Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2012

M. Bjorklund
Tail-f Systems
J. Schoenwaelder
Jacobs University

July 8, 2011

A YANG Data Model for SNMP Configuration
draft-bjorklund-netmod-snmp-cfg-01

Abstract

This document defines a collection of YANG definitions for configuring SNMP engines.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 9, 2012.

Copyright Notice

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] data model for the configuration of SNMP engines. The configuration model is consistent with the MIB modules defined in [RFC3411], [RFC3412], [RFC3413], [RFC3414], [RFC3415], [RFC3418], and [RFC3584] but takes advantage of YANG's ability to define hierarchical configuration data models. The structure of the model has been derived from existing proprietary configuration models implemented as command line interfaces.

The keywords "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. Overview

In order to preserve the modularity of SNMP, the YANG configuration data model is organized in a set of YANG submodules, all sharing the same module namespace. This allows to add configuration support for additional SNMP features while keeping the number of namespaces that have to be dealt with down to a minimum.

3. Data Model

3.1. General Considerations

Most YANG nodes are mapped 1-1 to the corresponding MIB object. The "reference" statement is used to indicate which corresponding MIB object the YANG node is mapped to. When there is not a simple 1-1 mapping, the "description" statement explains the mapping.

3.2. Common Definitions

The submodule "ietf-snmp-common" defines a set of common typedefs, features, and the top-level container "snmp". All configuration parameters defined in the other submodules are organized under this top-level container.

This submodule defines two YANG fetaures:

proxy: A server implements this feature if it can act as an SNMP Proxy.

notification-filter: A server implements this feature if it supports SNMP notification filtering.

3.3. Engine Configuration

The submodule "ietf-snmp-engine", which defines configuration parameters that are specific to SNMP engines, has the following structure:

```
+--rw snmp
   +--rw engine
      +--rw enabled?     boolean
      +--rw listen
      |  +--rw udp [ip port]
      |     +--rw ip      inet:ip-address
      |     +--rw port    inet:port-number
      +--rw version
      |  +--rw v1?    empty
      |  +--rw v2c?   empty
      |  +--rw v3?    empty
      +--rw engine-id?   snmp:engine-id
```

The leaf "/snmp/engine/enabled" can be used to enable/disable an SNMP engine. The container "/snmp/engine/listen" provides configuration of the transport endpoints the engine is listening to. This container is expected to be augmented for other transports. The "/snmp/engine/version" container can be used to enable/disable the

different message processing models.

3.4. Target Configuration

The submodule "ietf-snmp-target", which defines configuration parameters that correspond to the objects in SNMP-TARGET-MIB, has the following structure:

```
   +--rw snmp
      +--rw target [name]
         +--rw name       snmp:identifier
         +--rw (transport)
         |  +--:(udp)
         |     +--rw udp
         |        +--rw ip              inet:ip-address
         |        +--rw port?           inet:port-number
         |        +--rw prefix-length?  uint8
         +--rw tag*       snmp:identifier
         +--rw timeout?   uint32
         +--rw retries?   uint8
         +--rw (params)?
```

An entry in the list "/snmp/target" corresponds to an "snmpTargetAddrEntry".

The "snmpTargetAddrTDomain" and "snmpTargetAddrTAddress" objects are mapped to transport-specific YANG nodes. Each transport is configured as a separate case in the choice "transport". Currently, SNMP over UDP is supported.

In order to provide a simpler configuration model with less cross-references, the "target" list also inlines the "snmpTargetParamsEntry" pointed to by "snmpTargetAddrParams". This is accomplished with a choice "params", which is augmented by security model specific submodules, currently "ietf-snmp-community" (Section 3.7) and "ietf-snmp-usm" (Section 3.8).

The YANG model does not define a separate list that maps directly to "snmpTargetParamsTable". Since "snmpProxyTable" also has a reference to this table, "snmpProxyTable" also has a choice "params" which is augmented by security model specific submodules (Section 3.6).

3.5. Notification Configuration

The submodule "ietf-snmp-notification", which defines configuration parameters that correspond to the objects in SNMP-NOTIFICATION-MIB, has the following structure:

```
   +--rw snmp
      +--rw notify [name]
      |  +--rw name    snmp:identifier
      |  +--rw tag     leafref
      |  +--rw type?   enumeration
      +--rw notify-filter-profile [name]
      |  +--rw name       snmp:identifier
      |  +--rw include*   wildcard-object-identifier
      |  +--rw exclude*   wildcard-object-identifier
      +--rw enable-authen-traps?     boolean
```

It also augments the "target" list defined in the "ietf-snmp-target" submodule (Section 3.4) with one leaf:

```
  +--rw snmp
      +--rw target [name]
         ...
         +--rw notify-filter-profile?   leafref
```

An entry in the list "/snmp/notify" corresponds to an "snmpNotifyEntry".

An entry in the list "/snmp/notify-filter-profile" corresponds to an "snmpNotifyFilterProfileEntry". In the MIB, there is a sparse relationship between "snmpTargetParamsTable" and "snmpNotifyFilterProfileTable". In the YANG model, this sparse relationship is represented with a leafref leaf "notify-filter-profile" in the "/snmp/target" list, which refers to an entry in the "/snmp/notify-filter-profile" list.

The "snmpNotifyFilterTable" is represented as a list "filter" within the "/snmp/notify-filter-profile" list.

3.6. Proxy Configuration

The submodule "ietf-snmp-proxy", which defines configuration parameters that correspond to the objects in SNMP-PROXY-MIB, has the following structure:

```
   +--rw snmp
      +--rw proxy [name]
         +--rw name                    snmp:identifier
         +--rw type                    enumeration
         +--rw context-engine-id       snmp:engine-id
         +--rw context-name?           snmp:context-name
         +--rw params-in
         |  +--rw (params)
         +--rw single-target-out?      leafref
```

```
            +--rw multiple-target-out?    leafref
```

An entry in the list "/snmp/proxy" corresponds to an "snmpProxyEntry".

Like the "target" list (Section 3.4), the "proxy" list inlines the "snmpTargetParamsEntry" pointed to by "snmpProxyTargetParamsIn". This is accomplished with a choice "params", which is augmented by security model specific submodules, currently "ietf-snmp-community" (Section 3.7) and "ietf-snmp-usm" (Section 3.8).

