netconf
Internet-Draft
Intended status: Experimental
Expires: December 22, 2007

Y. Li
Huawei Technologies
D. Harrington
Huawei Technologies (USA)

June 20, 2007

Accessing MIBs using NETCONF
draft-li-ngo-access-mib-01

Abstract

This memo describes a simple mechanism for accessing the Management Information Base (MIB), using the existing NETCONF RPC infrastructure. It defines a set of specific operations for accessing the MIB as a capability of NETCONF.

Table of Contents

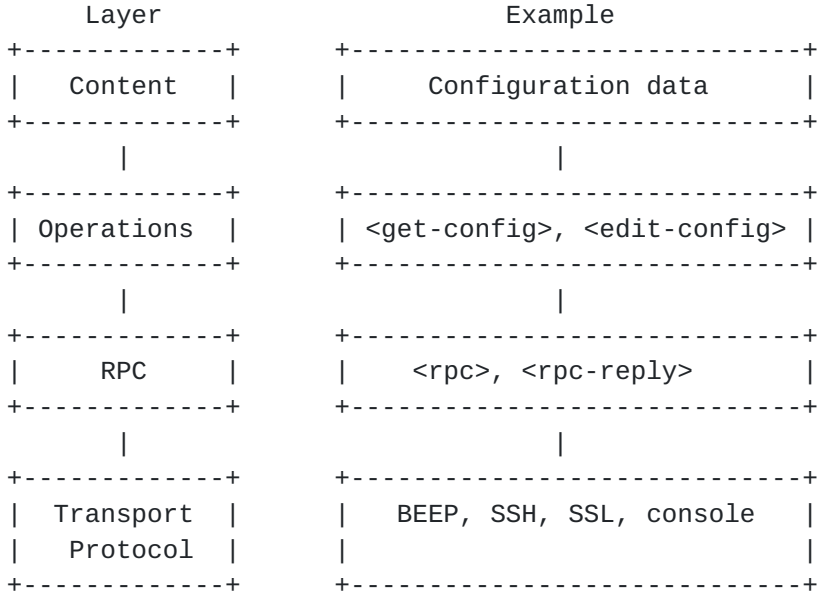
1. Introduction

This memo describes a simple mechanism for accessing the Management Information Base (MIB), using the existing NETCONF RPC infrastructure. It defines a set of specific operations for accessing the MIB as a capability of NETCONF.

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 RFC 2119 [RFC2119].

1.1. Motivation

NETCONF [RFC4741] conceptually partitions its design into four layers:



Transport Protocol, RPC and Operations layers have already been standardized to a degree, while the Content Layer has not been standardized yet. Device vendors have been asked to support NETCONF in their products, but there is no standard data definition language or standard content for NETCONF at this time.

SMIv2 [RFC2578] is the IETF-standard for network management data modeling, and the MIB is the IETF standard for organizing MIB modules. There are many standard MIB modules, and these have been supplemented by many proprietary MIB modules defined by device vendors. Many devices already support a wide range of MIB modules, and many network management applications already utilize the information modeled in the available MIB modules.

MIB objects are generally accessed through the Simple Network Management Protocol (SNMP), but RFC1052 [RFC1052] documents the IAB decision to use the MIB as the IETF standard for data modeling, and the intention of the IETF to permit the MIB to be utilized by multiple protocols.

The netconf community wants to develop a data modeling language with structures more complex than SMIv2 permits, and to have the netconf protocol support task-oriented RPCs to perform operations, rather than a data-oriented protocol, such as SNMP, that has a small number of verbs that are designed to directly manipulate data elements.

While the netconf community works on developing its data modeling language and data models, the netconf community has recommended that vendors simply use their own proprietary data models, written in an XML format. This approach is a problem for device vendors. They do not know what configuration data they should provide; if they use proprietary data, the customer desire for interoperability between devices and applications from different vendors cannot be met. If the IETF standardizes an XML-based data model in the near future, then there will be little return on the investment for developing a proprietary XML-based data model.

