Network Working Group
Request for Comments: 2720
Obsoletes: 2064
Category: Standards Track

N. Brownlee
The University of Auckland

October 1999

Traffic Flow Measurement: Meter MIB

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

The RTFM Traffic Measurement Architecture provides a general framework for describing and measuring network traffic flows. Flows are defined in terms of their Address Attribute values and measured by a 'Traffic Meter'.

This document defines a Management Information Base (MIB) for use in controlling an RTFM Traffic Meter, in particular for specifying the flows to be measured. It also provides an efficient mechanism for retrieving flow data from the meter using SNMP. Security issues concerning the operation of traffic meters are summarised.

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 objects for managing and collecting data from network Realtime Traffic Flow Meters, as described in [RTFM-ARC].

The MIB is 'basic' in the sense that it provides more than enough information for everyday traffic measurment. Furthermore, it can be easily extended by adding new attributes as required. The RTFM Working group is actively pursuing the development of the meter in this way.

2 The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].

- 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 [RFC1155], STD 16, RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in STD 58, RFC 2578 [RFC2578], RFC 2579 [RFC2579] and RFC 2580 [RFC2580].

- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].

- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].

- A set of fundamental applications described in RFC 2573 [RFC2573] and the view-based access control mechanism described in RFC 2575 [RFC2575].

A more detailed introduction to the current SNMP Management Framework can be found in [RFC2570].

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 (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 Overview

Traffic Flow Measurement seeks to provide a well-defined method for gathering traffic flow information from networks and internetworks. The background for this is given in "Internet Accounting Background" [ACT-BKG]. The Realtime Traffic Flow Measurement (rtfm) Working Group has produced a measurement architecture to achieve this goal; this is documented in "Traffic Flow Measurement: Architecture" [RTFM-ARC]. The architecture defines three entities:

- METERS, which observe network traffic flows and build up a table of flow data records for them,

- METER READERS, which collect traffic flow data from meters, and

- MANAGERS, which oversee the operation of meters and meter readers.

This memo defines the SNMP management information for a Traffic Flow Meter (TFM). Work in this field was begun by the Internet Accounting Working Group. It has been further developed and expanded by the Realtime Traffic Flow Measurement Working Group.

3.1 Scope of Definitions, Textual Conventions

All objects defined in this memo are registered in a single subtree within the mib-2 namespace [MIB-II, RFC2578], and are for use in network devices which may perform a PDU forwarding or monitoring function. For these devices, this MIB defines a group of objects with an SMI Network Management MGMT Code [ASG-NBR] of 40, i.e.

```
flowMIB OBJECT IDENTIFIER ::=  mib-2 40
```

as defined below.

The RTFM Meter MIB was first produced and tested using SNMPv1. It was converted into SNMPv2 following the guidelines in [RFC1908].

3.2 Usage of the MIB variables

The MIB is organised in four parts - control, data, rules and conformance statements.

The rules implement the set of packet-matching actions, as described in the "Traffic Flow Measurment: Architecture" document [RTFM-ARC]. In addition they provide for BASIC-style subroutines, allowing a network manager to dramatically reduce the number of rules required to monitor a large network.

Traffic flows are identified by a set of attributes for each of their end-points. Attributes include network addresses for each layer of the network protocol stack, and 'subscriber ids', which may be used to identify an accountable entity for the flow.

The conformance statements are set out as defined in [RFC2580]. They explain what must be implemented in a meter which claims to conform to this MIB.

To retrieve flow data one could simply do a linear scan of the flow table. This would certainly work, but would require a lot of protocol exchanges. To reduce the overhead in retrieving flow data the flow table uses a TimeFilter variable, defined as a Textual Convention in the RMON2 MIB [RMON2-MIB].

As an alternative method of reading flow data, the MIB provides a view of the flow table called the flowDataPackageTable. This is (logically) a four-dimensional array, subscripted by package selector, RuleSet, activity time and starting flow number. The package selector is a sequence of bytes which specifies a list of flow attributes.

A data package (as returned by the meter) is a sequence of values for the attributes specified in its selector, encoded using the Basic Encoding Rules [ASN-BER]. It allows a meter reader to retrieve all the attribute values it requires in a single MIB object. This, when used together with SNMPv2's GetBulk request, allows a meter reader to scan the flow table and upload a specified set of attribute values for flows which have changed since the last reading, and which were created by a specified rule set.

One aspect of data collection which needs emphasis is that all the MIB variables are set up to allow multiple independent meter readers to work properly, i.e. the flow table indexes are stateless. An alternative approach would have been to 'snapshot' the flow table, which would mean that the meter readers would have to be synchronized. The stateless approach does mean that two meter readers will never return exactly the same set of traffic counts, but over long periods (e.g. 15-minute collections over a day) the discrepancies are acceptable. If one really needs a snapshot, this can be achieved by switching to an identical rule set with a different RuleSet number, hence asynchronous collections may be regarded as a useful generalisation of synchronised ones.

The control variables are the minimum set required for a meter reader. Their number has been whittled down as experience has been gained with the MIB implementation. A few of them are 'general', i.e. they control the overall behaviour of the meter. These are set by a single 'master' manager, and no other manager should attempt to change their values. The decision as to which manager is the ' master' must be made by the network operations personnel responsible; this MIB does not attempt to define any interaction between managers.

There are three other groups of control variables, arranged into tables in the same way as in the RMON2 MIB [RMON2-MIB]. They are used as follows:

- RULE SET INFO: Before attempting to download a RuleSet, a manager must create a row in the flowRuleSetInfoTable and set its flowRuleInfoSize to a value large enough to hold the RuleSet. When the rule set is ready the manager must set flowRuleInfoRulesReady to 'true', indicating that the rule set is ready for use (but not yet 'running').

- METER READER INFO: Any meter reader wishing to collect data reliably for all flows from a RuleSet should first create a row in the flowReaderInfoTable with flowReaderRuleSet set to that RuleSet's index in the flowRuleSetInfoTable. It should write that row's flowReaderLastTime object each time it starts a collection

pass through the flow table. The meter will not recover a flow's memory until every meter reader holding a row for that flow's RuleSet has collected the flow's data.

- MANAGER INFO: Any manager wishing to run a RuleSet in the meter must create a row in the flowManagerInfo table, specifying the desired RuleSet to run and its corresponding 'standby' RuleSet (if one is desired). A current RuleSet is 'running' if its flowManagerRunningStandby value is false(2), similarly a standby RuleSet is 'running' if flowManagerRunningStandby is true(1).

Times within the meter are in terms of its Uptime, i.e. centiseconds since the meter started. For meters implemented as self-contained SNMP agents this will be the same as sysUptime, but this may not be true for meters implemented as subagents. Managers can read the meter's Uptime when neccessary (e.g. to set a TimeFilter value) by setting flowReaderLastTime, then reading its new value.