3.7. Community Configuration

The submodule "ietf-snmp-community", which defines configuration parameters that correspond to the objects in SNMP-COMMUNITY-MIB, has the following structure:

```
   +--rw snmp
      +--rw community [index]
         +--rw index            snmp:identifier
         +--rw (name)?
         |  +--:(text-name)
         |  |  +--rw text-name?       string
         |  +--:(binary-name)
         |     +--rw binary-name?     binary
         +--rw security-name    snmp:security-name
         +--rw engine-id?       snmp:engine-id
         +--rw context?         snmp:context-name
         +--rw target-tag?      leafref
```

It also augments the "/snmp/target/params" and "/snmp/proxy/params-in/params" choices with nodes for the Community-Based Security Model used by SNMPv1 and SNMPv2c:

```
      +--rw snmp
         +--rw target [name]
         |  ...
         |  +--rw (params)?
         |  |  +--:(v1)
         |  |  |  +--rw v1
         |  |  |     +--rw community    leafref
         |  |  +--:(v2c)
         |  |     +--rw v2c
         |  |        +--rw community    leafref
         |  +--rw mms?       union
         +--rw proxy
            +--rw params-in
               +--rw params
                  +--:(v1)
                  |  +--rw v1
                  |     +--rw community    leafref
                  +--:(v2c)
                     +--rw v2c
                        +--rw community    leafref
```

   An entry in the list "/snmp/community" corresponds to an "snmpCommunityEntry".

   When a case "v1" or "v2c" is chosen, it implies a snmpTargetParamsMPModel 0 (SNMPv1) or 1 (SNMPv2), and a snmpTargetParamsSecurityModel 1 (SNMPv1) or 2 (SNMPv2), respectively. Both cases implies a snmpTargetParamsSecurityLevel noAuthNoPriv.

3.8. User-based Security Model Configuration

   The submodule "ietf-snmp-usm", which defines configuration parameters that correspond to the objects in SNMP-USER-BASED-SM-MIB, has the following structure:

```
      +--rw snmp
         +--rw usm
            +--rw local
            |  +--rw user [name]
            |     +-- {common user params}
            +--rw remote [engine-id]
               +--rw engine-id    snmp:engine-id
               +--rw user [name]
                  +-- {common user params}
```

   The "{common user params}" are:

```
      +--rw name    snmp:identifier
      +--rw auth?
      |  +--rw (protocol)
      |     +--:(md5)
      |     |  +--rw md5
      |     |     +-- {common key params}
      |     +--:(sha)
      |        +--rw sha
      |           +-- {common key params}
      +--rw priv?
         +--rw (protocol)
            +--:(des)
            |  +--rw des
            |     +-- {common key params}
            +--:(aes)
               +--rw aes
                  +-- {common key params}
```

The "{common key params}" are:

```
      +--rw (key-type)?
         +--:(password)
         |  +--rw password?   string
         +--:(key)
            +--rw key?        string
```

It also augments the "/snmp/target/params" and "/snmp/proxy/params-in/params" choices with nodes for the SNMP User-based Security Model.

```
     +--rw snmp
         +--rw target [name]
            ...
         |  +--rw (params)?
         |     +--:(usm)
         |        +--rw usm
         |           +--rw user-name         snmp:security-name
         |           +--rw security-level    security-level
         +--rw proxy [name]
            ...
            +--rw params-in
               +--rw (params)
                  +--:(usm)
                     +--rw usm
                        +--rw user-name         snmp:security-name
                        +--rw security-level    security-level
```

In the MIB, there is a single table with local and remote users,

indexed by the engine id and user name. In the YANG model, there is one list of local users, and a nested list of remote users.

In the MIB, there are several objects related to changing the authentication and privacy keys. These objects are not present in the YANG model. Instead, there is a choice between a password or a localized key. If a password is given, it is used by the server to calculate a localized key, which is stored in the configuration. The clear-text password is never stored. This implies that if the engine id is changed, all users keys need to be changed as well.

3.9. View-based Access Control Model Configuration

The submodule "ietf-snmp-vacm", which defines configuration parameters that correspond to the objects in SNMP-VIEW-BASED-ACM-MIB, has the following structure:

```
   +--rw snmp
      +--rw vacm
         +--rw group [name]
         |  +--rw name      group-name
         |  +--rw member [security-name]
         |  |  +--rw security-name     snmp:security-name
         |  |  +--rw security-model*   snmp:security-model
         |  +--rw access [context security-model security-level]
         |     +--rw context           snmp:context-name
         |     +--rw context-match?    enumeration
         |     +--rw security-model    snmp:security-model-or-any
         |     +--rw security-level    snmp:security-level
         |     +--rw read-view?        leafref
         |     +--rw write-view?       leafref
         |     +--rw notify-view?      leafref
         +--rw view [name]
            +--rw name       view-name
            +--rw include*   snmp:wildcard-object-identifier
            +--rw exclude*   snmp:wildcard-object-identifier
```

The "vacmSecurityToGroupTable" and "vacmAccessTable" are mapped to a structure of nested lists in the YANG model. Groups are defined in the list "/snmp/vacm/group", and for each group, there is a sublist "member", which maps to "vacmSecurityToGroupTable", and a sublist "access", which maps to "vacmAccessTable".

MIB views are defined in the list "/snmp/vacm/view", and for each MIB view, there is a leaf-list of included subtree families and a leaf-list of excluded subtree families. This is more compact and thus more readable representation of the "vacmViewTreeFamilyTable".

4. Definitions

4.1. Module 'ietf-snmp'

<CODE BEGINS> file "ietf-snmp.yang"

```
 module ietf-snmp {

   namespace "urn:ietf:params:xml:ns:yang:ietf-snmp";
   prefix "snmp";

   include ietf-snmp-common {
     revision-date 2011-07-08;
   }
   include ietf-snmp-engine {
     revision-date 2011-07-08;
   }
   include ietf-snmp-target {
     revision-date 2011-07-08;
   }
   include ietf-snmp-notification {
     revision-date 2011-07-08;
   }
   include ietf-snmp-proxy {
     revision-date 2011-07-08;
   }
   include ietf-snmp-community {
     revision-date 2011-07-08;
   }
   include ietf-snmp-usm {
     revision-date 2011-07-08;
   }
   include ietf-snmp-vacm {
     revision-date 2011-07-08;
   }

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>
```

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This module contains a collection of YANG definitions for
      configuring SNMP engines.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }

 }
```

<CODE ENDS>

4.2. Submodule 'ietf-snmp-common'

<CODE BEGINS> file "ietf-snmp-common.yang"

```
 submodule ietf-snmp-common {

   belongs-to ietf-snmp {
     prefix snmp;
   }
```

```
   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of common YANG definitions
      for configuring SNMP engines.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }
```

```
   /* Collection of SNMP features */

   feature proxy {
     description
       "A server implements this feature if it can act as an
       SNMP Proxy";
   }

   feature notification-filter {
     description
       "A server implements this feature if it supports SNMP
       notification filtering.";
   }

   // FIXME: should we have v1 and v2c as features?