The MIB is a standard data model that most internetworking device vendors already support to provide interoperable management for their products. Vendors could use the MIB, expressed in XML, as one type of data model for NETCONF. The libsmi tools already have an option to convert MIB modules into corresponding XML Schemas, so this will make it easier to use MIB data with netconf.

This memo proposes a set of RPCs, packaged as a capability, for NETCONF to access MIB objects.

2. XML schemas for MIB definitions

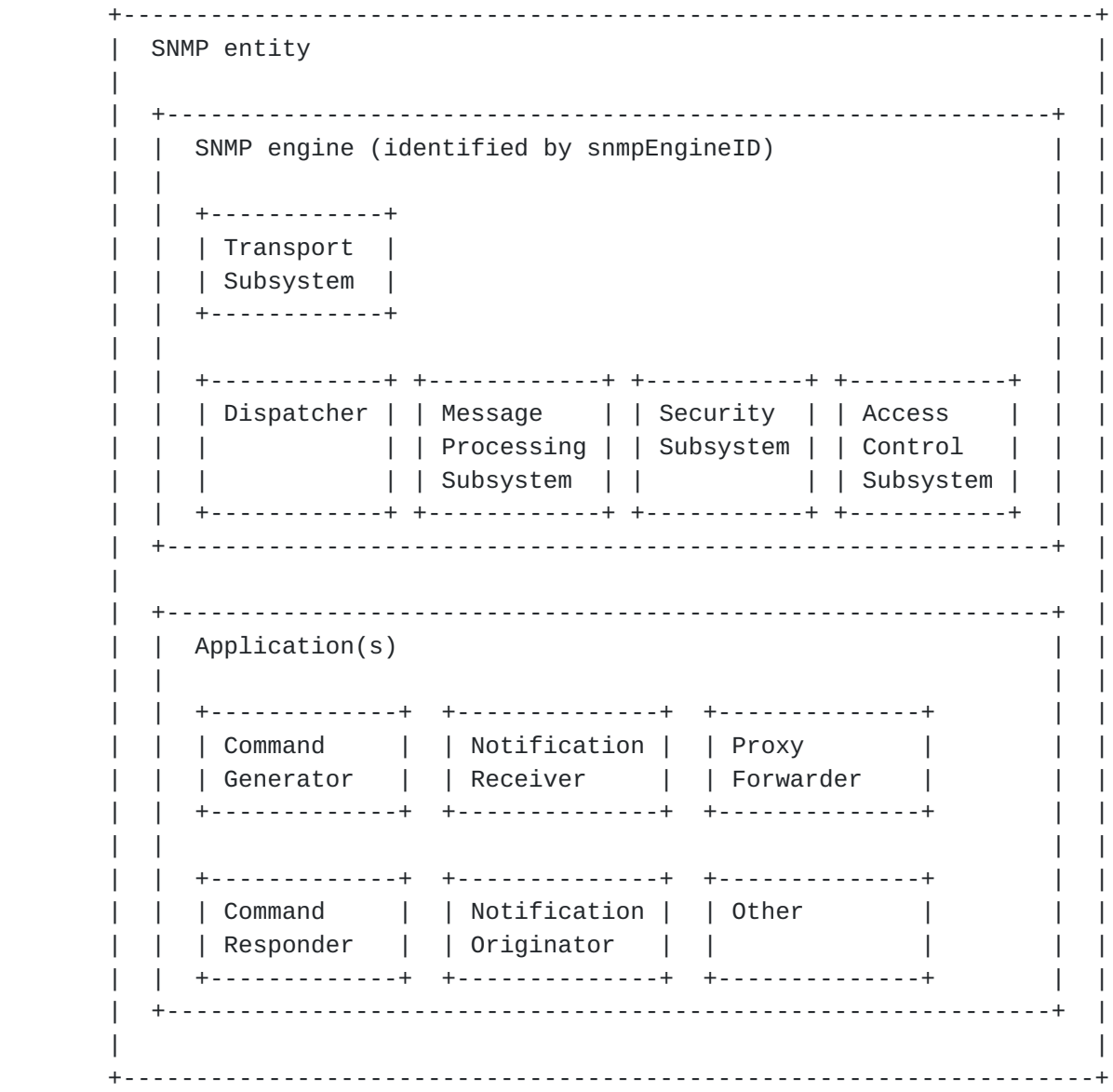
MIB modules are written using an adapted subset of Abstract Syntax Notation One, ASN.1 [refs.asn1] (1988), which is called the Structure of Management Information (SMI). NETCONF encodes configuration data in XML [refs.xml], and the data model of NETCONF should be described using XML DTD or XML schema [refs.schema] or other XML schema language. At the time of this writing, XML schema is preferred. Those MIB modules to be used with NETCONF should be rewritten in XSD, and for interoperability the resulting XML schema written by different people should have the same format.