4 Definitions

```
FLOW-METER-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE,
    Counter32, Counter64, Integer32, mib-2
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION, RowStatus, TimeStamp, TruthValue
        FROM SNMPv2-TC
    OBJECT-GROUP, MODULE-COMPLIANCE
        FROM SNMPv2-CONF
    ifIndex
        FROM IF-MIB
    TimeFilter
        FROM RMON2-MIB;

flowMIB MODULE-IDENTITY
    LAST-UPDATED "9910250000Z" -- October 25, 1999
    ORGANIZATION "IETF Realtime Traffic Flow Measurement Working Group"
    CONTACT-INFO
        "Nevil Brownlee, The University of Auckland

        Postal: Information Technology Sytems & Services
                The University of Auckland
                Private Bag 92-019
                Auckland, New Zealand

        Phone:  +64 9 373 7599 x8941
        E-mail: n.brownlee@auckland.ac.nz"
```

```
    DESCRIPTION
        "MIB for the RTFM Traffic Flow Meter."

    REVISION "9910250000Z"
    DESCRIPTION
        "Initial Version, published as RFC 2720."

     REVISION "9908301250Z"
     DESCRIPTION
         "UTF8OwnerString Textual Convention added, and used to
         replace OwnerString.  Conceptually the same as OwnerString,
         but facilitating internationalisation by using UTF-8
         encoding for its characters rather than US-ASCII."

    REVISION "9908191010Z"
    DESCRIPTION
        "Changes to SIZE specification for two variables:
          - flowRuleInfoName SIZE specified as (0..127)
          - flowRuleIndex SIZE increased to (1..2147483647)"

    REVISION "9712230937Z"
    DESCRIPTION
        "Two further variables deprecated:
          - flowRuleInfoRulesReady (use flowRuleInfoStatus intead)
          - flowDataStatus (contains no useful information)"

    REVISION "9707071715Z"
    DESCRIPTION
        "Significant changes since RFC 2064 include:
          - flowDataPackageTable added
          - flowColumnActivityTable deprecated
          - flowManagerCounterWrap deprecated"

    REVISION "9603080208Z"
    DESCRIPTION
        "Initial version of this MIB (RFC 2064)"
    ::= { mib-2 40 }

flowControl         OBJECT IDENTIFIER ::= { flowMIB 1 }

flowData            OBJECT IDENTIFIER ::= { flowMIB 2 }

flowRules           OBJECT IDENTIFIER ::= { flowMIB 3 }

flowMIBConformance  OBJECT IDENTIFIER ::= { flowMIB 4 }

-- Textual Conventions
```

```
UTF8OwnerString ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "127t"
    STATUS  current
    DESCRIPTION
        "An administratively assigned name for the owner of a
        resource, conceptually the same as OwnerString in the RMON
        MIB [RMON-MIB].

        To facilitate internationalisation, this name information
        is represented using the ISO/IEC IS 10646-1 character set,
        encoded as an octet string using the UTF-8 transformation
        format described in the UTF-8 standard [UTF-8]."
    SYNTAX  OCTET STRING (SIZE (0..127))

PeerType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Indicates the type of a PeerAddress (see below).  The values
        used are from the 'Address Family Numbers' section of the
        Assigned Numbers RFC [ASG-NBR].  Peer types from other address
        families may also be used, provided only that they are
        identified by their assigned Address Family numbers."
    SYNTAX  INTEGER {
        ipv4(1),
        ipv6(2),
        nsap(3),
        ipx(11),
        appletalk(12),
        decnet(13) }

PeerAddress ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Specifies the value of a peer address for various network
        protocols.  Address format depends on the actual protocol,
        as indicated below:

        IPv4:        ipv4(1)
            4-octet IpAddress  (defined in the SNMPv2 SMI [RFC2578])

        IPv6:        ipv6(2)
            16-octet IpAddress  (defined in the
                                 IPv6 Addressing RFC [V6-ADDR])

        CLNS:        nsap(3)
            NsapAddress  (defined in the SNMPv2 SMI [RFC2578])

        Novell:      ipx(11)
```

```
            4-octet Network number,
            6-octet Host number (MAC address)

        AppleTalk:   appletalk(12)
            2-octet Network number (sixteen bits),
            1-octet Host number (eight bits)

        DECnet:      decnet(13)
            1-octet Area number (in low-order six bits),
            2-octet Host number (in low-order ten bits)
        "
    SYNTAX OCTET STRING (SIZE (3..20))

AdjacentType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Indicates the type of an adjacent address.  May be a medium
        type or (if metering is taking place inside a tunnel) a
        PeerType (see above).

        The values used for IEEE 802 medium types are from the 'Network
        Management Parameters (ifType definitions)' section of the
        Assigned Numbers RFC [ASG-NBR].  Other medium types may also
        be used, provided only that they are identified by their
        assigned ifType numbers."
    SYNTAX  INTEGER {
        ip(1),
        nsap(3),
        ethernet(7),  -- ethernet-like [ENET-OBJ],
                      --    includes ethernet-csmacd(6)
        tokenring(9),
        ipx(11),
        appletalk(12),
        decnet(13),
        fddi(15) }

AdjacentAddress ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Specifies the value of an adjacent address.  May be a Medium
        Access Control (MAC) address or (if metering is taking place
        inside a tunnel) a PeerAddress (see above).

        MAC Address format depends on the actual medium, as follows:

        Ethernet:     ethernet(7)
            6-octet 802.3 MAC address in 'canonical' order
```

```
        Token Ring:   tokenring(9)
            6-octet 802.5 MAC address in 'canonical' order

        FDDI:         fddi(15)
            FddiMACLongAddress, i.e. a 6-octet MAC address
            in 'canonical' order  (defined in [FDDI-MIB])
        "
    SYNTAX OCTET STRING (SIZE (3..20))

TransportType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Indicates the type of a TransportAddress (see below).  Values
        will depend on the actual protocol; for IP they will be those
        given in the 'Protocol Numbers' section of the  Assigned Numbers
        RFC [ASG-NBR], including icmp(1), tcp(6) and udp(17)."
    SYNTAX  Integer32 (1..255)

TransportAddress ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Specifies the value of a transport address for various
        network protocols.  Format as follows:

        IP:
            2-octet UDP or TCP port number

        Other protocols:
            2-octet port number
        "
    SYNTAX OCTET STRING (SIZE (2))

RuleAddress ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Specifies the value of an address.  Is a superset of
        MediumAddress, PeerAddress and TransportAddress."
    SYNTAX OCTET STRING (SIZE (2..20))

FlowAttributeNumber ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Uniquely identifies an attribute within a flow data record."
    SYNTAX  INTEGER {
        flowIndex(1),
        flowStatus(2),
        flowTimeMark(3),
```

```
        sourceInterface(4),
        sourceAdjacentType(5),
        sourceAdjacentAddress(6),
        sourceAdjacentMask(7),
        sourcePeerType(8),
        sourcePeerAddress(9),
        sourcePeerMask(10),
        sourceTransType(11),
        sourceTransAddress(12),
        sourceTransMask(13),

        destInterface(14),
        destAdjacentType(15),
        destAdjacentAddress(16),
        destAdjacentMask(17),
        destPeerType(18),
        destPeerAddress(19),
        destPeerMask(20),
        destTransType(21),
        destTransAddress(22),
        destTransMask(23),

        pduScale(24),
        octetScale(25),

        ruleSet(26),
        toOctets(27),             -- Source-to-Dest
        toPDUs(28),
        fromOctets(29),           -- Dest-to-Source
        fromPDUs(30),
        firstTime(31),            -- Activity times
        lastActiveTime(32),

        sourceSubscriberID(33),   -- Subscriber ID
        destSubscriberID(34),
        sessionID(35),

        sourceClass(36),          -- Computed attributes
        destClass(37),
        flowClass(38),
        sourceKind(39),
        destKind(40),
        flowKind(41) }

RuleAttributeNumber ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Uniquely identifies an attribute which may be tested in
```

```
        a rule.  These include attributes whose values come directly
        from (or are computed from) the flow's packets, and the five
        'meter' variables used to hold an Attribute Number."
    SYNTAX  INTEGER {
        null(0),
        sourceInterface(4),       -- Source Address
        sourceAdjacentType(5),
        sourceAdjacentAddress(6),
        sourcePeerType(8),
        sourcePeerAddress(9),
        sourceTransType(11),
        sourceTransAddress(12),

        destInterface(14),        -- Dest Address
        destAdjacentType(15),
        destAdjacentAddress(16),
        destPeerType(18),
        destPeerAddress(19),
        destTransType(21),
        destTransAddress(22),

        sourceSubscriberID(33),   -- Subscriber ID
        destSubscriberID(34),
        sessionID(35),

        sourceClass(36),          -- Computed attributes
        destClass(37),
        flowClass(38),
        sourceKind(39),
        destKind(40),
        flowKind(41),

        matchingStoD(50),         -- Packet matching

        v1(51),                   -- Meter variables
        v2(52),
        v3(53),
        v4(54),
        v5(55) }

ActionNumber ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
        "Uniquely identifies the action of a rule, i.e. the Pattern
        Matching Engine's opcode number.  Details of the opcodes
        are given in the 'Traffic Flow Measurement: Architecture'
        document [RTFM-ARC]."
    SYNTAX  INTEGER {
```

```
        ignore(1),
        noMatch(2),
        count(3),
        countPkt(4),
        return(5),
        gosub(6),
        gosubAct(7),
        assign(8),
        assignAct(9),
        goto(10),
        gotoAct(11),
        pushRuleTo(12),
        pushRuleToAct(13),
        pushPktTo(14),
        pushPktToAct(15),
        popTo(16),
        popToAct(17) }

--
-- Control Group:  RuleSet Info Table
--

flowRuleSetInfoTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowRuleSetInfoEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "An array of information about the RuleSets held in the
        meter.

        Any manager may configure a new RuleSet for the meter by
        creating a row in this table with status active(1), and setting