   /* Collection of SNMP specific data types */

   typedef admin-string {
     type string {
       length "0..255";
     }
     description
       "Represents and SnmpAdminString as defined in RFC 3411.

       Note that the size of an SnmpAdminString is measured in
       octets, not characters.";
     reference "SNMP-FRAMEWORK-MIB.SnmpAdminString";
   }

   typedef identifier {
     type admin-string {
       length "1..32";
     }
     description
       "Identifiers are used to name items in the SNMP configuration
       data store.";
   }

   typedef context-name {
     type admin-string {
       length "0..32";
     }
     description
       "The context type represents an SNMP context name.";
   }

   typedef security-name {
```

```
    type admin-string {
      length "1..32";
    }
    description
      "The security-name type represents an SNMP security name.";
    reference
      "";
  }

  typedef security-model {
    type union {
      type enumeration {
        enum v1  { value 1; }
        enum v2c { value 2; }
        enum usm { value 3; }
        //enum tsm { value 4; }
      }
      // FIXME: remove this?  isn't it better to revise this
      // enum as need arises?
      type int32 {
        range "1..2147483647";
      }
    }
    reference
      "RFC3411: An Architecture for Describing SNMP Management
                Frameworks";
  }

  typedef security-model-or-any {
    type union {
      type enumeration {
        enum any { value 0; }
      }
      type security-model;
    }
    reference
      "RFC3411: An Architecture for Describing SNMP Management
                Frameworks";
  }

  typedef security-level {
    type enumeration {
      enum no-auth-no-priv { value 1; }
      enum auth-no-priv    { value 2; }
      enum auth-priv       { value 3; }
    }
    reference
      "RFC3411: An Architecture for Describing SNMP Management
```

```
                Frameworks";
    }

    typedef engine-id {
      type string {
        pattern '([0-9a-fA-F]){2}(:([0-9a-fA-F]){2}){4,31}';
      }
      description
        "The Engine ID specified as a list of colon-specified hexa-
        decimal octets e.g. '4F:4C:41:71'.";
      reference
        "RFC3411: An Architecture for Describing SNMP Management
                  Frameworks";
    }

    typedef wildcard-object-identifier {
      type string;
      description
        "The wildcard-object-identifier type represents an SNMP object
        identifier where subidentifiers can be given either as a label,
        in numeric form, or a wildcard, represented by a *.";
    }

    container snmp {
      description
        "Top-level container for SNMP related configuration and
        status objects.";
    }

  }

    <CODE ENDS>
```

4.3. Submodule 'ietf-snmp-engine'

```
    <CODE BEGINS> file "ietf-snmp-engine.yang"

  submodule ietf-snmp-engine {

    belongs-to ietf-snmp {
      prefix snmp;
    }

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

```
   }

   include ietf-snmp-common;

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions
      for configuring SNMP engines.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
```

```
    reference
      "RFC XXXX: A YANG Data Model for SNMP Configuration";
  }

  augment /snmp:snmp {

    container engine {

      description
        "Configuration of the SNMP engine.";

      leaf enabled {
        type boolean;
        default "false";
        description
          "Enables the SNMP engine.";
      }

      container listen {
        description
          "Configuration of the transport endpoints on which the
           engine listens. Submodules providing configuration for
           additional transports are expected to augment this
           container.";

        list udp {
          key "ip port";
          description
            "A list of IPv4 and IPv6 addresses and ports to which the
             engine listens.";

          leaf ip {
            type inet:ip-address;
            description
              "The IPv4 or IPv6 address on which the engine listens.";
          }
          leaf port {
            type inet:port-number;
            description
              "The UDP port on which the engine listens.";
          }
        }
      }

      container version {
        description
          "SNMP version used by the engine";
        leaf v1 {
```

```
          type empty;
        }
        leaf v2c {
          type empty;
        }
        leaf v3 {
          type empty;
          must "../../engine-id" {
            error-message
              "when v3 is configured, an engine-id must be set";
          }
        }
      }

      leaf engine-id {
        type snmp:engine-id;
        description
          "The local SNMP engine's administratively-assigned unique
           identifier.

           If this leaf is not set, the device automatically
           calculates an engine id, as described in RFC 3411.  A
           server MAY initialize this leaf with the automatically
           created value.";
        reference "SNMP-FRAMEWORK-MIB.snmpEngineID";
      }
    }
  }
}
```