The libsmi toolkit contains an smidump tool, which can transform a MIB module into various formats, including an XML schema. The libsmi toolkit has been available for many years, and a number of

applications already use the smidump XML schemas. For the sake of compatibility, this memo also uses XML schemas generated by smidump. The "Using Smidump to Convert MIB to XSD" [refs.mib-convert] expounds how the smidump converts a MIB to an XSD.

3. How This Capability Fits with the SNMP Architecture

RFC3411 provides an architecture for describing SNMP frameworks. There are similarities between the netconf architecture and the RFC3411 architecture.



The Netconf Protocol describes the message format for netconf, which is similar to a model in the message processing subsystem described

in RFC3412 [RFC3412]. The transport/application mappings of netconf are similar to transport models in the Transport Subsystem for SNMP, as described in [I-D.ietf-isms-tmsm].

Having the ability to access the MIB data is similar to an SNMP application, as described in RFC3413 [RFC3413]. However, an "application" in the SNMP architecture requires input parameters, for data addressing, and for access control purposes.

3.1. Data Addressing

Within SNMPv3, an individual item of information within an administrative domain is addressed by a four-tuple consisting of a contextEngineID, a contextName, an object type, and an object instance. The last two elements are typically encoded in an object identifier (OID) value. As of this writing, Netconf does not provide the equivalent of this four-tuple. This memo uses an XML-based address that encodes the object type and object instance, derived from the equivalent Object Identifier.

An SNMP version 3 message [RFC3412] carries a scopedPDU - a data structure containing a contextEngineID, a contextName, and a protocol data unit (PDU) which contains MIB data. The smidump schemas always have a <context> element that includes mandatory attributes to identify the SNMP context of the MIB data.

3.2. Access Control

Within SNMPv3, an access control model determines who is allowed to perform which operations to different data sets of information. The RFC3411 architecture describes a series of parameters that are provided to the isAccessAllowed() service primitive to make this determination. The parameters include a securityModel and securityName to identify the (possibly-authenticated) principal

```
statusInformation =              -- success or errorIndication
    isAccessAllowed(
    IN   securityModel             -- Security Model in use
    IN   securityName              -- principal who wants to access
    IN   securityLevel             -- Level of Security
    IN   viewType                  -- read, write, or notify view
    IN   contextName               -- context containing variableName
    IN   variableName              -- OID for the managed object
         )
```

A securityModel is the model within the SNMP security subsystem that will provide authentication, integrity checking, encryption, and

other security services for SNMP response messages. Netconf provides these services through its transport/application mappings. Netconf does not provide an indicator of which transport mapping will be utilized to secure the response message.

The securityLevel identifies whether authentication and encryption services will be used on the response message. Netconf leaves the configuration of authentication and encryption services to the human administrator, and does not provide a mechanism-independent indicator for the security services provided during transport. Implicit in this design is the assumption that a transport mapping provides a constant level of security. Without an indicator of the services provided, access control for netconf cannot be determined based on the security properties of the message transport.

A securityName is a mechanism-independent identifier of the "who" on whose behalf SNMP services are provided or processing takes place. Netconf provides for the authentication of the "who" on whose behalf services are provided or processing takes place, but does not provide a mechanism-independent identifier to pass the identity to an access control mechanism.