        values for all the objects in its rules.  At this stage the new
        RuleSet is available but not 'running', i.e. it is not being
        used by the meter to produce entries in the flow table.

        To actually 'run' a RuleSet a manager must create a row in
        the flowManagerInfoTable, set it's flowManagerStatus to
        active(1), and set either its CurrentRuleSet or StandbyRuleSet
        to point to the RuleSet to be run.

        Once a RuleSet is running a manager may not change any of the
        objects within the RuleSet itself.  Any attempt to do so should
        result in a notWritable(17) SNMP error-status for such objects.

        A manager may stop a RuleSet running by removing all
        references to it in the flowManagerInfoTable (i.e. by setting
        CurrentRuleSet and StandbyRuleSet values to 0).  This provides
```

```
        a way to stop RuleSets left running if a manager fails.
        For example, when a manager is started, it could search the
        meter's flowManager table and stop all RuleSets having a
        specified value of flowRuleInfoOwner.

        To prevent a manager from interfering with variables belonging
        to another manager, the meter should use MIB views [RFC2575] so
        as to limit each manager's access to the meter's variables,
        effectively dividing the single meter into several virtual
        meters, one for each independent manager."
    ::= { flowControl 1 }

flowRuleSetInfoEntry OBJECT-TYPE
    SYNTAX  FlowRuleSetInfoEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Information about a particular RuleSet."
    INDEX  { flowRuleInfoIndex }
    ::= { flowRuleSetInfoTable 1 }

FlowRuleSetInfoEntry ::= SEQUENCE {
    flowRuleInfoIndex         Integer32,
    flowRuleInfoSize          Integer32,
    flowRuleInfoOwner         UTF8OwnerString,
    flowRuleInfoTimeStamp     TimeStamp,
    flowRuleInfoStatus        RowStatus,
    flowRuleInfoName          OCTET STRING,
    flowRuleInfoRulesReady    TruthValue,
    flowRuleInfoFlowRecords   Integer32
    }

flowRuleInfoIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "An index which selects an entry in the flowRuleSetInfoTable.
        Each such entry contains control information for a particular
        RuleSet which the meter may run."
    ::= { flowRuleSetInfoEntry 1 }

flowRuleInfoSize OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Number of rules in this RuleSet.  Setting this variable will
```

```
        cause the meter to allocate space for these rules."
    ::= { flowRuleSetInfoEntry 2 }

flowRuleInfoOwner OBJECT-TYPE
    SYNTAX  UTF8OwnerString
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Identifies the manager which 'owns' this RuleSet.  A manager
        must set this variable when creating a row in this table."
    ::= { flowRuleSetInfoEntry 3 }

flowRuleInfoTimeStamp OBJECT-TYPE
    SYNTAX  TimeStamp
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Time this row's associated RuleSet was last changed."
    ::= { flowRuleSetInfoEntry 4 }

flowRuleInfoStatus OBJECT-TYPE
    SYNTAX  RowStatus
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "The status of this flowRuleSetInfoEntry. If this value is
        not active(1) the meter must not attempt to use the row's
        associated RuleSet.  Once its value has been set to active(1)
        a manager may not change any of the other variables in the
        row, nor the contents of the associated RuleSet.  Any attempt
        to do so should result in a notWritable(17) SNMP error-status
        for such variables or objects.

        To download a RuleSet, a manger could:
           - Locate an open slot in the RuleSetInfoTable.
           - Create a RuleSetInfoEntry by setting the status for this
             open slot to createAndWait(5).
           - Set flowRuleInfoSize and flowRuleInfoName as required.
           - Download the rules into the row's rule table.
           - Set flowRuleInfoStatus to active(1).

        The RuleSet would then be ready to run. The manager is not
        allowed to change the value of flowRuleInfoStatus from
        active(1) if the associated RuleSet is being referenced by any
        of the entries in the flowManagerInfoTable.

        Setting RuleInfoStatus to destroy(6) destroys the associated
        RuleSet together with any flow data collected by it."
```

```
    ::= { flowRuleSetInfoEntry 5 }

flowRuleInfoName OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (0..127))
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "An alphanumeric identifier used by managers and readers to
        identify a RuleSet.  For example, a manager wishing to run a
        RuleSet named WWW-FLOWS could search the flowRuleSetInfoTable
        to see whether the WWW-FLOWS RuleSet is already available on
        the meter.

        Note that references to RuleSets in the flowManagerInfoTable
        use indexes for their flowRuleSetInfoTable entries.  These may
        be different each time the RuleSet is loaded into a meter."
    ::= { flowRuleSetInfoEntry 6 }

flowRuleInfoRulesReady OBJECT-TYPE
    SYNTAX  TruthValue
    MAX-ACCESS  read-create
    STATUS  deprecated
    DESCRIPTION
        "Indicates whether the rules for this row's associated RuleSet
        are ready for use.  The meter will refuse to 'run' the RuleSet
        unless this variable has been set to true(1).
        While RulesReady is false(2), the manager may modify the
        RuleSet, for example by downloading rules into it."
    ::= { flowRuleSetInfoEntry 7 }

flowRuleInfoFlowRecords OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The number of entries in the flow table for this RuleSet.
        These may be current (waiting for collection by one or more
        meter readers) or idle (waiting for the meter to recover
        their memory)."
    ::= { flowRuleSetInfoEntry 8 }

--
-- Control Group:  Interface Info Table
--

flowInterfaceTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowInterfaceEntry
    MAX-ACCESS  not-accessible
```

```
    STATUS  current
    DESCRIPTION
        "An array of information specific to each meter interface."
    ::= { flowControl 2 }

flowInterfaceEntry OBJECT-TYPE
    SYNTAX  FlowInterfaceEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Information about a particular interface."
    INDEX   { ifIndex }
    ::= { flowInterfaceTable 1 }

FlowInterfaceEntry ::= SEQUENCE {
    flowInterfaceSampleRate   Integer32,
    flowInterfaceLostPackets  Counter32
    }

flowInterfaceSampleRate OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "The parameter N for statistical counting on this interface.
        Set to N to count 1/Nth of the packets appearing at this
        interface.  A sampling rate of 1 counts all packets.
        A sampling rate of 0 results in the interface being ignored
        by the meter.

        A meter should choose its own algorithm to introduce variance
        into the sampling so that exactly every Nth packet is counted.
        The IPPM Working Group's RFC 'Framework for IP Performance
        Metrics' [IPPM-FRM] explains why this should be done, and sets
        out an algorithm for doing it."
    DEFVAL { 1 }
    ::= { flowInterfaceEntry 1 }

flowInterfaceLostPackets OBJECT-TYPE
    SYNTAX  Counter32
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The number of packets the meter has lost for this interface.
        Such losses may occur because the meter has been unable to
        keep up with the traffic volume."
    ::= { flowInterfaceEntry 2 }
```

```
--
-- Control Group:  Meter Reader Info Table
--

-- Any meter reader wishing to collect data reliably for flows
-- should first create a row in this table.  It should write that
-- row's flowReaderLastTime object each time it starts a collection
-- pass through the flow table.

-- If a meter reader (MR) does not create a row in this table, e.g.
-- because its MIB view [RFC2575] did not allow MR create access to
-- flowReaderStatus, collection can still proceed but the meter will
-- not be aware of meter reader MR.  This could lead the meter to
-- recover flows before they have been collected by MR.

flowReaderInfoTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowReaderInfoEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "An array of information about meter readers which have
        registered their intent to collect flow data from this meter."
    ::= { flowControl 3 }

flowReaderInfoEntry OBJECT-TYPE
    SYNTAX  FlowReaderInfoEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Information about a particular meter reader."
    INDEX  { flowReaderIndex }
    ::= { flowReaderInfoTable 1 }

FlowReaderInfoEntry ::= SEQUENCE {
    flowReaderIndex            Integer32,
    flowReaderTimeout          Integer32,
    flowReaderOwner            UTF8OwnerString,
    flowReaderLastTime         TimeStamp,
    flowReaderPreviousTime     TimeStamp,
    flowReaderStatus           RowStatus,
    flowReaderRuleSet          Integer32
    }

flowReaderIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
```

```
        "An index which selects an entry in the flowReaderInfoTable."
    ::= { flowReaderInfoEntry 1 }

flowReaderTimeout OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Specifies the maximum time (in seconds) between flow data
        collections for this meter reader.  If this time elapses
        without a collection, the meter should assume that this meter
        reader has stopped collecting, and delete this row from the
        table.  A value of zero indicates that this row should not be
        timed out."
    ::= { flowReaderInfoEntry 2 }

flowReaderOwner OBJECT-TYPE
    SYNTAX  UTF8OwnerString
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Identifies the meter reader which created this row."
    ::= { flowReaderInfoEntry 3 }

flowReaderLastTime OBJECT-TYPE
    SYNTAX  TimeStamp
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Time this meter reader began its most recent data collection.

        This variable should be written by a meter reader as its first
        step in reading flow data.  The meter will set this LastTime
        value to its current Uptime, and set its PreviousTime value
        (below) to the old  LastTime.  This allows the meter to
        recover flows which have been inactive since PreviousTime,
        for these have been collected at least once.

        If the meter reader fails to write flowLastReadTime, collection
        may still proceed but the meter may not be able to recover
