       (key)
    /namespace
    /location
```

The elements identifier, version, and format are used as a key in the schema list. These are used in the <get-schema> operation.

identifier (string)
   Identifier for the schema list entry. The identifier is used in the <get-schema> operation and may be used for other means such as file retrieval.

version (string)
   Version of the schema supported. Multiple versions MAY be supported simultaneously by a NETCONF server. Each version MUST be reported individually in the schema list, i.e., with same identifier, possibly different location, but different version.

   For YANG data models, version is the value of the most recent YANG 'revision' statement in the module or submodule, or the empty string if no 'revision' statement is present.

format (identifyref, schema-format)
   The data modeling language the schema is written in. The data modeling language is represented as a YANG identity. This document defines the identities "xsd", "yang", "yin", "rng", and "rnc" (see Section 5).

namespace (inet:uri)
   The Extensible Markup Language (XML) namespace [XML-NAMES] defined by the schema.

location (union: enum, inet:uri)
One or more locations from which this specific schema can be retrieved. The list SHOULD contain at least one entry per schema.

2.1.4. The /netconf-state/sessions Subtree

Includes session-specific data for NETCONF management sessions. The session list MUST include all currently active NETCONF sessions.

```
session
    /session-id (key)
    /transport
    /username
    /source-host
    /login-time
    /in-rpcs
    /in-bad-rpcs
    /out-rpc-errors
    /out-notifications
```

session-id (uint32, 1..max)
Unique identifier for the session. This value is the NETCONF session identifier, as defined in [RFC4741].

transport (identityref, transport)
Identifies the transport for each session. The transport is represented as a YANG identity. This document defines the identities "netconf-ssh", "netconf-soap-over-beep", "netconf-soap-over-https", "netconf-beep", and "netconf-tls" (see Section 5).

username (string)
The username is the client identity that was authenticated by the NETCONF transport protocol. The algorithm used to derive the username is NETCONF transport protocol specific and in addition specific to the authentication mechanism used by the NETCONF transport protocol.

source-host (inet:host)
Host identifier (IP address or name) of the NETCONF client.

login-time (yang:date-and-time)
Time at the server at which the session was established.

in-rpcs (yang:zero-based-counter32)
Number of correct <rpc> messages received.

in-bad-rpcs (yang:zero-based-counter32)
Number of messages received when an <rpc> message was expected, that were not correct <rpc> messages. This includes XML parse errors and errors on the rpc layer.

out-rpc-errors (yang:zero-based-counter32)
Number of <rpc-reply> messages sent that contained an <rpc-error> element.

out-notifications (yang:zero-based-counter32)
Number of <notification> messages sent.

2.1.5. The /netconf-state/statistics Subtree

Statistical data pertaining to the NETCONF server.

```
statistics
    /netconf-start-time
    /in-bad-hellos
    /in-sessions
    /dropped-sessions
    /in-rpcs
    /in-bad-rpcs
    /out-rpc-errors
    /out-notifications
```

statistics:
Contains management-session-related performance data for the NETCONF server.

netconf-start-time (yang:date-and-time)
Date and time at which the management subsystem was started.

in-bad-hellos (yang:zero-based-counter32)
Number of sessions silently dropped because an invalid <hello> message was received.

in-sessions (yang:zero-based-counter32)
Number of sessions started.

dropped-sessions (yang:zero-based-counter32)
Number of sessions that were abnormally terminated, e.g., due to idle timeout or transport close.

in-rpcs (yang:zero-based-counter32)
Number of correct <rpc> messages received.

in-bad-rpcs (yang:zero-based-counter32)
Number of messages received when an <rpc> message was expected, which were not correct <rpc> messages.

out-rpc-errors (yang:zero-based-counter32)
Number of <rpc-reply> messages sent that contained an <rpc-error> element.

out-notifications (yang:zero-based-counter32)
Number of <notification> messages sent.

3. Schema Specific Operations

3.1. The <get-schema> Operation

Description:

This operation is used to retrieve a schema from the NETCONF server.

Parameters:

identifier (string):
Identifier for the schema list entry.
Mandatory parameter.

version (string):
Version of the schema requested.
Optional parameter.

format (identityref, schema-format):
The data modeling language of the schema.
Default value is 'yang' when not specified.
Optional parameter.

Positive Response:

The NETCONF server returns the requested schema.

Negative Response:

If the requested schema does not exist, the <error-tag> is 'invalid-value'.

If more than one schema matches the requested parameters, the <error-tag> is 'operation-failed', and <error-app-tag> is 'data-not-unique'.

4. Examples

4.1. Retrieving Schema List via <get> Operation

A NETCONF client retrieves the list of supported schema from a NETCONF server by retrieving the /netconf-state/schemas subtree via a <get> operation.

Available schema for the requesting session are returned in the reply containing the <identifier>, <version>, <format>, and <location> elements.

The response data can be used to determine the available schema and their versions. The schema itself (i.e., schema content) is not returned in the response. The optional <location> element contains a URI, which can be used to retrieve the schema by another protocol such as ftp [RFC0959] or http(s) [RFC2616] [RFC2818], or the special value 'NETCONF', which means that the schema can be retrieved from the device via the <get-schema> operation.

Example:

```
<rpc message-id="101"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter type="subtree">
      <netconf-state xmlns=
      "urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
        <schemas/>
      </netconf-state>
    </filter>
  </get>
</rpc>
```

The NETCONF server returns a list of schema available for retrieval.

```
<rpc-reply message-id="101"
           xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-state
    xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
      <schemas>
        <schema>
          <identifier>foo</identifier>
          <version>1.0</version>
          <format>xsd</format>
          <namespace>http://example.com/foo</namespace>
          <location>ftp://ftp.example.com/schemas/foo_1.0.xsd</location>
          <location>http://www.example.com/schema/foo_1.0.xsd</location>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>foo</identifier>
          <version>1.1</version>
          <format>xsd</format>
          <namespace>http://example.com/foo</namespace>
          <location>ftp://ftp.example.com/schemas/foo_1.1.xsd</location>
          <location>http://www.example.com/schema/foo_1.1.xsd</location>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>bar</identifier>
          <version>2008-06-01</version>
          <format>yang</format>
          <namespace>http://example.com/bar</namespace>
          <location>
            http://example.com/schema/bar@2008-06-01.yang
          </location>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>bar-types</identifier>
          <version>2008-06-01</version>
          <format>yang</format>
          <namespace>http://example.com/bar</namespace>
          <location>
            http://example.com/schema/bar-types@2008-06-01.yang
          </location>
          <location>NETCONF</location>
        </schema>
      </schemas>
    </netconf-state>
  </data>
</rpc-reply>
```

4.2. Retrieving Schema Instances

Given the reply in the previous section, the following examples illustrate the retrieval of 'foo', 'bar', and 'bar-types' schema at multiple locations, with multiple formats, and in multiple locations.

1. foo, version 1.0 in xsd format:

   a. Via FTP using location
      ftp://ftp.example.com/schemas/foo_1.0.xsd

   b. Via HTTP using location
      http://www.example.com/schema/foo_1.0.xsd

   c. Via <get-schema> using identifier, version, and
      format parameters.

```
<rpc message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-schema
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
    <identifier>foo</identifier>
    <version>1.0</version>
    <format>xsd</format>
  </get-schema>
</rpc>

<rpc-reply message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <!-- foo 1.0 xsd schema contents here -->
    </xs:schema>
  </data>
</rpc-reply>
```

2. bar, version 2008-06-01 in YANG format:

   a. Via HTTP using location
      http://example.com/schema/bar@2008-06-01.yang

   b. Via <get-schema> using identifier and version
      parameters:

```
<rpc message-id="102"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-schema
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
    <identifer>bar</identifer>
    <version>2008-06-01</version>
  </get-schema>
</rpc>

<rpc-reply message-id="102"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
    module bar {
      //default format (yang) returned
      //bar version 2008-06-01 yang module
      //contents here ...
    }
  </data>
</rpc-reply>
```

3. bar-types, version 2008-06-01 in default YANG format:

a. Via <get-schema> using identifer parameter:

```
<rpc message-id="103"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-schema
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
    <identifer>bar-types</identifer>
  </get-schema>
</rpc>

<rpc-reply message-id="103"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
    module bar-types {
      //default format (yang) returned
      //latest revision returned
      //is version 2008-06-01 yang module
      //contents here ...
    }
  </data>
</rpc-reply>
```

5. NETCONF Monitoring Data Model

The data model described in this memo is defined in the following YANG module.

This YANG module imports typedefs from [RFC6021] and references [RFC4741], [RFC4742], [RFC4743], [RFC4744], [RFC5539], [xmlschema-1], [RFC6020], [ISO/IEC19757-2:2008], and [RFC5717].