<CODE ENDS>

4.4. Submodule 'ietf-snmp-target'

<CODE BEGINS> file "ietf-snmp-target.yang"

```
submodule ietf-snmp-target {

  belongs-to ietf-snmp {
    prefix snmp;
  }

  import ietf-inet-types {
    prefix inet;
  }

  include ietf-snmp-common;
```

```
   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions
      for configuring SNMP targets.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   reference
     "RFC3413: Simple Network Management Protocol (SNMP) Applications";

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
```

```
      "RFC XXXX: A YANG Data Model for SNMP Configuration";
  }

  augment /snmp:snmp {

    list target {
      key name;
      description
        "List of targets.";
      reference "SNMP-TARGET-MIB.snmpTargetAddrTable";

      leaf name {
        type snmp:identifier;
        description
          "Identifies the target.";
        reference "SNMP-TARGET-MIB.snmpTargetAddrName";
      }
      choice transport {
        mandatory true;
        description
          "Transport address of the target.

           The snmpTargetAddrTDomain and snmpTargetAddrTAddress
           objects are mapped to transport-specific YANG nodes.  Each
           transport is configured as a separate case in this
           choice. Submodules providing configuration for additional
           transports are expected to augment this choice.";
        reference "SNMP-TARGET-MIB.snmpTargetAddrTDomain
                   SNMP-TARGET-MIB.snmpTargetAddrTAddress";
        case udp {
          reference "SNMPv2-TM.snmpUDPDomain
                     TRANSPORT-ADDRESS-MIB.transportDomainUdpIpv4
                     TRANSPORT-ADDRESS-MIB.transportDomainUdpIpv4z
                     TRANSPORT-ADDRESS-MIB.transportDomainUdpIpv6
                     TRANSPORT-ADDRESS-MIB.transportDomainUdpIpv6z";
          container udp {
            leaf ip {
              type inet:ip-address;
              mandatory true;
              reference "SNMP-TARGET-MIB.snmpTargetAddrTAddress";
            }
            leaf port {
              type inet:port-number;
              default 162;
              description
                "UDP port number";
              reference "SNMP-TARGET-MIB.snmpTargetAddrTAddress";
            }
```

```
            leaf prefix-length {
              type uint8;
              description
                "The value of this leaf must match the value of
                 ../snmp:ip. If ../snmp:ip contains an ipv4 address,
                 this leaf must be less than or equal to 32.  If it
                 contains an ipv6 address, it must be less than or
                 equal to 128.

                 Note that the prefix-length is currently only used by
                 the Community-based Security Model to filter incoming
                 messages. Furthermore, the prefix-length filtering
                 is not covering all possible filters supported by the
                 corresponding MIB object.";
              reference "SNMP-COMMUNITY-MIB.snmpTargetAddrTMask";
            }
          }
        }
      }
      leaf-list tag {
        type snmp:identifier;
        description
          "List of tag values used to select target address.";
        reference "SNMP-TARGET-MIB.snmpTargetAddrTagList";
      }
      leaf timeout {
        type uint32;
        units "0.01 seconds";
        default 1500;
        description
          "Needed only if this target can receive InformRequest-PDUs.";
        reference "SNMP-TARGET-MIB.snmpTargetAddrTimeout";
      }
      leaf retries {
        type uint8;
        default 3;
        description
          "Needed only if this target can receive InformRequest-PDUs.";
        reference "SNMP-TARGET-MIB.snmpTargetAddrRetryCount";
      }
      choice params {

        // FIMXE: mandatory or not?

        description
          "This choice is augmented with case nodes containing
           security model specific configuration parameters.  Each
           such case represents one entry in the
```

```
            snmpTargetParamsTable.

            When the snmpTargetAddrParams object contains a reference
            to a non-existing snmpTargetParamsEntry, this choice does
            not contain any case, and vice versa.";
          reference "SNMP-TARGET-MIB.snmpTargetAddrParams
                     SNMP-TARGET-MIB.snmpTargetParamsTable";
        }
      }
    }
  }
```

<CODE ENDS>

4.5. Submodule 'ietf-snmp-notification'

<CODE BEGINS> file "ietf-snmp-notification.yang"

```
 submodule ietf-snmp-notification {

   belongs-to ietf-snmp {
     prefix snmp;
   }

   include ietf-snmp-common;
   include ietf-snmp-target;

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions
```

```
      for configuring SNMP notifications.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   reference
     "RFC3413: Simple Network Management Protocol (SNMP) Applications";

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }

   augment /snmp:snmp {

     list notify {
       key name;
       description
         "Targets that will receive notifications.

          Entries in this lists are mapped 1-1 to entries in
          snmpNotifyTable, except that if an entry in snmpNotifyTable
          has a snmpNotifyTag for which no snmpTargetAddrEntry exists,
          then the snmpNotifyTable entry is not mapped to an entry in
          this list.";
       reference "SNMP-NOTIFICATION-MIB.snmpNotifyTable";

       leaf name {
         type snmp:identifier;
         description
           "An arbitrary name for the list entry.";
```

```
        reference "SNMP-NOTIFICATION-MIB.snmpNotifyName";
      }
      leaf tag {
        type leafref {
          path "/snmp/target/tag";
        }
        mandatory true;
        description
          "Target tag, selects a set of notification targets.";
        reference "SNMP-NOTIFICATION-MIB.snmpNotifyTag";
      }
      leaf type {
        type enumeration {
          enum trap { value 1; }
          enum inform { value 2; }
        }
        default trap;
        description
          "Defines the notification type to be generated.";
        reference "SNMP-NOTIFICATION-MIB.snmpNotifyType";
      }
    }

    list notify-filter-profile {
      if-feature snmp:notification-filter;
      key name;

      description
        "Notification filter profiles.

         The leaf /snmp/target/notify-filter-profile is used
         to associate a filter profile with a target.

         If an entry in this list is referred to by one or more
         /snmp/target/notify-filter-profile, each such
         notify-filter-profile is represented by one
         snmpNotifyFilterProfileEntry.

         If an entry in this list is not referred to by any
         /snmp/target/notify-filter-profile, the entry is not mapped
         to snmpNotifyFilterProfileTable.";
      reference "SNMP-NOTIFICATION-MIB.snmpNotifyFilterProfileTable
                 SNMP-NOTIFICATION-MIB.snmpNotifyFilterTable";

      leaf name {
        type snmp:identifier;
        description
          "Name of the filter profile";
```

```
        reference "SNMP-NOTIFICATION-MIB.snmpNotifyFilterProfileName";
      }

      leaf-list include {
        type wildcard-object-identifier;
        description
          "A family of subtrees included in this filter.";
        reference "SNMP-NOTIFICATION-MIB.snmpNotifyFilterSubtree
                   SNMP-NOTIFICATION-MIB.snmpNotifyFilterMask
                   SNMP-NOTIFICATION-MIB.snmpNotifyFilterType";
      }

      leaf-list exclude {
        type wildcard-object-identifier;
        description
          "A family of subtrees excluded from this filter.";
        reference "SNMP-NOTIFICATION-MIB.snmpNotifyFilterSubtree
                   SNMP-NOTIFICATION-MIB.snmpNotifyFilterMask
                   SNMP-NOTIFICATION-MIB.snmpNotifyFilterType";
      }
    }

    leaf enable-authen-traps {
      type boolean;
      description
        "Indicates whether the SNMP entity is permitted to
         generate authenticationFailure traps.";
      reference "SNMPv2-MIB.snmpEnableAuthenTraps";
    }
  }

  augment /snmp:snmp/snmp:target {
    reference "SNMP-NOTIFICATION-MIB.snmpNotifyFilterProfileTable";
    leaf notify-filter-profile {
      if-feature snmp:notification-filter;
      type leafref {
        path "/snmp/notify-filter-profile/name";
      }
      description
        "This leafref leaf is used to represent the sparse relationship
         between the /snmp/target list and the
         /snmp/notify-filter-profile list.";

      reference "SNMP-NOTIFICATION-MIB.snmpNotifyFilterProfileName";
    }
  }

}
```

```
   <CODE ENDS>
```