The viewType is an abstraction for a class of operation; SNMP supports read, write, and notify types of operations. Netconf does not yet have a clear description of the types of operations that may be supported in the future. When accessing MIB data with the operations described in this memo, a read/write/notify classification of operations can be used.

As described above, the smidump schemas provide mechanisms to identify the contextName (context) and instance identifier (variableName).

4. Accessing MIB Capability

4.1. Description

The :accessing-mib capability indicates that the server supports MIB definitions as data models, and supports retrieving and modifying MIB objects in the <running> configuration through the operations specified in this memo.

4.2. Dependencies

none.

4.3. Capability Identifier

The :accessing-mib capability is identified by the following capability string:

urn:ietf:params:netconf:capability:accessing-mib:1.0

4.4. New Operations

4.4.1. General NETCONF Operations vs SNMP-specific Operations

Although the semantics of <mib-get> and <mib-set> operations which are proposed in this memo are closely similar to of <get> and <edit-config> operations in NETCONF, this memo does not intend to reuse the operations in NETCONF. Because the <mib-get> and <mib-set> operations could be freely extended to support the SNMP-specific features, for example, the view-based access control model, without changing existing NETCONF operations.

4.4.2. <mib-get>

Description:

Retrieve MIB objects from the <running> configuration datastore.

Attributes:

max-repetitions:

The attribute is only applied to subtree filtering. Entry elements in the <config> subtree, such as <ifEntry> and <tcpConnEntry>, may contain a "max-repetitions" attribute that indicates the maximum number of element instances that the client wants to retrieve. Starting with the given index of the entry, the server retrieves an entry instance in lexicographical order until either of following conditions occurs:

+ The retrieved number of the entry element instances has reached the value of "max-repetitions" attribute.
+ No more entry element instances are available.

Parameters:

filter:

This parameter specifies the portion of the MIB objects to retrieve. If this parameter is empty, all MIB objects are returned.

The filter element may optionally contain a 'type' attribute. This attribute indicates the type of filtering syntax used within the filter element. The default filtering mechanism in

this memo is referred to as subtree filtering and is described in [RFC4741] Section 6. The value 'subtree' explicitly identifies this type of filtering.

If the NETCONF peer supports the :xpath capability ([RFC4741] Section 8.9), the value 'xpath' may be used to indicate that the filter element contains an XPath expression.

Positive Response:

If the device was able to satisfy the request, an <rpc-reply> is sent. The <data> section contains the appropriate subset.

Negative Response:

An <rpc-error> element is included in the <rpc-reply> if the request cannot be completed for any reason. [todo: detail the error conditions, and the rpc-error codes that might be returned.]

Example:

Get the value of ifNumber, ifMtu.1 and ifSpeed.1.

```
<rpc message-id="101"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:mib="urn:ietf:params:xml:ns:netconf:mib:1.0">
  <mib:mib-get>
    <filter type="subtree">
      <snmp-data
        xmlns="http://www.ibr.cs.tu-bs.de/projects/libsmi/xsd/IF-MIB">
        <context ipaddr="192.0.2.1" port="830"
            community="public" time="2006-12-05T08:08:08Z">
          <interfaces>
            <ifNumber/>
          </interfaces>
          <ifEntry ifIndex="1">
            <ifType/>
            <ifMtu/>
          </ifEntry>
        </context>
      </snmp-data>
    </filter>
  </mib:mib-get>
</rpc>

<rpc-reply message-id="101"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <snmp-data
        xmlns="http://www.ibr.cs.tu-bs.de/projects/libsmi/xsd/IF-MIB">
      <context ipaddr="192.0.2.1" port="830"
           community="public" time="2006-12-05T08:08:08Z">
        <interfaces>
          <ifNumber>10</ifNumber>
        </interfaces>
        <ifEntry ifIndex="1">
          <ifType>ethernetCsmacd</ifType>
          <ifMtu>1500</ifMtu>
        </ifEntry>
      </context>
    </snmp-data>
  </data>
</rpc-reply>
```