        inactive flows until the flowReaderTimeout has been reached
        for this entry."
    ::= { flowReaderInfoEntry 4 }

flowReaderPreviousTime OBJECT-TYPE
    SYNTAX  TimeStamp
    MAX-ACCESS  read-only
    STATUS  current
```

```
    DESCRIPTION
        "Time this meter reader began the collection before last."
    ::= { flowReaderInfoEntry 5 }

flowReaderStatus OBJECT-TYPE
    SYNTAX  RowStatus
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "The status of this FlowReaderInfoEntry. A value of active(1)
        implies that the associated reader should be collecting data
        from the meter.  Once this variable has been set to active(1)
        a manager may only change this row's flowReaderLastTime and
        flowReaderTimeout variables."
    ::= { flowReaderInfoEntry 6 }

flowReaderRuleSet OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "An index to the array of RuleSets.  Specifies a set of rules
        of interest to this meter reader.  The reader will attempt to
        collect any data generated by the meter for this RuleSet, and
        the meter will not recover the memory of any of the RuleSet's
        flows until this collection has taken place.  Note that a
        reader may have entries in this table for several RuleSets."
    ::= { flowReaderInfoEntry 7 }

--
-- Control Group:  Manager Info Table
--

-- Any manager wishing to run a RuleSet must create a row in this
-- table.  Once it has a table row, the manager may set the control
-- variables in its row so as to cause the meter to run any valid
-- RuleSet held by the meter.

-- A single manager may run several RuleSets; it must create a row
-- in this table for each of them.  In short, each row of this table
-- describes (and controls) a 'task' which the meter is executing.

flowManagerInfoTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowManagerInfoEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "An array of information about managers which have
```

```
        registered their intent to run RuleSets on this meter."
    ::= { flowControl 4 }

flowManagerInfoEntry OBJECT-TYPE
    SYNTAX  FlowManagerInfoEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Information about a particular meter 'task.'  By creating
        an entry in this table and activating it, a manager requests
        that the meter 'run' the indicated RuleSet.

        The entry also specifies a HighWaterMark and a StandbyRuleSet.
        If the meter's flow table usage exceeds this task's
        HighWaterMark the meter will stop running the task's
        CurrentRuleSet and switch to its StandbyRuleSet.

        If the value of the task's StandbyRuleSet is 0 when its
        HighWaterMark is exceeded, the meter simply stops running the
        task's CurrentRuleSet.  By careful selection of HighWaterMarks
        for the various tasks a manager can ensure that the most
        critical RuleSets are the last to stop running as the number
        of flows increases.

        When a manager has determined that the demand for flow table
        space has abated, it may cause the task to switch back to its
        CurrentRuleSet by setting its flowManagerRunningStandby
        variable to false(2)."
    INDEX  { flowManagerIndex }
    ::= { flowManagerInfoTable 1 }

FlowManagerInfoEntry ::= SEQUENCE {
    flowManagerIndex           Integer32,
    flowManagerCurrentRuleSet  Integer32,
    flowManagerStandbyRuleSet  Integer32,
    flowManagerHighWaterMark   Integer32,
    flowManagerCounterWrap     INTEGER,
    flowManagerOwner           UTF8OwnerString,
    flowManagerTimeStamp       TimeStamp,
    flowManagerStatus          RowStatus,
    flowManagerRunningStandby  TruthValue
    }

flowManagerIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
```

```
        "An index which selects an entry in the flowManagerInfoTable."
    ::= { flowManagerInfoEntry 1 }

flowManagerCurrentRuleSet OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Index to the array of RuleSets.  Specifies which set of
        rules is the 'current' one for this task.  The meter will
        be 'running' the current RuleSet if this row's
        flowManagerRunningStandby value is false(2).

        When the manager sets this variable the meter will stop using
        the task's old current RuleSet and start using the new one.
        Specifying RuleSet 0 (the empty set) stops flow measurement
        for this task."
    ::= { flowManagerInfoEntry 2 }

flowManagerStandbyRuleSet OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Index to the array of RuleSets.  After reaching HighWaterMark
        (see below) the manager will switch to using the task's
        StandbyRuleSet in place of its CurrentRuleSet.  For this to be
        effective the designated StandbyRuleSet should have a coarser
        reporting granularity then the CurrentRuleSet.  The manager may
        also need to decrease the meter reading interval so that the
        meter can recover flows measured by this task's CurrentRuleSet."
    DEFVAL { 0 }  -- No standby
    ::= { flowManagerInfoEntry 3 }

flowManagerHighWaterMark OBJECT-TYPE
    SYNTAX  Integer32 (0..100)
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "A value expressed as a percentage, interpreted by the meter
        as an indication of how full the flow table should be before
        it should switch to the standby RuleSet (if one has been
        specified) for this task.  Values of 0% or 100% disable the
        checking represented by this variable."
    ::= { flowManagerInfoEntry 4 }

flowManagerCounterWrap OBJECT-TYPE
    SYNTAX  INTEGER { wrap(1), scale(2) }
```

```
    MAX-ACCESS  read-create
    STATUS  deprecated
    DESCRIPTION
        "Specifies whether PDU and octet counters should wrap when
        they reach the top of their range (normal behaviour for
        Counter64 objects), or whether their scale factors should
        be used instead.  The combination of counter and scale
        factor allows counts to be returned as non-negative binary
        floating point numbers, with 64-bit mantissas and 8-bit
        exponents."
    DEFVAL { wrap }
    ::= { flowManagerInfoEntry 5 }

flowManagerOwner OBJECT-TYPE
    SYNTAX  UTF8OwnerString
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Identifies the manager which created this row."
    ::= { flowManagerInfoEntry 6 }

flowManagerTimeStamp OBJECT-TYPE
    SYNTAX  TimeStamp
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Time this row was last changed by its manager."
    ::= { flowManagerInfoEntry 7 }

flowManagerStatus OBJECT-TYPE
    SYNTAX  RowStatus
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "The status of this row in the flowManagerInfoTable.  A value
        of active(1) implies that this task may be activated, by
        setting its CurrentRuleSet and StandbyRuleSet variables.
        Its HighWaterMark and RunningStandby variables may also be
        changed."
    ::= { flowManagerInfoEntry 8 }

flowManagerRunningStandby OBJECT-TYPE
    SYNTAX  TruthValue
    MAX-ACCESS  read-create
    STATUS  current
    DESCRIPTION
        "Set to true(1) by the meter to indicate that it has switched
        to runnning this task's StandbyRuleSet in place of its
```

```
        CurrentRuleSet.  To switch back to the CurrentRuleSet, the
        manager may simply set this variable to false(2)."
    DEFVAL { false }
    ::= { flowManagerInfoEntry 9 }

--
-- Control Group:  General Meter Control Variables
--

flowFloodMark OBJECT-TYPE
    SYNTAX  Integer32 (0..100)
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "A value expressed as a percentage, interpreted by the meter
        as an indication of how full the flow table should be before
        it should take some action to avoid running out of resources
        to handle new flows, as discussed in section 4.6 (Handling
        Increasing Traffic Levels) of the RTFM Architecture RFC
        [RTFM-ARC].

        Values of 0% or 100% disable the checking represented by
        this variable."
    DEFVAL { 95 } -- Enabled by default.
    ::= { flowControl 5 }

flowInactivityTimeout OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "The time in seconds since the last packet seen, after which
        a flow becomes 'idle.'  Note that although a flow may be
        idle, it will not be discarded (and its memory recovered)
        until after its data has been collected by all the meter
        readers registered for its RuleSet."
    DEFVAL { 600 } -- 10 minutes
    ::= { flowControl 6 }

flowActiveFlows OBJECT-TYPE
    SYNTAX  Integer32
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The number of flows which are currently in use."
    ::= { flowControl 7 }

flowMaxFlows OBJECT-TYPE
```

```
    SYNTAX  Integer32
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The maximum number of flows allowed in the meter's
        flow table.  At present this is determined when the meter
        is first started up."
    ::= { flowControl 8 }

flowFloodMode OBJECT-TYPE
    SYNTAX  TruthValue
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "Indicates that the meter has passed its FloodMark and is
        not running in its normal mode.

        When the manager notices this it should take action to remedy
        the problem which caused the flooding.  It should then monitor
        flowActiveFlows so as to determine when the flood has receded.
        At that point the manager may set flowFloodMode to false(2) to
        resume normal operation."
    ::= { flowControl 9 }

--
-- The Flow Table
--

-- This is a table kept by a meter, with one flow data entry for every
-- flow being measured.  Each flow data entry stores the attribute
-- values for a traffic flow.  Details of flows and their attributes
-- are given in the 'Traffic Flow Measurement: Architecture'
-- document [RTFM-ARC].

-- From time to time a meter reader may sweep the flow table so as
-- to read counts.  This is most effectively achieved by using the
-- TimeMark variable together with successive GetBulk requests to
-- retrieve the values of the desired flow attribute variables.

-- This scheme allows multiple meter readers to independently use the
-- same meter; the meter readers do not have to be synchronised and
-- they may use different collection intervals.

-- If identical sets of counts are required from a meter, a manager
-- could achieve this using two identical copies of a RuleSet in that
-- meter and switching back and forth between them.  This is discussed
-- further in the RTFM Architecture document [RTFM-ARC].
```

```
flowDataTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowDataEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "The list of all flows being measured."
    ::= { flowData 1 }

flowDataEntry OBJECT-TYPE
    SYNTAX  FlowDataEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "The flow data record for a particular flow."
    INDEX { flowDataRuleSet, flowDataTimeMark, flowDataIndex }
    ::= { flowDataTable 1 }

FlowDataEntry ::= SEQUENCE {
    flowDataIndex                   Integer32,
    flowDataTimeMark                TimeFilter,
    flowDataStatus                  INTEGER,

    flowDataSourceInterface         Integer32,
    flowDataSourceAdjacentType      AdjacentType,
    flowDataSourceAdjacentAddress   AdjacentAddress,
    flowDataSourceAdjacentMask      AdjacentAddress,
    flowDataSourcePeerType          PeerType,
    flowDataSourcePeerAddress       PeerAddress,
    flowDataSourcePeerMask          PeerAddress,
    flowDataSourceTransType         TransportType,
    flowDataSourceTransAddress      TransportAddress,
    flowDataSourceTransMask         TransportAddress,

    flowDataDestInterface           Integer32,
    flowDataDestAdjacentType        AdjacentType,
    flowDataDestAdjacentAddress     AdjacentAddress,
    flowDataDestAdjacentMask        AdjacentAddress,
    flowDataDestPeerType            PeerType,
    flowDataDestPeerAddress         PeerAddress,
    flowDataDestPeerMask            PeerAddress,
    flowDataDestTransType           TransportType,
    flowDataDestTransAddress        TransportAddress,
    flowDataDestTransMask           TransportAddress,

    flowDataPDUScale                Integer32,
    flowDataOctetScale              Integer32,

    flowDataRuleSet                 Integer32,
```

```
    flowDataToOctets                Counter64,    -- Source->Dest
    flowDataToPDUs                  Counter64,
    flowDataFromOctets              Counter64,    -- Dest->Source
    flowDataFromPDUs                Counter64,
    flowDataFirstTime               TimeStamp,    -- Activity times
    flowDataLastActiveTime          TimeStamp,

    flowDataSourceSubscriberID      OCTET STRING,
    flowDataDestSubscriberID        OCTET STRING,
    flowDataSessionID               OCTET STRING,

    flowDataSourceClass             Integer32,
    flowDataDestClass               Integer32,
    flowDataClass                   Integer32,
    flowDataSourceKind              Integer32,
    flowDataDestKind                Integer32,
    flowDataKind                    Integer32
    }

flowDataIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Value of this flow data record's index within the meter's
        flow table."
    ::= { flowDataEntry 1 }

flowDataTimeMark OBJECT-TYPE
    SYNTAX  TimeFilter
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "A TimeFilter for this entry.  Allows GetNext and GetBulk
        to find flow table rows which have changed since a specified
        value of the meter's Uptime."
    ::= { flowDataEntry 2 }

flowDataStatus OBJECT-TYPE
    SYNTAX  INTEGER { inactive(1), current(2) }
    MAX-ACCESS  read-only
    STATUS  deprecated
    DESCRIPTION
        "Status of this flow data record."
    ::= { flowDataEntry 3 }

flowDataSourceInterface OBJECT-TYPE
    SYNTAX  Integer32
```

```
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Index of the interface associated with the source address
        for this flow.  It's value is one of those contained in the
        ifIndex field of the meter's interfaces table."
    ::= { flowDataEntry 4 }

flowDataSourceAdjacentType OBJECT-TYPE
    SYNTAX  AdjacentType
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Adjacent address type of the source for this flow.

        If metering is being performed at the network level,
        AdjacentType will indicate the medium for the interface on
        which the flow was observed and AdjacentAddress will be the
        MAC address for that interface.  This is the usual case.

        If traffic is being metered inside a tunnel, AdjacentType will
        be the peer type of the host at the end of the tunnel and
        AdjacentAddress will be the peer address for that host."
    ::= { flowDataEntry 5 }

flowDataSourceAdjacentAddress OBJECT-TYPE
    SYNTAX  AdjacentAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Address of the adjacent device on the path for the source
        for this flow."
    ::= { flowDataEntry 6 }

flowDataSourceAdjacentMask OBJECT-TYPE
    SYNTAX  AdjacentAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "1-bits in this mask indicate which bits must match when
        comparing the adjacent source address for this flow."
    ::= { flowDataEntry 7 }

flowDataSourcePeerType OBJECT-TYPE
    SYNTAX  PeerType
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
```

```
        "Peer address type of the source for this flow."
    ::= { flowDataEntry 8 }

flowDataSourcePeerAddress OBJECT-TYPE
    SYNTAX  PeerAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Address of the peer device for the source of this flow."
    ::= { flowDataEntry 9 }

flowDataSourcePeerMask OBJECT-TYPE
    SYNTAX  PeerAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "1-bits in this mask indicate which bits must match when
        comparing the source peer address for this flow."
    ::= { flowDataEntry 10 }

flowDataSourceTransType OBJECT-TYPE
    SYNTAX  TransportType
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Transport address type of the source for this flow.  The
        value of this attribute will depend on the peer address type."
    ::= { flowDataEntry 11 }

flowDataSourceTransAddress OBJECT-TYPE
    SYNTAX  TransportAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Transport address for the source of this flow."
    ::= { flowDataEntry 12 }

flowDataSourceTransMask OBJECT-TYPE
    SYNTAX  TransportAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "1-bits in this mask indicate which bits must match when
        comparing the transport source address for this flow."
    ::= { flowDataEntry 13 }

flowDataDestInterface OBJECT-TYPE
    SYNTAX  Integer32
```

```
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Index of the interface associated with the dest address for
        this flow.  This value is one of the values contained in the
        ifIndex field of the interfaces table."
    ::= { flowDataEntry 14 }

flowDataDestAdjacentType OBJECT-TYPE
    SYNTAX  AdjacentType
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Adjacent address type of the destination for this flow."
    ::= { flowDataEntry 15 }

flowDataDestAdjacentAddress OBJECT-TYPE
    SYNTAX  AdjacentAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Address of the adjacent device on the path for the
        destination for this flow."
    ::= { flowDataEntry 16 }

flowDataDestAdjacentMask OBJECT-TYPE
    SYNTAX  AdjacentAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "1-bits in this mask indicate which bits must match when
        comparing the adjacent destination address for this flow."
    ::= { flowDataEntry 17 }

flowDataDestPeerType OBJECT-TYPE
    SYNTAX  PeerType
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Peer address type of the destination for this flow."
    ::= { flowDataEntry 18 }

flowDataDestPeerAddress OBJECT-TYPE
    SYNTAX  PeerAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Address of the peer device for the destination of this flow."
```

```
    ::= { flowDataEntry 19 }

flowDataDestPeerMask OBJECT-TYPE
    SYNTAX  PeerAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "1-bits in this mask indicate which bits must match when
        comparing the destination peer type for this flow."
    ::= { flowDataEntry 20 }

flowDataDestTransType OBJECT-TYPE
    SYNTAX  TransportType
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Transport address type of the destination for this flow.  The
        value of this attribute will depend on the peer address type."
    ::= { flowDataEntry 21 }

flowDataDestTransAddress OBJECT-TYPE
    SYNTAX  TransportAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Transport address for the destination of this flow."
    ::= { flowDataEntry 22 }

flowDataDestTransMask OBJECT-TYPE
    SYNTAX  TransportAddress
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "1-bits in this mask indicate which bits must match when
        comparing the transport destination address for this flow."
    ::= { flowDataEntry 23 }

flowDataPDUScale OBJECT-TYPE
    SYNTAX  Integer32 (0..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The scale factor applied to this particular flow.  Indicates
        the number of bits the PDU counter values should be moved left
        to obtain the actual values."
    ::= { flowDataEntry 24 }

flowDataOctetScale OBJECT-TYPE
```

```
    SYNTAX  Integer32 (0..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The scale factor applied to this particular flow.  Indicates
        the number of bits the octet counter values should be moved
        left to obtain the actual values."
    ::= { flowDataEntry 25 }

flowDataRuleSet OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "The RuleSet number of the RuleSet which created this flow.
        Allows a manager to use GetNext or GetBulk requests to find
        flows belonging to a particular RuleSet."
    ::= { flowDataEntry 26 }

flowDataToOctets OBJECT-TYPE
    SYNTAX  Counter64
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The count of octets flowing from source to destination
        for this flow."
    ::= { flowDataEntry 27 }

flowDataToPDUs OBJECT-TYPE
    SYNTAX  Counter64
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The count of packets flowing from source to destination
        for this flow."
    ::= { flowDataEntry 28 }

flowDataFromOctets OBJECT-TYPE
    SYNTAX  Counter64
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The count of octets flowing from destination to source
        for this flow."
    ::= { flowDataEntry 29 }

flowDataFromPDUs OBJECT-TYPE
    SYNTAX  Counter64
```

```
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The count of packets flowing from destination to source
        for this flow."
    ::= { flowDataEntry 30 }

flowDataFirstTime OBJECT-TYPE
    SYNTAX  TimeStamp
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The time at which this flow was first entered in the table"
    ::= { flowDataEntry 31 }

flowDataLastActiveTime OBJECT-TYPE
    SYNTAX  TimeStamp
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The last time this flow had activity, i.e. the time of