```
<CODE BEGINS> file "ietf-netconf-monitoring@2010-10-04.yang"

module ietf-netconf-monitoring {

  namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring";
  prefix "ncm";

  import ietf-yang-types { prefix yang; }
  import ietf-inet-types { prefix inet; }

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

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

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

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

     Editor:   Mark Scott
               <mailto:mark.scott@ericsson.com>

     Editor:   Martin Bjorklund
               <mailto:mbj@tail-f.com>";

  description
    "NETCONF Monitoring Module.
     All elements in this module are read-only.

     Copyright (c) 2010 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
     (http://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC 6022; see
     the RFC itself for full legal notices.";

  revision 2010-10-04 {
    description
      "Initial revision.";
    reference
      "RFC 6022: YANG Module for NETCONF Monitoring";
  }

  typedef netconf-datastore-type {
    type enumeration {
      enum running;
      enum candidate;
      enum startup;
    }
    description
      "Enumeration of possible NETCONF datastore types.";
    reference
      "RFC 4741: NETCONF Configuration Protocol";
  }

  identity transport {
    description
      "Base identity for NETCONF transport types.";
  }

  identity netconf-ssh {
    base transport;
    description
      "NETCONF over Secure Shell (SSH).";
    reference
      "RFC 4742: Using the NETCONF Configuration Protocol
                 over Secure SHell (SSH)";
  }

  identity netconf-soap-over-beep {
    base transport;
    description
      "NETCONF over Simple Object Access Protocol (SOAP) over
       Blocks Extensible Exchange Protocol (BEEP).";
```

```
    reference
      "RFC 4743: Using NETCONF over the Simple Object
                 Access Protocol (SOAP)";
  }

  identity netconf-soap-over-https {
    base transport;
    description
      "NETCONF over Simple Object Access Protocol (SOAP)
      over Hypertext Transfer Protocol Secure (HTTPS).";
    reference
      "RFC 4743: Using NETCONF over the Simple Object
                 Access Protocol (SOAP)";
  }

  identity netconf-beep {
    base transport;
    description
      "NETCONF over Blocks Extensible Exchange Protocol (BEEP).";
    reference
      "RFC 4744: Using the NETCONF Protocol over the
                 Blocks Extensible Exchange Protocol (BEEP)";
  }

  identity netconf-tls {
    base transport;
    description
      "NETCONF over Transport Layer Security (TLS).";
    reference
      "RFC 5539: NETCONF over Transport Layer Security (TLS)";
  }

  identity schema-format {
    description
      "Base identity for data model schema languages.";
  }

  identity xsd {
    base schema-format;
    description
      "W3C XML Schema Definition.";
    reference
      "W3C REC REC-xmlschema-1-20041028:
         XML Schema Part 1: Structures";
  }
```

```
  identity yang {
    base schema-format;
    description
      "The YANG data modeling language for NETCONF.";
    reference
      "RFC 6020:  YANG - A Data Modeling Language for the
                  Network Configuration Protocol (NETCONF)";
  }

  identity yin {
    base schema-format;
    description
      "The YIN syntax for YANG.";
    reference
      "RFC 6020:  YANG - A Data Modeling Language for the
                  Network Configuration Protocol (NETCONF)";
  }

  identity rng {
    base schema-format;
    description
      "Regular Language for XML Next Generation (RELAX NG).";
    reference
      "ISO/IEC 19757-2:2008: RELAX NG";
  }

  identity rnc {
    base schema-format;
    description
      "Relax NG Compact Syntax";
    reference
      "ISO/IEC 19757-2:2008: RELAX NG";
  }

  grouping common-counters {
    description
      "Counters that exist both per session, and also globally,
       accumulated from all sessions.";

    leaf in-rpcs {
      type yang:zero-based-counter32;
      description
        "Number of correct <rpc> messages received.";
    }
    leaf in-bad-rpcs {
      type yang:zero-based-counter32;
```