Get the values of ifMtu and ifSpeed of first three ifEntries.

```
<rpc message-id="101"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:mib="urn:ietf:params:xml:ns:netconf:mib:1.0">
  <mib:mib-get>
    <filter type="sub-tree">
      <snmp-data
        xmlns="http://www.ibr.cs.tu-bs.de/projects/libsmi/xsd/IF-MIB">
        <context ipaddr="192.0.2.1" port="830"
            community="public" time="2006-12-05T08:08:08Z">
          <ifEntry ifIndex="1" mib:max-repetitions="3">
            <ifType/>
            <ifMtu/>
          </ifEntry>
        </context>
      </snmp-data>
    </filter>
  </mib:mib-get>
</rpc>

<rpc-reply message-id="101"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <snmp-data
      xmlns="http://www.ibr.cs.tu-bs.de/projects/libsmi/xsd/IF-MIB">
      <context ipaddr="192.0.2.1" port="830"
          community="public" time="2006-12-05T08:08:08Z">
        <ifEntry ifIndex="1">
          <ifType>ethernetCsmacd</ifType>
          <ifMtu>1500</ifMtu>
        </ifEntry>
        <ifEntry ifIndex="2">
          <ifType>ethernetCsmacd</ifType>
          <ifMtu>1500</ifMtu>
        </ifEntry>
        <ifEntry ifIndex="3">
          <ifType>ppp</ifType>
          <ifMtu>1800</ifMtu>
        </ifEntry>
      </context>
    </snmp-data>
  </data>
</rpc-reply>
```

4.4.3. <mib-set>

Description:

Set MIB objects in the <running> configuration datastore.

Attributes:

operation:

Elements in the <config> subtree may contain an "operation" attribute. The attribute specifies that MIB objects in the subtree whose root is identified by the element containing this attribute perform the operation. The attribute MAY appear on multiple elements throughout the <config> subtree. If a MIB object locates in multiple overlapped subtrees, which are specified different operations, the MIB object perform the operation of the smallest subtree.

If the operation attribute doesn't appear in the <config> subtree, all MIB objects in the subtree perform replace operation.

The operation attribute has one of the following values:

replace: The MIB objects in the subtree whose root is identified by the element containing this attribute replaces any related MIB objects in the <running> configuration datastore. This is the default behavior.

create: This value is only applied for an entry element. It indicates that create an entry instance as the entry element described.

delete: This value is only applied for an entry element. It indicates that delete an entry instance as the entry element described.

Although this memo explicitly use the "create" or "delete" verb as a value of an "operation" attribute of configuration data in a <mib-set> request, the the RowStatus object is still needed. Because the RowStatus object is able to temporarily make a table entry instance invalid instead of deleting it. Creating and deleting a table entry instance can use either the RowStatus object or the "operation" attribute.

Parameters:

test-option:

The test-option element may be specified only if the device advertises the :validate capability (Section 8.6).

The test-option element has one of the following values:

test-then-set: Perform a validation test before attempting to set. If validation errors occur, do not perform the <mib-set> operation. This is the default test-option.

set: Perform a set without a validation test first.

error-option:

The error-option element has one of the following values:

stop-on-error: Abort the edit-config operation on first error. This is the default error-option.

continue-on-error: Continue to process MIB objects on error; error is recorded and negative response is generated if any errors occur.

rollback-on-error: If an error condition occurs such that an error severity <rpc-error> element is generated, the server will stop processing the edit-config operation and restore the specified configuration to its complete state at the start of this edit-config operation. This option requires the server to support the :rollback-on-error capability described in [RFC4741] Section 8.5.

config:

A portion of the MIB tree as defined by a MIB module in XML schema. The contents MUST be placed in an appropriate namespace, to allow the device to detect the appropriate MIB module.

Positive Response:

If the device was able to satisfy the request, an <rpc-reply> is sent containing an <ok> element.

Negative Response:

An <rpc-error> response is sent if the request cannot be completed for any reason.