        arrival of the most recent PDU belonging to this flow."
    ::= { flowDataEntry 32 }

flowDataSourceSubscriberID OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (4..20))
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Subscriber ID associated with the source address for this
        flow.  A Subscriber ID is an unspecified text string, used
        to ascribe traffic flows to individual users.  At this time
        the means by which a Subscriber ID may be associated with a
        flow is unspecified."
    ::= { flowDataEntry 33 }

flowDataDestSubscriberID OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (4..20))
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Subscriber ID associated with the destination address for
        this flow.  A Subscriber ID is an unspecified text string,
        used to ascribe traffic flows to individual users.  At this
        time the means by which a Subscriber ID may be associated
        with a flow is unspecified."
    ::= { flowDataEntry 34 }
```

```
flowDataSessionID OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (4..10))
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Session ID for this flow.  Such an ID might be allocated
        by a network access server to distinguish a series of sessions
        between the same pair of addresses, which would otherwise
        appear to be parts of the same accounting flow."
    ::= { flowDataEntry 35 }

flowDataSourceClass OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Source class for this flow.  Determined by the rules, set by
        a PushRule action when this flow was entered in the table."
    ::= { flowDataEntry 36 }

flowDataDestClass OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Destination class for this flow.  Determined by the rules, set
        by a PushRule action when this flow was entered in the table."
    ::= { flowDataEntry 37 }

flowDataClass OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Class for this flow.  Determined by the rules, set by a
        PushRule action when this flow was entered in the table."
    ::= { flowDataEntry 38 }

flowDataSourceKind OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Source kind for this flow.  Determined by the rules, set by
        a PushRule action when this flow was entered in the table."
    ::= { flowDataEntry 39 }

flowDataDestKind OBJECT-TYPE
```

```
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Destination kind for this flow.  Determined by the rules, set
        by a PushRule action when this flow was entered in the table."
    ::= { flowDataEntry 40 }

flowDataKind OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "Class for this flow.  Determined by the rules, set by a
        PushRule action when this flow was entered in the table."
    ::= { flowDataEntry 41 }

--
-- The Activity Column Table
--

flowColumnActivityTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowColumnActivityEntry
    MAX-ACCESS  not-accessible
    STATUS   deprecated
    DESCRIPTION
        "Index into the Flow Table.  Allows a meter reader to retrieve
        a list containing the flow table indexes of flows which were
        last active at or after a given time, together with the values
        of a specified attribute for each such flow."
    ::= { flowData 2 }

flowColumnActivityEntry OBJECT-TYPE
    SYNTAX  FlowColumnActivityEntry
    MAX-ACCESS  not-accessible
    STATUS  deprecated
    DESCRIPTION
        "The Column Activity Entry for a particular attribute,
        activity time and flow."
    INDEX { flowColumnActivityAttribute, flowColumnActivityTime,
            flowColumnActivityIndex }
    ::= { flowColumnActivityTable 1 }

FlowColumnActivityEntry ::= SEQUENCE {
    flowColumnActivityAttribute   FlowAttributeNumber,
    flowColumnActivityTime        TimeFilter,
    flowColumnActivityIndex       Integer32,
    flowColumnActivityData        OCTET STRING
```

```
    }

flowColumnActivityAttribute OBJECT-TYPE
    SYNTAX  FlowAttributeNumber
    MAX-ACCESS  read-only
    STATUS  deprecated
    DESCRIPTION
        "Specifies the attribute for which values are required from
        active flows."
    ::= { flowColumnActivityEntry 1 }

flowColumnActivityTime OBJECT-TYPE
    SYNTAX  TimeFilter
    MAX-ACCESS  read-only
    STATUS  deprecated
    DESCRIPTION
        "This variable is a copy of flowDataLastActiveTime in the
        flow data record identified by the flowColumnActivityIndex
        value of this flowColumnActivityTable entry."
    ::= { flowColumnActivityEntry 2 }

flowColumnActivityIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  read-only
    STATUS  deprecated
    DESCRIPTION
        "Index of a flow table entry which was active at or after
        a specified flowColumnActivityTime."
    ::= { flowColumnActivityEntry 3 }

flowColumnActivityData OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (3..1000))
    MAX-ACCESS  read-only
    STATUS  deprecated
    DESCRIPTION
        "Collection of attribute data for flows active after
        flowColumnActivityTime.  Within the OCTET STRING is a
        sequence of { flow index, attribute value } pairs, one for
        each active flow.  The end of the sequence is marked by a
        flow index value of 0, indicating that there are no more
        rows in this column.

        The format of objects inside flowColumnFlowData is as follows.
        All numbers are unsigned.  Numbers and strings appear with
        their high-order bytes leading.  Numbers are fixed size, as
        specified by their SYNTAX in the flow table (above), i.e. one
        octet for flowAddressType and small constants, and four octets
        for Counter and TimeStamp.  Strings are variable-length, with
```

```
        the length given in a single leading octet.

        The following is an attempt at an ASN.1 definition of
        flowColumnActivityData:

        flowColumnActivityData ::= SEQUENCE flowRowItemEntry
        flowRowItemEntry ::= SEQUENCE {
           flowRowNumber    Integer32 (1..65535),
                                -- 0 indicates the end of this column
           flowDataValue   flowDataType -- Choice depends on attribute
           }
        flowDataType ::= CHOICE {
            flowByteValue   Integer32 (1..255),
            flowShortValue  Integer32 (1..65535),
            flowLongValue   Integer32,
            flowStringValue OCTET STRING  -- Length (n) in first byte,
                  -- n+1 bytes total length, trailing zeroes truncated
            }"
    ::= { flowColumnActivityEntry 4 }

--
-- The Data Package Table
--

flowDataPackageTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowDataPackageEntry
    MAX-ACCESS  not-accessible
    STATUS   current
    DESCRIPTION
        "Index into the Flow Table.  Allows a meter reader to retrieve
        a sequence containing the values of a specified set of
        attributes for a flow which came from a specified RuleSet and
        which was last active at or after a given time."
    ::= { flowData 3 }

flowDataPackageEntry OBJECT-TYPE
    SYNTAX  FlowDataPackageEntry
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "The data package containing selected variables from
        active rows in the flow table."
    INDEX { flowPackageSelector,
        flowPackageRuleSet, flowPackageTime, flowPackageIndex }
    ::= { flowDataPackageTable 1 }

FlowDataPackageEntry ::= SEQUENCE {
    flowPackageSelector    OCTET STRING,
```

```
    flowPackageRuleSet     Integer32,
    flowPackageTime        TimeFilter,
    flowPackageIndex       Integer32,
    flowPackageData        OCTET STRING
    }

flowPackageSelector OBJECT-TYPE
    SYNTAX  OCTET STRING
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Specifies the attributes for which values are required from
        an active flow.  These are encoded as a sequence of octets
        each containing a FlowAttribute number, preceded by an octet
        giving the length of the sequence (not including the length
        octet).  For a flowPackageSelector to be valid, it must
        contain at least one attribute."
    ::= { flowDataPackageEntry 1 }

flowPackageRuleSet OBJECT-TYPE
    SYNTAX  Integer32 (1..255)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Specifies the index (in the flowRuleSetInfoTable) of the rule
        set which produced the required flow."
    ::= { flowDataPackageEntry 2 }

flowPackageTime OBJECT-TYPE
    SYNTAX  TimeFilter
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "This variable is a copy of flowDataLastActiveTime in the
        flow data record identified by the flowPackageIndex
        value of this flowPackageTable entry."
    ::= { flowDataPackageEntry 3 }

flowPackageIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Index of a flow table entry which was active at or after
        a specified flowPackageTime."
    ::= { flowDataPackageEntry 4 }

flowPackageData OBJECT-TYPE
```

```
    SYNTAX  OCTET STRING
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "A collection of attribute values for a single flow, as
        specified by this row's indexes.  The attribute values are
        contained within a BER-encoded sequence [ASN-1, ASN-BER],
        in the order they appear in their flowPackageSelector.

        For example, to retrieve a flowPackage containing values for
        attributes 11, 18 and 29, for a flow in RuleSet 7, with flow
        index 3447, one would GET the package whose Object Identifier
        (OID) is
            flowPackageData . 3.11.18.29 . 7. 0 . 3447

        To get a package for the next such flow which had been
        active since time 12345 one would GETNEXT the package whose
        Object Identifier (OID) is
            flowPackageData . 3.11.18.29 . 7. 12345 . 3447"
    ::= { flowDataPackageEntry 5 }

--
-- The Rule Table
--

-- This is an array of RuleSets; the 'running' ones are indicated
-- by the entries in the meter's flowManagerInfoTable.  Several
-- RuleSets can be held in a meter so that the manager can change the
-- running RuleSets easily, for example with time of day.  Note that
-- a manager may not change the rules in any RuleSet currently
-- referenced within the flowManagerInfoTable (either as 'current' or
-- 'standby')!  See the 'Traffic Flow Measurement: Architecture'
-- document [RTFM-ARC] for details of rules and how they are used.

-- Space for a RuleSet is allocated by setting the value of
-- flowRuleInfoSize in the rule table's flowRuleSetInfoTable row.
-- Values for each row in the RuleSet (Selector, Mask, MatchedValue,