4.6. Submodule 'ietf-snmp-proxy'

```
   <CODE BEGINS> file "ietf-snmp-proxy.yang"

 submodule ietf-snmp-proxy {

   belongs-to ietf-snmp {
     prefix snmp;
   }

   include ietf-snmp-common;
   include ietf-snmp-target;

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions
      for configuring SNMP proxies.

      Copyright (c) 2011 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 XXXX; see
```

```
      the RFC itself for full legal notices.";

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

   reference
     "RFC3413: Simple Network Management Protocol (SNMP) Applications";

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }

   augment /snmp:snmp {
     if-feature snmp:proxy;

     list proxy {
       key name;

       description
         "List of proxy parameters.";
       reference "SNMP-PROXY-MIB.snmpProxyTable";

       leaf name {
         type snmp:identifier;
         description
           "Identifies the proxy parameter entry.";
         reference "SNMP-PROXY-MIB.snmpProxyName";
       }
       leaf type {
         type enumeration {
           enum read;
           enum write;
           enum trap;
           enum inform;
         }
         mandatory true;
         reference "SNMP-PROXY-MIB.snmpProxyType";
       }
       leaf context-engine-id {
         type snmp:engine-id;
         mandatory true;
         reference "SNMP-PROXY-MIB.snmpProxyContextEngineID";
```

```
        }
        leaf context-name {
          type snmp:context-name;
          reference "SNMP-PROXY-MIB.snmpProxyContextName";
        }
        container params-in {
          choice params {
            mandatory true;
            description
              "This choice is augmented with case nodes containing
               security model specific configuration parameters.  Each
               such case represents one entry in the
               snmpTargetParamsTable.

               When the snmpProxyTargetParamsIn object contains a
               reference to a non-existing snmpTargetParamsEntry, this
               choice does not contain any case, and vice versa.";
          }
          reference "SNMP-PROXY-MIB.snmpProxyTargetParamsIn";
        }
        leaf single-target-out {
          when "../type = read or ../type = write";
          type leafref {
            path "/snmp:snmp/snmp:target/snmp:name";
          }
          description
            "When the snmpProxySingleTargetOut object contains
             a value which does not select an snmpTargetAddrEntry,
             this leaf does not exist.";
          reference "SNMP-PROXY-MIB.snmpProxySingleTargetOut";
        }
        leaf multiple-target-out {
          when "../type = trap or ../type = inform";
          type leafref {
            path "/snmp:snmp/snmp:target/snmp:tag";
          }
          description
            "When the snmpProxyMultipleTargetOut object contains
             a value which does not select any snmpTargetAddrEntries,
             this leaf does not exist.";
          reference "SNMP-PROXY-MIB.snmpProxyMultipleTargetOut";
        }
      }
    }
  }
```

```
   <CODE ENDS>
```

4.7. Submodule 'ietf-snmp-community'

```
   <CODE BEGINS> file "ietf-snmp-community.yang"

 submodule ietf-snmp-community {

   belongs-to ietf-snmp {
     prefix snmp;
   }

   include ietf-snmp-common;
   include ietf-snmp-target;
   include ietf-snmp-proxy;

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions
      for configuring community-based SNMP.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   reference
     "RFC3584: Coexistence between Version 1, Version 2, and Version 3
               of the Internet-standard Network Management Framework";

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }

   augment /snmp:snmp {

     list community {
       key index;

       description
         "List of communities";
       reference "SNMP-COMMUNITY-MIB.snmpCommunityTable";

       leaf index {
         type snmp:identifier;
         description
           "Index into the community list.";
         reference "SNMP-COMMUNITY-MIB.snmpCommunityIndex";
       }
       choice name {
         description
           "The community name, either specified as a string
            or as a binary.  The binary name is used when the
            community name contains characters that are not legal
            in a string.

            If not set, the value of 'security-name' is operationally
            used as the snmpCommunityName.";
         reference "SNMP-COMMUNITY-MIB.snmpCommunityName";
         leaf text-name {
           type string;
           description
```

```
            "A community name that can be represented as a
             YANG string.";
        }
        leaf binary-name {
          type binary;
          description
            "A community name represented as a binary value.";
        }
      }
      leaf security-name {
        type snmp:security-name;
        mandatory true;
        description
          "The snmpCommunitySecurityName of this entry.";
        reference "SNMP-COMMUNITY-MIB.snmpCommunitySecurityName";
      }
      leaf engine-id {
        if-feature snmp:proxy;
        type snmp:engine-id;
        description
          "If not set, the value of the local SNMP engine is
           operationally used by the device.";
        reference "SNMP-COMMUNITY-MIB.snmpCommunityContextEngineID";
      }
      leaf context {
        type snmp:context-name;
        default "";
        description
          "The context in which management information is accessed
           when using the community string specified by this entry.";
        reference "SNMP-COMMUNITY-MIB.snmpCommunityContextName";
      }
      leaf target-tag {
        type leafref {
          path "/snmp/target/tag";
        }
        description
          "Used to limit access for this community to the specified
           targets.";
        reference "SNMP-COMMUNITY-MIB.snmpCommunityTransportTag";
      }
    }
  }

  grouping v1-target-params {
    container v1 {
      description
        "SNMPv1 parameters type.
```

```
          Represents snmpTargetParamsMPModel '0',
          snmpTargetParamsSecurityModel '1', and
          snmpTargetParamsSecurityLevel 'noAuthNoPriv'.";
        leaf community {
          type leafref {
            path "/snmp/community/security-name";
          }
          mandatory true;
          reference "SNMP-TARGET-MIB.snmpTargetParamsSecurityName";
        }
      }
    }

    grouping v2c-target-params {
      container v2c {
        description
          "SNMPv2 community parameters type.
           Represents snmpTargetParamsMPModel '1',
           snmpTargetParamsSecurityModel '2', and
           snmpTargetParamsSecurityLevel 'noAuthNoPriv'.";
        leaf community {
          type leafref {
            path "/snmp/community/security-name";
          }
          mandatory true;
          reference "SNMP-TARGET-MIB.snmpTargetParamsSecurityName";
        }
      }
    }

    augment /snmp:snmp/snmp:target/snmp:params {
      case v1 {
        uses v1-target-params;
      }
      case v2c {
        uses v2c-target-params;
      }
    }

    augment /snmp:snmp/snmp:proxy/snmp:params-in/snmp:params {
      case v1 {
        uses v1-target-params;
      }
      case v2c {
        uses v2c-target-params;
      }
    }
```

```
   augment /snmp:snmp/snmp:target {
     leaf mms {
       when "snmp:params/snmp:v1 or snmp:params/snmp:v2c";
       type union {
         type enumeration {
           enum "unknown";
         }
         type int32 {
           range "484..max";
         }
       }
       default "484";
       reference
         "SNMP-COMMUNITY-MIB.snmpTargetAddrMMS";
     }
   }

 }


   <CODE ENDS>
```