```
      description
        "Number of messages received when an <rpc> message was expected,
         that were not correct <rpc> messages.  This includes XML parse
         errors and errors on the rpc layer.";
    }
    leaf out-rpc-errors {
      type yang:zero-based-counter32;
      description
        "Number of <rpc-reply> messages sent that contained an
         <rpc-error> element.";
    }
    leaf out-notifications {
      type yang:zero-based-counter32;
      description
        "Number of <notification> messages sent.";
    }
  }

  container netconf-state {
    config false;
    description
      "The netconf-state container is the root of the monitoring
       data model.";

    container capabilities {
      description
        "Contains the list of NETCONF capabilities supported by the
         server.";

      leaf-list capability {
        type inet:uri;
        description
          "List of NETCONF capabilities supported by the server.";
      }
    }

    container datastores {
      description
        "Contains the list of NETCONF configuration datastores.";

      list datastore {
        key name;
        description
          "List of NETCONF configuration datastores supported by
           the NETCONF server and related information.";

        leaf name {
          type netconf-datastore-type;
```

```
      description
        "Name of the datastore associated with this list entry.";
    }
    container locks {
      presence
        "This container is present only if the datastore
         is locked.";
      description
        "The NETCONF <lock> and <partial-lock> operations allow
         a client to lock specific resources in a datastore.  The
         NETCONF server will prevent changes to the locked
         resources by all sessions except the one that acquired
         the lock(s).

         Monitoring information is provided for each datastore
         entry including details such as the session that acquired
         the lock, the type of lock (global or partial) and the
         list of locked resources.  Multiple locks per datastore
         are supported.";

      grouping lock-info {
        description
          "Lock related parameters, common to both global and
           partial locks.";

        leaf locked-by-session {
          type uint32;
          mandatory true;
          description
            "The session ID of the session that has locked
             this resource.  Both a global lock and a partial
             lock MUST contain the NETCONF session-id.

             If the lock is held by a session that is not managed
             by the NETCONF server (e.g., a CLI session), a session
             id of 0 (zero) is reported.";
          reference
            "RFC 4741: NETCONF Configuration Protocol";
        }
        leaf locked-time {
          type yang:date-and-time;
          mandatory true;
          description
            "The date and time of when the resource was
             locked.";
        }
      }
```

```
          choice lock-type {
            description
              "Indicates if a global lock or a set of partial locks
               are set.";

            container global-lock {
              description
                "Present if the global lock is set.";
              uses lock-info;
            }

            list partial-lock {
              key lock-id;
              description
                "List of partial locks.";
              reference
                "RFC 5717: Partial Lock Remote Procedure Call (RPC) for
                           NETCONF";

              leaf lock-id {
                type uint32;
                description
                  "This is the lock id returned in the <partial-lock>
                   response.";
              }
              uses lock-info;
              leaf-list select {
                type yang:xpath1.0;
                min-elements 1;
                description
                  "The xpath expression that was used to request
                   the lock.  The select expression indicates the
                   original intended scope of the lock.";
              }
              leaf-list locked-node {
                type instance-identifier;
                description
                  "The list of instance-identifiers (i.e., the
                   locked nodes).

                   The scope of the partial lock is defined by the list
                   of locked nodes.";
              }
            }
          }
        }
      }
    }
```

```
    container schemas {
      description
        "Contains the list of data model schemas supported by the
         server.";

      list schema {
        key "identifier version format";

        description
          "List of data model schemas supported by the server.";

        leaf identifier {
          type string;
          description
            "Identifier to uniquely reference the schema.  The
             identifier is used in the <get-schema> operation and may
             be used for other purposes such as file retrieval.

             For modeling languages that support or require a data
             model name (e.g., YANG module name) the identifier MUST
             match that name.  For YANG data models, the identifier is
             the name of the module or submodule.  In other cases, an
             identifier such as a filename MAY be used instead.";
        }
        leaf version {
          type string;
          description
            "Version of the schema supported.  Multiple versions MAY be
             supported simultaneously by a NETCONF server.  Each
             version MUST be reported individually in the schema list,
             i.e., with same identifier, possibly different location,
             but different version.