-- Action and Parameter) can then be set by the meter.

-- Although an individual rule within a RuleSet could be modified,
-- it is much safer to simply download a complete new RuleSet.

flowRuleTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF FlowRuleEntry
    MAX-ACCESS  not-accessible
    STATUS   current
    DESCRIPTION
        "Contains all the RuleSets which may be used by the meter."
```

```
    ::= { flowRules 1 }

flowRuleEntry OBJECT-TYPE
    SYNTAX  FlowRuleEntry
    MAX-ACCESS  not-accessible
    STATUS   current
    DESCRIPTION
        "The rule record itself."
    INDEX { flowRuleSet, flowRuleIndex }
    ::= { flowRuleTable 1 }

FlowRuleEntry ::= SEQUENCE {
    flowRuleSet                 Integer32,
    flowRuleIndex               Integer32,
    flowRuleSelector            RuleAttributeNumber,
    flowRuleMask                RuleAddress,
    flowRuleMatchedValue        RuleAddress,
    flowRuleAction              ActionNumber,
    flowRuleParameter           Integer32
    }

flowRuleSet OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "Selects a RuleSet from the array of RuleSets."
    ::= { flowRuleEntry 1 }

flowRuleIndex OBJECT-TYPE
    SYNTAX  Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS  current
    DESCRIPTION
        "The index into the Rule table.  N.B: These values will
        normally be consecutive, given the fall-through semantics
        of processing the table."
    ::= { flowRuleEntry 2 }

flowRuleSelector OBJECT-TYPE
    SYNTAX  RuleAttributeNumber
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "Indicates the attribute to be matched.

        null(0) is a special case; null rules always succeed.
```

```
        matchingStoD(50) is set by the meter's Packet Matching Engine.
        Its value is true(1) if the PME is attempting to match the
        packet with its addresses in Source-to-Destination order (i.e.
        as they appear in the packet), and false(2) otherwise.
        Details of how packets are matched are given in the 'Traffic
        Flow Measurement: Architecture' document [RTFM-ARC].
        v1(51), v2(52), v3(53), v4(54) and v5(55) select meter
        variables, each of which can hold the name (i.e. selector
        value) of an address attribute.  When one of these is used
        as a selector, its value specifies the attribute to be
        tested.  Variable values are set by an Assign action."
    ::= { flowRuleEntry 3 }

flowRuleMask OBJECT-TYPE
    SYNTAX  RuleAddress
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "The initial mask used to compute the desired value.  If the
        mask is zero the rule's test will always succeed."
    ::= { flowRuleEntry 4 }

flowRuleMatchedValue OBJECT-TYPE
    SYNTAX  RuleAddress
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "The resulting value to be matched for equality.
        Specifically, if the attribute chosen by the flowRuleSelector
        logically ANDed with the mask specified by the flowRuleMask
        equals the value specified in the flowRuleMatchedValue, then
        continue processing the table entry based on the action
        specified by the flowRuleAction entry.  Otherwise, proceed to
        the next entry in the rule table."
    ::= { flowRuleEntry 5 }

flowRuleAction OBJECT-TYPE
    SYNTAX  ActionNumber
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "The action to be taken if this rule's test succeeds, or if
        the meter's 'test' flag is off.  Actions are opcodes for the
        meter's Packet Matching Engine; details are given in the
        'Traffic Flow Measurement: Architecture' document [RTFM-ARC]."
    ::= { flowRuleEntry 6 }

flowRuleParameter OBJECT-TYPE
```

```
    SYNTAX  Integer32 (1..65535)
    MAX-ACCESS  read-write
    STATUS  current
    DESCRIPTION
        "A parameter value providing extra information for this rule's
        action.  Most of the actions use the parameter value to specify
        which rule to execute after this rule's test has failed; details
        are given in the 'Traffic Flow Measurement: Architecture'
        document [RTFM-ARC]."
    ::= { flowRuleEntry 7 }

--
-- Traffic Flow Meter conformance statement
--

flowMIBCompliances
    OBJECT IDENTIFIER ::= { flowMIBConformance 1 }

flowMIBGroups
    OBJECT IDENTIFIER ::= { flowMIBConformance 2 }

flowControlGroup OBJECT-GROUP
    OBJECTS  {
        flowRuleInfoSize, flowRuleInfoOwner,
            flowRuleInfoTimeStamp, flowRuleInfoStatus,
            flowRuleInfoName,
            flowRuleInfoRulesReady,
            flowRuleInfoFlowRecords,
        flowInterfaceSampleRate,
            flowInterfaceLostPackets,
        flowReaderTimeout, flowReaderOwner,
            flowReaderLastTime, flowReaderPreviousTime,
            flowReaderStatus, flowReaderRuleSet,
        flowManagerCurrentRuleSet, flowManagerStandbyRuleSet,
            flowManagerHighWaterMark,
            flowManagerCounterWrap,
            flowManagerOwner, flowManagerTimeStamp,
            flowManagerStatus, flowManagerRunningStandby,
        flowFloodMark,
            flowInactivityTimeout, flowActiveFlows,
            flowMaxFlows, flowFloodMode }
    STATUS  deprecated
    DESCRIPTION
        "The control group defines objects which are used to control
        an accounting meter."
    ::= {flowMIBGroups 1 }

flowDataTableGroup OBJECT-GROUP
```

```
    OBJECTS  {
--      flowDataIndex,                    <- INDEX, not-accessible
        flowDataStatus,
        flowDataSourceInterface,
        flowDataSourceAdjacentType,
        flowDataSourceAdjacentAddress, flowDataSourceAdjacentMask,
        flowDataSourcePeerType,
        flowDataSourcePeerAddress, flowDataSourcePeerMask,
        flowDataSourceTransType,
        flowDataSourceTransAddress, flowDataSourceTransMask,
        flowDataDestInterface,
        flowDataDestAdjacentType,
        flowDataDestAdjacentAddress, flowDataDestAdjacentMask,
        flowDataDestPeerType,
        flowDataDestPeerAddress, flowDataDestPeerMask,
        flowDataDestTransType,
        flowDataDestTransAddress, flowDataDestTransMask,
--      flowDataRuleSet,                  <- INDEX, not-accessible
        flowDataToOctets, flowDataToPDUs,
        flowDataFromOctets, flowDataFromPDUs,
        flowDataFirstTime, flowDataLastActiveTime,
        flowDataSourceClass, flowDataDestClass, flowDataClass,
        flowDataSourceKind, flowDataDestKind, flowDataKind
        }
    STATUS  deprecated
    DESCRIPTION
        "The flow table group defines objects which provide the
        structure for the flow table, including the creation time
        and activity time indexes into it.  In addition it defines
        objects which provide a base set of flow attributes for the
        adjacent, peer and transport layers, together with a flow's
        counters and times.  Finally it defines a flow's class and
        kind attributes, which are set by rule actions."
    ::= {flowMIBGroups 2 }

flowDataScaleGroup OBJECT-GROUP
    OBJECTS  {
        flowManagerCounterWrap,
        flowDataPDUScale, flowDataOctetScale
        }
    STATUS  deprecated
    DESCRIPTION
        "The flow scale group defines objects which specify scale
        factors for counters."
    ::= {flowMIBGroups 3 }

flowDataSubscriberGroup OBJECT-GROUP
    OBJECTS  {
```

```
        flowDataSourceSubscriberID, flowDataDestSubscriberID,
        flowDataSessionID
        }
    STATUS  current
    DESCRIPTION
        "The flow subscriber group defines objects which may be used
        to identify the end point(s) of a flow."
    ::= {flowMIBGroups 4 }

flowDataColumnTableGroup OBJECT-GROUP
    OBJECTS  {
        flowColumnActivityAttribute,
        flowColumnActivityIndex,
        flowColumnActivityTime,
        flowColumnActivityData
        }
    STATUS  deprecated
    DESCRIPTION
        "The flow column table group defines objects which can be used
        to collect part of a column of attribute values from the flow
        table."
    ::= {flowMIBGroups 5 }

flowDataPackageGroup OBJECT-GROUP
    OBJECTS  {
        flowPackageData
        }
    STATUS  current
    DESCRIPTION
        "The data package group defines objects which can be used
        to collect a specified set of attribute values from a row of
        the flow table."
    ::= {flowMIBGroups 6 }

flowRuleTableGroup OBJECT-GROUP
    OBJECTS  {
        flowRuleSelector,
        flowRuleMask, flowRuleMatchedValue,
        flowRuleAction, flowRuleParameter
        }
    STATUS  current
    DESCRIPTION
        "The rule table group defines objects which hold the set(s)
        of rules specifying which traffic flows are to be accounted
        for."
    ::= {flowMIBGroups 7 }

flowDataScaleGroup2 OBJECT-GROUP
```

```
    OBJECTS  {
--      flowManagerCounterWrap,           <- Deprecated
        flowDataPDUScale, flowDataOctetScale
        }
    STATUS  current
    DESCRIPTION
        "The flow scale group defines objects which specify scale
        factors for counters.  This group replaces the earlier
        version of flowDataScaleGroup above (now deprecated)."
    ::= {flowMIBGroups 8}

flowControlGroup2 OBJECT-GROUP
    OBJECTS  {
        flowRuleInfoSize, flowRuleInfoOwner,
            flowRuleInfoTimeStamp, flowRuleInfoStatus,
            flowRuleInfoName,
--          flowRuleInfoRulesReady,       <- Deprecated
            flowRuleInfoFlowRecords,
        flowInterfaceSampleRate,
            flowInterfaceLostPackets,
        flowReaderTimeout, flowReaderOwner,
            flowReaderLastTime, flowReaderPreviousTime,
            flowReaderStatus, flowReaderRuleSet,
        flowManagerCurrentRuleSet, flowManagerStandbyRuleSet,
            flowManagerHighWaterMark,
--          flowManagerCounterWrap,       <- Moved to DataScaleGroup
            flowManagerOwner, flowManagerTimeStamp,
            flowManagerStatus, flowManagerRunningStandby,
        flowFloodMark,
            flowInactivityTimeout, flowActiveFlows,
            flowMaxFlows, flowFloodMode }
    STATUS  current
    DESCRIPTION
        "The control group defines objects which are used to control
        an accounting meter.  It replaces the earlier version of
        flowControlGroup above (now deprecated)."
    ::= {flowMIBGroups 9 }

flowMIBCompliance MODULE-COMPLIANCE
    STATUS  current
    DESCRIPTION
        "The compliance statement for a Traffic Flow Meter."
    MODULE
        MANDATORY-GROUPS  {
            flowControlGroup2,
            flowDataTableGroup,
            flowDataPackageGroup,
            flowRuleTableGroup
```

```
            }
    ::= { flowMIBCompliances 1 }

END
```