Example:

Set ifLinkUpDownTrapEnable.1 to "enable" and ifAlias.2 to "Beijing".

```
<rpc message-id="101"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:mib="urn:ietf:params:xml:ns:netconf:mib:1.0">
  <mib:mib-set>
    <config>
      <snmp-data
        xmlns="http://www.ibr.cs.tu-bs.de/projects/libsmi/xsd/IF-MIB">
        <context ipaddr="192.0.2.1" port="830"
            community="public" time="2006-12-05T08:08:08Z">
          <ifEntry ifIndex="1">
            <ifLinkUpDownTrapEnable>enable</ifLinkUpDownTrapEnable>
            <ifAlias>Beijing</ifAlias>
          </ifEntry>
        </context>
      </snmp-data>
    </config>
  </mib:mib-set>
</rpc>

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

Create an ifStackEntry instance.

```
<rpc message-id="101"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:mib="urn:ietf:params:xml:ns:netconf:mib:1.0">
  <mib:mib-set>
    <snmp-data
      xmlns="http://www.ibr.cs.tu-bs.de/projects/libsmi/xsd/IF-MIB">
      <context ipaddr="192.0.2.1" port="830"
          community="public" time="2006-12-05T08:08:08Z">
        <ifStackEntry ifStackHigherLayer="1" ifStackLowerLayer="2"
                      mib:operation="create"/>
      </context>
    </snmp-data>
  </mib:mib-set>
</rpc>

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

5. Schema for Accessing-mib Capability

```
 <?xml version="1.0" encoding="UTF-8"?>
 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns="urn:ietf:params:xml:ns:netconf:mib:1.0"
            xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0"
            targetNamespace="urn:ietf:params:xml:ns:netconf:mib:1.0"
            elementFormDefault="qualified"
            attributeFormDefault="unqualified"
            xml:lang="en">
   <xs:annotation>
     <xs:documentation>
       This schema defines elements for accessing-mib capability.
     </xs:documentation>
   </xs:annotation>

   <!-- import standard XML definitions -->
   <xs:import namespace="http://www.w3.org/XML/1998/namespace"
         schemaLocation="http://www.w3.org/2001/xml.xsd"/>
   <!-- import base netconf definitions -->
   <xs:import namespace="urn:ietf:params:xml:ns:netconf:base:1.0"
         schemaLocation="./netconf.xsd"/>

   <!-- <mib-set> operation -->
```

```
      <xs:simpleType name="setOperationType">
        <xs:restriction base="xs:string">
          <xs:enumeration value="replace"/>
          <xs:enumeration value="create"/>
          <xs:enumeration value="delete"/>
        </xs:restriction>
      </xs:simpleType>
      <xs:attribute name="operation"
                    type="setOperationType" default="replace"/>
      <xs:complexType name="mibSetType">
        <xs:complexContent>
          <xs:extension base="netconf:rpcOperationType">
            <xs:sequence>
              <xs:annotation>
                <xs:documentation>
                  Use of the test-option element requires the
                  :validate capability.
                </xs:documentation>
              </xs:annotation>
              <xs:element name="test-option"
                          type="netconf:testOptionType"
                          minOccurs="0"/>
              <xs:element name="error-option"
                          type="netconf:errorOptionType"
                          minOccurs="0"/>
              <xs:element name="config"
                          type="netconf:configInlineType"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="mib-set" type="mibSetType"
                  substitutionGroup="netconf:rpcOperation"/>
      <!-- <mib-get> operation -->
      <xs:attribute name="max-repetitions"
                    type="xs:positiveInteger" default="1"/>
      <xs:complexType name="mibGetType">
        <xs:complexContent>
          <xs:extension base="netconf:rpcOperationType">
            <xs:sequence>
              <xs:element name="filter"
                          type="netconf:filterInlineType"
                          minOccurs="0"/>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:element name="mib-get" type="mibGetType"
```

```
                substitutionGroup="netconf:rpcOperation"/>
    </xs:schema>
```

6. Security Considerations

Although this document provides Netconf RPCs to emulate SNMP operations to access MIBs, the Netconf protocol does not currently have a way to provide the necessary parameters to apply SNMP-based access controls. These parameters are discussed in Section 3.2.

