Network Working Group
Request for Comments: 2864
Category: Standards Track

K. McCloghrie
Cisco Systems
G. Hanson
ADC Telecommunications

June 2000

The Inverted Stack Table Extension to the Interfaces Group MIB

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects which provide an inverted mapping of the interface stack table used for managing network interfaces.

2. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [1].

- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in STD 16, RFC 1155 [2], STD 16, RFC 1212 [3] and RFC 1215 [4]. The second version, called SMIv2, is described in STD 58, which consists of RFC 2578 [5], RFC 2579 [6] and RFC 2580 [7].

- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [8]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [9] and RFC 1906 [10]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [10], RFC 2572 [11] and RFC 2574 [12].

- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [8]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [13].

- A set of fundamental applications described in RFC 2573 [14] and the view-based access control mechanism described in RFC 2575 [15].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [18].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIv2. A MIB conforming to the SMIv1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (e.g., use of Counter64). Some machine readable information in SMIv2 will be converted into textual descriptions in SMIv1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB.

3. Interface Sub-Layers and the ifStackTable

MIB-II [16] defines objects for managing network interfaces by providing a generic interface definition together with the ability to define media-specific extensions. The generic objects are known as the 'interfaces' group.

Experience in defining media-specific extensions showed the need to distinguish between the multiple sub-layers beneath the internetwork-layer. Consider, for example, an interface with PPP running over an HDLC link which uses a RS232-like connector. Each of these sub-layers has its own media-specific MIB module.

The latest definition of the 'interfaces' group in the IF-MIB [17] satisfies this need by having each sub-layer be represented by its own conceptual row in the ifTable. It also defines an additional MIB table, the ifStackTable, to identify the "superior" and "subordinate" sub-layers through ifIndex "pointers" to the appropriate conceptual rows in the ifTable.

Each conceptual row in the ifStackTable represents a relationship between two interfaces, where this relationship is that the "higher-layer" interface runs "on top" of the "lower-layer" interface. For example, if a PPP module operated directly over a serial interface, the PPP module would be a "higher layer" to the serial interface, and the serial interface would be a "lower layer" to the PPP module. This concept of "higher-layer" and "lower-layer" is the same as embodied in the definitions of the ifTable's packet counters.

The ifStackTable is INDEX-ed by the ifIndex values of the two interfaces involved in the relationship. By necessity, one of these ifIndex values must come first, and the IF-MIB chose to have the higher-layer interface first, and the lower-layer interface second. Due to this, it is straight-forward for a Network Management application to read a subset of the ifStackTable and thereby determine the interfaces which run underneath a particular interface. However, to determine which interfaces run on top of a particular interface, a Network Management application has no alternative but to read the whole table. This is very inefficient when querying a device which has many interfaces, and many conceptual rows in its ifStackTable.

This MIB provides an inverted Interfaces Stack Table, the ifInvStackTable. While it contains no additional information beyond that already contained in the ifStackTable, the ifInvStackTable has the ifIndex values in its INDEX clause in the reverse order, i.e., the lower-layer interface first, and the higher-layer interface second. As a result, the ifInvStackTable is an inverted version of

the same information contained in the ifStackTable. Thus, the ifInvStackTable provides an efficient means for a Network Management application to read a subset of the ifStackTable and thereby determine which interfaces run on top of a particular interface.