             For YANG data models, version is the value of the most
             recent YANG 'revision' statement in the module or
             submodule, or the empty string if no 'revision' statement
             is present.";
        }
        leaf format {
          type identityref {
            base schema-format;
          }
          description
            "The data modeling language the schema is written
             in (currently xsd, yang, yin, rng, or rnc).
```

```
            For YANG data models, 'yang' format MUST be supported and
            'yin' format MAY also be provided.";
        }
        leaf namespace {
          type inet:uri;
          mandatory true;
          description
            "The XML namespace defined by the data model.

            For YANG data models, this is the module's namespace.
            If the list entry describes a submodule, this field
            contains the namespace of the module to which the
            submodule belongs.";
        }
        leaf-list location {
          type union {
            type enumeration {
              enum "NETCONF";
            }
            type inet:uri;
          }
          description
            "One or more locations from which the schema can be
             retrieved.  This list SHOULD contain at least one
             entry per schema.

             A schema entry may be located on a remote file system
             (e.g., reference to file system for ftp retrieval) or
             retrieved directly from a server supporting the
             <get-schema> operation (denoted by the value 'NETCONF').";
        }
      }
    }
```

```
    container sessions {
      description
        "The sessions container includes session-specific data for
         NETCONF management sessions.  The session list MUST include
         all currently active NETCONF sessions.";

      list session {
        key session-id;
        description
          "All NETCONF sessions managed by the NETCONF server
           MUST be reported in this list.";

        leaf session-id {
          type uint32 {
            range "1..max";
          }
          description
            "Unique identifier for the session.  This value is the
             NETCONF session identifier, as defined in RFC 4741.";
          reference
            "RFC 4741: NETCONF Configuration Protocol";
        }
        leaf transport {
          type identityref {
            base transport;
          }
          mandatory true;
          description
            "Identifies the transport for each session, e.g.,
            'netconf-ssh', 'netconf-soap', etc.";
        }
        leaf username  {
          type string;
          mandatory true;
          description
            "The username is the client identity that was authenticated
            by the NETCONF transport protocol.  The algorithm used to
            derive the username is NETCONF transport protocol specific
            and in addition specific to the authentication mechanism
            used by the NETCONF transport protocol.";
        }
        leaf source-host {
          type inet:host;
          description
            "Host identifier of the NETCONF client.  The value
             returned is implementation specific (e.g., hostname,
             IPv4 address, IPv6 address)";
        }
```

```
        leaf login-time {
          type yang:date-and-time;
          mandatory true;
          description
            "Time at the server at which the session was established.";
        }
        uses common-counters {
          description
            "Per-session counters.  Zero based with following reset
             behaviour:
               - at start of a session
               - when max value is reached";
        }
      }
    }

    container statistics {
      description
        "Statistical data pertaining to the NETCONF server.";

      leaf netconf-start-time {
        type yang:date-and-time;
        description
          "Date and time at which the management subsystem was
           started.";
      }
      leaf in-bad-hellos {
        type yang:zero-based-counter32;
        description
          "Number of sessions silently dropped because an
          invalid <hello> message was received.  This includes <hello>
          messages with a 'session-id' attribute, bad namespace, and
          bad capability declarations.";
      }
      leaf in-sessions {
        type yang:zero-based-counter32;
        description
          "Number of sessions started.  This counter is incremented
           when a <hello> message with a <session-id> is sent.

          'in-sessions' - 'in-bad-hellos' =
              'number of correctly started netconf sessions'";
      }
      leaf dropped-sessions {
        type yang:zero-based-counter32;
```

```
        description
          "Number of sessions that were abnormally terminated, e.g.,
           due to idle timeout or transport close.  This counter is not
           incremented when a session is properly closed by a
           <close-session> operation, or killed by a <kill-session>
           operation.";
      }
      uses common-counters {
        description
          "Global counters, accumulated from all sessions.
           Zero based with following reset behaviour:
             - re-initialization of NETCONF server
             - when max value is reached";
      }
    }
  }

  rpc get-schema {
    description
      "This operation is used to retrieve a schema from the
       NETCONF server.

       Positive Response:
         The NETCONF server returns the requested schema.

       Negative Response:
         If requested schema does not exist, the <error-tag> is
         'invalid-value'.