If implementations are based on implementations of SNMP, then these parameters could be provided in an implementation-dependent manner. For compatibility, and balanced security, it is RECOMMENDED that such implementations provide MIB data access control comparable to the View-based Access Control model defined in [RFC3415].

7. IANA Considerations

8. References

8.1. Normative References

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

[refs.xml] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0", W3C XML, February 1998, <http://www.w3.org/TR/1998/REC-xml-19980210>.

[refs.schema] Fallside, D., Walmsley, P., Thompson, H., Beech, D., Maloney, M., and N. Mendelsohn, "XML Schema 1.0", W3C XML Schema, October 2004, <http://www.w3.org/XML/Schema>.

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

[refs.libsmi] Strauss, F. and T. Klie, "Integrating SNMP Agents with XML-based Management Systems", July 2004, <http://www.comsoc.org/livepubs/ci1/public/2004/jul/index.html>.

[refs.asn1] International Organization for Standardization, "Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)", December 1987.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC3412] Case, J., Harrington, D., Presuhn, R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3412, December 2002.

[RFC3413] Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications", STD 62, RFC 3413, December 2002.

[RFC3415] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.

[I-D.ietf-isms-tmsm] Harrington, D. and J. Schoenwaelder, "Transport Mapping Security Model (TMSM) Architectural Extension for the Simple Network Management Protocol (SNMP)", draft-ietf-isms-tmsm-08 (work in progress), May 2006.

8.2. Informative References

[RFC1052] Cerf, V., "IAB recommendations for the development of Internet network management standards", RFC 1052, April 1988.

[refs.mib-convert] Li, Y., "Using Smidump to Convert MIB to XSD", May 2007.

Appendix A. Open Issues

- do we need max-repetitions? It was not kept in the RFC 4088 URI format. Max-repetitions could be useful when retrieving a table that contains numerous entries. The whole table can be retrieved by multiple <mib-get> requests. Otherwise, the size of response message will be huge, and it will take a long time to transport the message.
- Can I xpath filter on arbitrary content? I assume this is the case but that should probably be spelled out (e.g. retrieve all interfaces where the name starts with a given prefix - something operators find useful). If this is the case, what is the semantics of max-repetitions? an interface is identified by an ifIndex in MIBs, not by an interface's name. so it is hard to

filter interfaces by name or a given prefix. max-repetitions is an instruction, not an attribute of data. It is really a problem when it is used in xpath filter.

- To complete this work, someone must write a specification how smidump actually converts MIB modules to XML schemas so that there is agreement how to exchange MIB data.
-

Appendix B. Previous Work

Research was done in the 1990s into integrating SNMP agents with XML-based management systems. The paper [refs.libsmi] discusses some of the research that was done.

The "mibdump" tool is an implementation of the SMI to XML conversion, and something that could be connected to a netconf front-end.

There was a fancy Web wrapper which used xpath to actually select MIB data.

```
$ lynx -dump 'http://www.ibr.cs.tu-bs.de/snmp-xml-gw?\
    get=/snmp-data/context[@hostname="talisker.ibr.cs.tu-bs.de"]\
    /ifEntry[ifOperStatus="up" and (ifOutOctets > 0 or ifInOctets >
0)]/ifDescr'
```

See the minutes from the 12th NMRG meeting in Colorado Springs where Frank and Torsten discussed the SMI to XML mapping in some more details.

Authors' Addresses

Yan Li
Huawei Technologies
No.3 Xinxi Road, Shangdi Information Industry Base
Beijing, HaiDian District 100085
P.R.China

Phone: +86 10 8288 2008
EMail: liyan_77@huawei.com

David Harrington
Huawei Technologies (USA)
1700 Alma Dr. Suite 100
Plano, TX 75075
USA

Phone: +1 603 436 8634
EMail: dharrington@huawei.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).