4.8. Submodule 'ietf-snmp-usm'

```
   <CODE BEGINS> file "ietf-snmp-usm.yang"

 submodule ietf-snmp-usm {

   belongs-to ietf-snmp {
     prefix snmp;
   }

   include ietf-snmp-common;
   include ietf-snmp-target;
   include ietf-snmp-proxy;

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>
```

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions for
      configuring the User-based Security Model (USM) of SNMP.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   reference
     "RFC3414: User-based Security Model (USM) for version 3 of the
               Simple Network Management Protocol (SNMPv3).";

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }

   grouping key {
     choice key-type {
       leaf password {
         type string;
         description
           "If this leaf is set, the server uses its value to create
            a localized key, according to the algorithm described in
            RFC 3414.  The resulting localized key is stored in the
            configuration, in the 'key' leaf.  The clear-text password
```

```
            is never stored, and thus never returned in a read
            operation.

            Note that if the engine id is changed, the passwords for
            the engine's users need to be set again, in order to
            re-calculate the localized keys.";
        }
        leaf key {
          type string {
            pattern '([0-9a-fA-F]){2}(:([0-9a-fA-F]){2})*';
          }
          description
            "Localized key specified as a list of colon-specified
             hexa-decimal octets";
        }
      }
    }

    grouping user-list {
      list user {
        key "name";

        reference "SNMP-USER-BASED-SM-MIB.usmUserTable";

        leaf name {
          type snmp:identifier {
            length "1..32";
          }
          reference "SNMP-USER-BASED-SM-MIB.usmUserName";
        }
        container auth {
          presence "enables authentication";
          description
            "Enables authentication of the user";
          choice protocol {
            mandatory true;
            reference "SNMP-USER-BASED-SM-MIB.usmUserAuthProtocol";
            container md5 {
              uses key;
              reference "SNMP-USER-BASED-SM-MIB.usmHMACMD5AuthProtocol";
            }
            container sha {
              uses key;
              reference "SNMP-USER-BASED-SM-MIB.usmHMACSHAAuthProtocol";
            }
          }
        }
        container priv {
```

```
        must "../auth" {
          error-message
            "when privacy is used, authentication must also be used";
        }
        presence "enables encryption";
        description
          "Enables encryption of SNMP messages.";

        choice protocol {
          mandatory true;
          reference "SNMP-USER-BASED-SM-MIB.usmUserPrivProtocol";
          container des {
            uses key;
            reference "SNMP-USER-BASED-SM-MIB.usmDESPrivProtocol";
          }
          container aes {
            uses key;
            reference "SNMP-USM-AES-MIB.usmAesCfb128Protocol";
          }
        }
      }
    }
  }

  augment /snmp:snmp {

    container usm {
      description
        "Configuration of the User-based Security Model";
      container local {
        uses user-list;
      }

      list remote {
        key "engine-id";

        leaf engine-id {
          type snmp:engine-id;
          reference "SNMP-USER-BASED-SM-MIB.usmUserEngineID";
        }

        uses user-list;
      }
    }
  }

  grouping usm-target-params {
    container usm {
```

```
      description
        "User based SNMPv3 parameters type.
         Represents snmpTargetParamsMPModel '3'.";
      leaf user-name {
        type snmp:security-name;
        mandatory true;
        reference
          "SNMP-TARGET-MIB.snmpTargetParamsSecurityName";
      }
      leaf security-level {
        type security-level;
        mandatory true;
        reference
          "SNMP-TARGET-MIB.snmpTargetParamsSecurityLevel";
      }
    }
  }

  augment /snmp:snmp/snmp:target/snmp:params {
    case usm {
      uses usm-target-params;
    }
  }

  augment /snmp:snmp/snmp:proxy/snmp:params-in/snmp:params {
    case usm {
      uses usm-target-params;
    }
  }

  augment /snmp:snmp/snmp:target {
    // FIXME: choice between leaf discovery { type empty; }
    // and engine-id?
    leaf engine-id {
      type leafref {
        path "/snmp/usm/remote/engine-id";
      }
      must '../usm/user-name' {
        error-message
          "When engine-id is set, usm/user-name must also be set.";
      }
      must '/snmp/usm/remote[engine-id=current()]/'
        + 'user[name=current()/../usm/user-name]' {
        error-message
          "When engine-id is set, the usm/user-name must exist in
           the /snmp/usm/remote list for this engine-id.";
      }
      description
```

```
          "Needed only if this target can receive InformRequest-PDUs
           over SNMPv3.

           This object is not present in the SNMP MIBs.  In
           RFC 3412, it is a implementation specific matter how this
           engine-id is handled.";
        reference "RFC 3412 7.1.9a";
      }
    }

 }
```

<CODE ENDS>

4.9. Submodule 'ietf-snmp-vacm'

<CODE BEGINS> file "ietf-snmp-vacm.yang"

```
 submodule ietf-snmp-vacm {

   belongs-to ietf-snmp {
     prefix snmp;
   }

   include ietf-snmp-common;

   organization
     "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

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

      WG Chair: David Kessens
                <mailto:david.kessens@nsn.com>

      WG Chair: Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>

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

      Editor:   Juergen Schoenwaelder
                <mailto:j.schoenwaelder@jacobs-university.de>";

   description
     "This submodule contains a collection of YANG definitions
      for configuring the View-based Access Control Model (VACM)
```

```
      of SNMP.

      Copyright (c) 2011 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 XXXX; see
      the RFC itself for full legal notices.";

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

   reference
     "RFC3415: View-based Access Control Model (VACM) for the
               Simple Network Management Protocol (SNMP)";

   // RFC Ed.: update the date below with the date of RFC publication
   // and remove this note.

   revision 2011-07-08 {
     description
       "Initial revision.";
     reference
       "RFC XXXX: A YANG Data Model for SNMP Configuration";
   }

   typedef view-name {
     type snmp:identifier;
     description
       "The view-name type represents an SNMP VACM view name.";
   }

   typedef group-name {
     type snmp:identifier;
     description
       "The group-name type represents an SNMP VACM group name.";
   }

   augment /snmp:snmp {

     container vacm {
       description
         "Configuration of the View-based Access Control Model";
```

```
      list group {
        key name;
        description
          "VACM Groups.