         If more than one schema matches the requested parameters, the
         <error-tag> is 'operation-failed', and <error-app-tag> is
         'data-not-unique'.";

    input {
      leaf identifier {
        type string;
        mandatory true;
        description
          "Identifier for the schema list entry.";
      }
      leaf version {
        type string;
        description
          "Version of the schema requested.  If this parameter is not
           present, and more than one version of the schema exists on
           the server, a 'data-not-unique' error is returned, as
           described above.";
      }
```

```
      leaf format {
        type identityref {
          base schema-format;
        }
        description
           "The data modeling language of the schema.  If this
            parameter is not present, and more than one formats of
            the schema exists on the server, a 'data-not-unique' error
            is returned, as described above.";
      }
    }
    output {
        anyxml data {
          description
            "Contains the schema content.";
      }
    }
  }
}

<CODE ENDS>
```

6. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC4741]. The lowest NETCONF layer is the secure transport layer and the mandatory to implement secure transport is SSH [RFC4742].

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes.

These are the containers, list nodes, and data nodes with their specific sensitivity/vulnerability:

/netconf-state/sessions/session/username: Contains identity information that could be used in an attempt to authenticate with the server.

This username is only meant for monitoring, and SHOULD NOT be used for other purposes, such as access control, without a detailed discussion of the limitations of this reported username. For example, it is possible that server A and server B might report the same username, but these might be for different persons.

7. Acknowledgements

The authors would like to thank Andy Bierman, Mehmet Ersue, Washam Fan, David Harrington, Balazs Lengyel, Hideki Okita, Juergen Schoenwaelder, Bert Wijnen, and many other members of the NETCONF WG for providing important input to this document. The authors would also like to specifically acknowledge Sharon Chisholm's work on "NETCONF Monitoring Schema" [NETCONF] and contribution to this document.

8. IANA Considerations

This document registers one URI in "The IETF XML Registry". Following the format in [RFC3688], the following has been registered.

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

This document registers one module in the "YANG Module Names" registry. Following the format in [RFC6020], the following has been registered.

```
name: ietf-netconf-monitoring
namespace: urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring
prefix: ncm
reference: RFC 6022
```

9. References

9.1. Normative References

[ISO/IEC19757-2:2008] ISO/IEC, "Document Schema Definition Language (DSDL) -- Part 2: Regular-grammar-based validation -- RELAX NG", December 2008, <http://www.iso.org/iso/catalogue_detail.htm?csnumber=37605>.

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

[RFC4741] Enns, R., "NETCONF Configuration Protocol", RFC 4741, December 2006.

[RFC4742] Wasserman, M. and T. Goddard, "Using the NETCONF Configuration Protocol over Secure SHell (SSH)", RFC 4742, December 2006.

[RFC4743] Goddard, T., "Using NETCONF over the Simple Object Access Protocol (SOAP)", RFC 4743, December 2006.

[RFC4744] Lear, E. and K. Crozier, "Using the NETCONF Protocol over the Blocks Extensible Exchange Protocol (BEEP)", RFC 4744, December 2006.

[RFC5539] Badra, M., "NETCONF over Transport Layer Security (TLS)", RFC 5539, May 2009.

[RFC5717] Lengyel, B. and M. Bjorklund, "Partial Lock Remote Procedure Call (RPC) for NETCONF", RFC 5717, December 2009.

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

[RFC6021] Schoenwaelder, J., Ed., "Common YANG Data Types", October 2010.

[XML-NAMES] Hollander, D., Tobin, R., Thompson, H., Bray, T., and A. Layman, "Namespaces in XML 1.0 (Third Edition)", World Wide Web Consortium Recommendation REC-xml-names-20091208, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208>.

[xmlschema-1] Biron, Paul V. and Ashok. Malhotra, "XML Schema Part 1: Structures Second Edition W3C Recommendation 28 October 2004", October 2004, <http://www.w3.org/TR/xmlschema-1>.

9.2. Informative References

[NETCONF] Chisholm, S. and H. Trevino, "NETCONF Monitoring Schema", Work in Progress, February 2007.

[RFC0959] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, October 1985.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

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

Authors' Addresses

Mark Scott
Ericsson
3500 Carling Ave
Nepean, Ontario K2H 8E9
Canada

EMail: mark.scott@ericsson.com

Martin Bjorklund
Tail-f Systems
Klara Norra Kyrkogata 31
SE-111 22 Stockholm,
Sweden

EMail: mbj@tail-f.com