4. Definitions

```
IF-INVERTED-STACK-MIB DEFINITIONS ::= BEGIN

IMPORTS
  MODULE-IDENTITY, OBJECT-TYPE, mib-2      FROM SNMPv2-SMI
  RowStatus                                FROM SNMPv2-TC
  MODULE-COMPLIANCE, OBJECT-GROUP          FROM SNMPv2-CONF
  ifStackGroup2,
  ifStackHigherLayer, ifStackLowerLayer    FROM IF-MIB;

ifInvertedStackMIB MODULE-IDENTITY
  LAST-UPDATED "200006140000Z"
  ORGANIZATION "IETF Interfaces MIB Working Group"
  CONTACT-INFO
          "   Keith McCloghrie
              Cisco Systems, Inc.
              170 West Tasman Drive
              San Jose, CA  95134-1706
              US

              408-526-5260
              kzm@cisco.com"
  DESCRIPTION
          "The MIB module which provides the Inverted Stack Table for
          interface sub-layers."
  REVISION      "200006140000Z"
  DESCRIPTION
          "Initial revision, published as RFC 2864"
  ::= { mib-2 77 }

ifInvMIBObjects OBJECT IDENTIFIER ::= { ifInvertedStackMIB 1 }

--
--           The Inverted Interface Stack Group
--

ifInvStackTable  OBJECT-TYPE
   SYNTAX        SEQUENCE OF IfInvStackEntry
   MAX-ACCESS    not-accessible
   STATUS        current
   DESCRIPTION
          "A table containing information on the relationships between
```

```
          the multiple sub-layers of network interfaces.  In
          particular, it contains information on which sub-layers run
          'underneath' which other sub-layers, where each sub-layer
          corresponds to a conceptual row in the ifTable.  For
          example, when the sub-layer with ifIndex value x runs
          underneath the sub-layer with ifIndex value y, then this
          table contains:

            ifInvStackStatus.x.y=active

          For each ifIndex value, z, which identifies an active
          interface, there are always at least two instantiated rows
          in this table associated with z.  For one of these rows, z
          is the value of ifStackHigherLayer; for the other, z is the
          value of ifStackLowerLayer.  (If z is not involved in
          multiplexing, then these are the only two rows associated
          with z.)

          For example, two rows exist even for an interface which has
          no others stacked on top or below it:

            ifInvStackStatus.z.0=active
            ifInvStackStatus.0.z=active

          This table contains exactly the same number of rows as the
          ifStackTable, but the rows appear in a different order."
   REFERENCE
          "ifStackTable of RFC 2863"
   ::= { ifInvMIBObjects 1 }

ifInvStackEntry  OBJECT-TYPE
   SYNTAX        IfInvStackEntry
   MAX-ACCESS    not-accessible
   STATUS        current
   DESCRIPTION
          "Information on a particular relationship between two sub-
          layers, specifying that one sub-layer runs underneath the
          other sub-layer.  Each sub-layer corresponds to a conceptual
          row in the ifTable."
   INDEX { ifStackLowerLayer, ifStackHigherLayer }
   ::= { ifInvStackTable 1 }

IfInvStackEntry ::=
  SEQUENCE {
      ifInvStackStatus       RowStatus
   }

ifInvStackStatus  OBJECT-TYPE
```

```
  SYNTAX         RowStatus
  MAX-ACCESS     read-only
  STATUS         current
  DESCRIPTION
          "The status of the relationship between two sub-layers.

          An instance of this object exists for each instance of the
          ifStackStatus object, and vice versa.  For example, if the
          variable ifStackStatus.H.L exists, then the variable
          ifInvStackStatus.L.H must also exist, and vice versa.  In
          addition, the two variables always have the same value.

          However, unlike ifStackStatus, the ifInvStackStatus object
          is NOT write-able.  A network management application wishing
          to change a relationship between sub-layers H and L cannot
          do so by modifying the value of ifInvStackStatus.L.H, but
          must instead modify the value of ifStackStatus.H.L.  After
          the ifStackTable is modified, the change will be reflected
          in this table."
  ::= { ifInvStackEntry 1 }

-- conformance information

ifInvConformance OBJECT IDENTIFIER ::= { ifInvMIBObjects 2 }

ifInvGroups      OBJECT IDENTIFIER ::= { ifInvConformance 1 }
ifInvCompliances OBJECT IDENTIFIER ::= { ifInvConformance 2 }

-- compliance statements

ifInvCompliance MODULE-COMPLIANCE
  STATUS  current
  DESCRIPTION
          "The compliance statement for SNMP entities which provide
          inverted information on the layering of network interfaces."

  MODULE  -- this module
      MANDATORY-GROUPS { ifInvStackGroup }

      OBJECT       ifInvStackStatus
      SYNTAX       INTEGER { active(1) }
      DESCRIPTION
          "Support is only required for 'active'."
```

```
  MODULE  IF-MIB
      MANDATORY-GROUPS { ifStackGroup2 }

  ::= { ifInvCompliances 1 }

-- units of conformance

ifInvStackGroup    OBJECT-GROUP
  OBJECTS { ifInvStackStatus }
  STATUS  current
  DESCRIPTION
          "A collection of objects providing inverted information on
          the layering of MIB-II interfaces."
  ::= { ifInvGroups 1 }

END
```

5. Acknowledgements

This memo has been produced by the IETF's Interfaces MIB working-group.

6. References

[1] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, January 1998.

[2] Rose, M. and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.

[3] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[4] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

[5] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[6] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[7] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[8] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.

[9] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.

[10] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.

[11] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, January 1998.

[12] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, January 1998.

[13] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.

[14] Levi, D., Meyer, P. and B. Stewart, "SNMP Applications", RFC 2573, January 1998.

[15] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, January 1998.

[16] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets - MIB-II", STD 17, RFC 1213, March 1991.

[17] McCloghrie, K. and F. Kastenholz, "The Interface Group MIB", RFC 2863, June 2000.

[18] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.

7. Security Considerations

There are no management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. So, if this MIB is implemented correctly, then there is no risk that an intruder can alter or create any management objects of this MIB via direct SNMP SET operations.

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB.

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model RFC 2574 [12] and the View- based Access Control Model RFC 2575 [15] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

8. Authors' Addresses

Keith McCloghrie
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706

Phone: 408-526-5260
EMail: kzm@cisco.com

Gary Hanson
ADC Telecommunications
14375 NW Science Park Drive
Portland, Oregon, 97229

Phone: (800)733-5511 x6333
EMail: gary_hanson@adc.com

9. Notice on Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

10. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.