           This data model has a different structure than the MIB.
           Groups are explicitly defined in this list, and group
           members are defined in the 'member' list (mapped to
           vacmSecurityToGroupTable), and access for the group is
           defined in the 'access' list (mapped to
           vacmAccessTable).";
        reference "SNMP-VIEW-BASED-ACM-MIB.vacmSecurityToGroupTable
                   SNMP-VIEW-BASED-ACM-MIB.vacmAccessTable";

        leaf name {
          type group-name;
          description
            "The name of this VACM group.";
          reference "SNMP-VIEW-BASED-ACM-MIB.vacmGroupName";
        }

        list member {
          key "security-name";
          min-elements 1;
          description
            "A member of this VACM group. According to VACM, every
             group must have at least one member.

             A certain combination of security-name and security-model
             MUST NOT be present in more than one group.";
          reference "SNMP-VIEW-BASED-ACM-MIB.vacmSecurityToGroupTable";

          leaf security-name {
            type snmp:security-name;
            description
              "The securityName of a group member.";
            reference "SNMP-VIEW-BASED-ACM-MIB.vacmSecurityName";
          }

          leaf-list security-model {
            type snmp:security-model;
            min-elements 1;
            description
              "The security models under which this security-name
               is a member of this group.";
            reference "SNMP-VIEW-BASED-ACM-MIB.vacmSecurityModel";
          }
        }
```

```
        list access {
          key "context security-model security-level";
          description
            "Definition of access right for groups";
          reference "SNMP-VIEW-BASED-ACM-MIB.vacmAccessTable";

          leaf context {
            type snmp:context-name;
            description
              "The context (prefix) under which the access rights
               apply.";
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessContextPrefix";
          }

          leaf context-match {
            type enumeration {
              enum exact;
              enum prefix;
            }
            default exact;
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessContextMatch";
          }

          leaf security-model {
            type snmp:security-model-or-any;
            description
              "The security model under which the access rights
              apply.";
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessSecurityModel";
          }

          leaf security-level {
            type snmp:security-level;
            description
              "The minimum security level under which the access
               rights apply.";
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessSecurityLevel";
          }

          leaf read-view {
            type leafref {
              path "/snmp/vacm/view/name";
            }
            description
```

```
              "The name of the MIB view of the SNMP context
               authorizing read access. If this leaf does not
               exist in a configuration, it maps to a zero-length
               vacmAccessReadViewName.";
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessReadViewName";
          }

          leaf write-view {
            type leafref {
              path "/snmp/vacm/view/name";
            }
            description
              "The name of the MIB view of the SNMP context
               authorizing write access. If this leaf does not
               exist in a configuration, it maps to a zero-length
               vacmAccessWriteViewName.";
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessWriteViewName";
          }

          leaf notify-view {
            type leafref {
              path "/snmp/vacm/view/name";
            }
            description
              "The name of the MIB view of the SNMP context
               authorizing notify access. If this leaf does not
               exist in a configuration, it maps to a zero-length
               vacmAccessNotifyViewName.";
            reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmAccessNotifyViewName";
          }
        }
      }

      list view {
        key name;
        description
          "Definition of MIB views.";
        reference
          "SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilyTable";

        leaf name {
          type view-name;
          description
            "The name of this VACM MIB view.";
          reference
```

```
            "SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilyName";
        }

        leaf-list include {
          type snmp:wildcard-object-identifier;
          description
            "A family of subtrees included in this MIB view.";
          reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilySubtree
               SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilyMask
               SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilyType";
        }

        leaf-list exclude {
          type snmp:wildcard-object-identifier;
          description
            "A family of subtrees excluded from this MIB view.";
          reference
              "SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilySubtree
               SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilyMask
               SNMP-VIEW-BASED-ACM-MIB.vacmViewTreeFamilyType";
        }
      }
    }
  }
}
```

<CODE ENDS>

5. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in RFC 3688, the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-snmp

Registrant Contact: The NETMOD WG of the IETF.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

```
name:         ietf-snmp
namespace:    urn:ietf:params:xml:ns:yang:ietf-snmp
prefix:       snmp
reference:    RFC XXXX
```

The document registers the following YANG submodules in the YANG Module Names registry [RFC6020].

```
        name:         ietf-snmp-common
        parent:       ietf-snmp
        reference:    RFC XXXX

        name:         ietf-snmp-engine
        parent:       ietf-snmp
        reference:    RFC XXXX

        name:         ietf-snmp-community
        parent:       ietf-snmp
        reference:    RFC XXXX

        name:         ietf-snmp-notification
        parent:       ietf-snmp
        reference:    RFC XXXX

        name:         ietf-snmp-target
        parent:       ietf-snmp
        reference:    RFC XXXX

        name:         ietf-snmp-usm
        parent:       ietf-snmp
        reference:    RFC XXXX

        name:         ietf-snmp-vacm
        parent:       ietf-snmp
        reference:    RFC XXXX
```

6. Security Considerations

The YANG module and submodules defined in this memo are designed to be accessed via the NETCONF protocol [I-D.ietf-netconf-4741bis]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [I-D.ietf-netconf-rfc4742bis].

There are a number of data nodes defined in the YANG module and submodules which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

<list subtrees and data nodes and state why they are sensitive>

Some of the readable data nodes in the YANG module and submodules 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 subtrees and data nodes and their sensitivity/vulnerability:

<list subtrees and data nodes and state why they are sensitive>

7. Acknowledgments

The authors want to thank David Spakes for his review and valuable comments.

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.

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

8.2. Informative References

[I-D.ietf-netconf-4741bis] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", draft-ietf-netconf-4741bis-10 (work in progress), March 2011.

[I-D.ietf-netconf-rfc4742bis] Wasserman, M. and T. Goddard, "Using the NETCONF Configuration Protocol over Secure Shell (SSH)", draft-ietf-netconf-rfc4742bis-08 (work in progress), March 2011.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[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.

[RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, 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.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC3584] Frye, R., Levi, D., Routhier, S., and B. Wijnen, "Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework", BCP 74, RFC 3584, August 2003.

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

Appendix A. Example configurations

A.1. Engine Configuration Example

Below is an XML instance document showing a configuration of an SNMP engine listening on UDP port 161 on IPv4 and IPv6 endpoints and accepting SNMPv2c and SNMPv3 messages.

```
<snmp xmlns="urn:ietf:params:xml:ns:yang:ietf-snmp">
  <engine>
    <enabled>true</enabled>
    <listen>
      <udp>
        <ip>0.0.0.0</ip>
        <port>161</port>
      </udp>
      <udp>
        <ip>::</ip>
        <port>161</port>
      </udp>
    </listen>
    <version>
      <v2c/>
      <v3/>
    </version>
    <engine-id>80:00:02:b8:04:61:62:63</engine-id>
  </engine>
</snmp>
```

A.2. Community Configuration Example

Below is an XML instance document showing a configuration that maps the community name "public" to the security-name "community-public" on the local engine with the default context name. The target tag "community-public-access" filters the access to this community name.

```
<snmp xmlns="urn:ietf:params:xml:ns:yang:ietf-snmp">
  <community>
    <index>1</index>
    <text-name>public</text-name>
    <security-name>community-public</security-name>
    <target-tag>community-public-access</target-tag>
  </community>
</snmp>
```

[TODO: Add example for the target params table.]

A.3. User-based Security Model Configuration Example

Below is an XML instance document showing the configuration of a local user "joey" who has no authentication or privacy keys. For the remote SNMP engine identified by the snmpEngineID '800002b804616263'H, two users are configure. The user "matt" has a localized SHA authentication key and the user "russ" has a localized SHA authentication key and an AES encryption key.

```
<snmp xmlns="urn:ietf:params:xml:ns:yang:ietf-snmp">
  <usm>
    <local>
      <user>
        <name>joey</name>
      </user>
    </local>
    <remote>
      <engine-id>00:00:00:00:00:00:00:00:00:00:00:02</engine-id>
      <user>
        <name>matt</name>
        <auth>
          <sha>
            <!--
                The 'key' value is split into two lines to match
                the RFC formatting rules.
            -->
            <key>66:95:fe:bc:92:88:e3:62:82:23:
                 5f:c7:15:1f:12:84:97:b3:8f:3f</key>
          </sha>
        </auth>
      </user>
      <user>
        <name>russ</name>
        <auth>
          <sha>
            <!--
                The 'key' value is split into two lines to match
                the RFC formatting rules.
            -->
            <key>66:95:fe:bc:92:88:e3:62:82:23:
                 5f:c7:15:1f:12:84:97:b3:8f:3f</key>
          </sha>
        </auth>
        <priv>
          <aes>
            <key>66:95:fe:bc:92:88:e3:62:82:23:5f:c7:15:1f:12:84</key>
          </aes>
        </priv>
```

```
        </user>
      </remote>
    </usm>
    <target>
      <name>bluebox</name>
      <udp>
        <ip>2001:db8::abcd</ip>
        <port>161</port>
      </udp>
      <tag>blue</tag>
      <usm>
        <user-name>matt</user-name>
        <security-level>auth-no-priv</security-level>
      </usm>
    </target>
  </snmp>
```

A.4. Target and Notification Configuration Example

Below is an XML instance document showing the configuration of a notification generator application (see Appendix A of [RFC3413]). Note that the USM specific objects are defined in the ietf-snmp-usm.yang submodule.

```
   <snmp xmlns="urn:ietf:params:xml:ns:yang:ietf-snmp">
     <target>
       <name>addr1</name>
       <udp>
         <ip>128.1.2.3</ip>
         <port>162</port>
       </udp>
       <tag>group1</tag>
       <usm>
         <user-name>joe</user-name>
         <security-level>auth-no-priv</security-level>
       </usm>
     </target>
     <target>
       <name>addr2</name>
       <udp>
         <ip>128.2.4.6</ip>
         <port>162</port>
       </udp>
       <tag>group1</tag>
       <usm>
         <user-name>joe</user-name>
         <security-level>auth-no-priv</security-level>
       </usm>
     </target>
     <target>
       <name>addr3</name>
       <udp>
         <ip>128.1.5.9</ip>
         <port>162</port>
       </udp>
       <tag>group2</tag>
       <usm>
         <user-name>bob</user-name>
         <security-level>auth-priv</security-level>
       </usm>
     </target>
     <notify>
       <name>group1</name>
       <tag>group1</tag>
       <type>trap</type>
     </notify>
     <notify>
       <name>group2</name>
       <tag>group2</tag>
       <type>trap</type>
     </notify>
   </snmp>
```

A.5. Proxy Configuration Example

[TODO]

A.6. View-based Access Control Model Configuration Example

Below is an XML instance document showing the minimum-secure VACM configuration (see Appendix A of [RFC3415]).

```
<snmp xmlns="urn:ietf:params:xml:ns:yang:ietf-snmp">
  <vacm>
    <group>
      <name>initial</name>
      <member>
        <security-name>initial</security-name>
        <security-model>usm</security-model>
      </member>
      <access>
        <context></context>
        <security-model>usm</security-model>
        <security-level>no-auth-no-priv</security-level>
        <read-view>restricted</read-view>
        <notify-view>restricted</notify-view>
      </access>
      <access>
        <context></context>
        <security-model>usm</security-model>
        <security-level>auth-no-priv</security-level>
        <read-view>internet</read-view>
        <write-view>internet</write-view>
        <notify-view>internet</notify-view>
      </access>
    </group>
    <view>
      <name>initial</name>
      <include>1.3.6.1</include>
    </view>
    <view>
      <name>restricted</name>
      <include>1.3.6.1</include>
    </view>
  </vacm>
</snmp>
```

The following XML instance document shows the semi-secure VACM configuration (only the view configuration is different).

```
   <snmp xmlns="urn:ietf:params:xml:ns:yang:ietf-snmp">
     <vacm>
       <group>
         <name>initial</name>
         <member>
           <security-name>initial</security-name>
           <security-model>usm</security-model>
         </member>
         <access>
           <context></context>
           <security-model>usm</security-model>
           <security-level>no-auth-no-priv</security-level>
           <read-view>restricted</read-view>
           <notify-view>restricted</notify-view>
         </access>
         <access>
           <context></context>
           <security-model>usm</security-model>
           <security-level>auth-no-priv</security-level>
           <read-view>internet</read-view>
           <write-view>internet</write-view>
           <notify-view>internet</notify-view>
         </access>
       </group>
       <view>
         <name>initial</name>
         <include>1.3.6.1</include>
       </view>
       <view>
         <name>restricted</name>
         <include>1.3.6.1</include>
       </view>
     </vacm>
   </snmp>
```

Authors' Addresses

Martin Bjorklund
Tail-f Systems

Email: mbj@tail-f.com

Juergen Schoenwaelder
Jacobs University

Email: j.schoenwaelder@jacobs-university.de