5 Security Considerations

5.1 SNMP Concerns

There are a number of management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

There are a number of managed objects in this MIB that may contain sensitive information. These include all the objects in the Control Group (since they control access to meter resources by Managers and Meter Readers) and those in the Flow Table (since they hold the collected traffic flow data).

It is thus important to control even GET access to these objects and possibly to even encrypt the values of these object when sending them over the network via SNMP. Not all versions of SNMP provide features for such a secure environment.

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 [RFC2574] and the View-based Access Control Model [RFC2575] 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.

5.2 Traffic Meter Concerns

This MIB describes how an RTFM traffic meter is controlled, and provides a way for traffic flow data to be retrieved from it by a meter reader. This is essentially an application using SNMP as a method of communication between co-operating hosts; it does not - in itself - have any inherent security risks.

Since, however, the traffic flow data can be extremely valuable for network management purposes it is vital that sensible precautions be taken to keep the meter and its data secure. In particular, an attacker must not be permitted to write any of the meter's variables! This requires that access to the meter for control purposes (e.g. loading RuleSets and reading flow data) be restricted. Such restriction could be achieved in many ways, for example:

- Physical Separation. Meter(s) and meter reader(s) could be deployed so that control capabilities are kept within a separate network, access to which is carefully controlled.

- Application-layer Security. A minimal level of security for SNMP can be provided by using 'community' strings (which are essentially clear-text passwords) with SNMPv2C [RFC1157]. Where stronger security is needed, users should consider using the User-based Security Model [RFC2574] and the View-based Access Control Model [RFC2575].

- Lower-layer Security. Access to the meter can be protected using encryption at the network layer. For example, one could run SNMP to the meter through an encrypted TCP tunnel.

When implementing a meter it may be sensible to use separate network interfaces for control and for metering. If this is done the control network can be set up so that it doesn't carry any 'user' traffic, and the metering interfaces can ignore any user attempts to take control of the meter.

Users should also consider how they will address attempts to circumvent a meter, i.e. to prevent it from measuring flows. Such attempts are essentially denial-of-service attacks on the metering interfaces. For example

- Port Scan attacks. The attacker sends packets to each of a very large number of IP (Address : Port) pairs. Each of these packets creates a new flow in the meter; if there are enough of them the meter will recognise a 'flood' condition, and will probably stop creating new flows. As a minimum, users (and implementors) should ensure that meters can recover from flood conditions as soon as possible after they occur.

- Counter Wrap attacks: The attacker sends enough packets to cause the counters in a flow to wrap several times between meter readings, thus causing the counts to be artificially low. The change to using 64-bit counters in this MIB reduces this problem significantly.

Users can reduce the severity of both the above attacks by ensuring that their meters are read often enough to prevent them being flooded. The resulting flow data will contain a record of the attacking packets, which may well be useful in determining where any attack came from.

6 IANA Considerations

The RTFM Architecture document [RTFM-ARC], has two sets of assigned numbers: Opcodes for the PME (Pattern Matching Engine) and RTFM Attribute numbers. All the assigned numbers used in the Meter MIB appear in Textual Conventions. The numbers they use are derived as follows:

The MIB's 'Type' textual conventions use names and numbers from the Assigned Numbers RFC [ASG-NBR]:

MediumType	Uses ifType Definitions
PeerType	Uses Address Family Numbers
TransportType	Uses Protocol Numbers

The MIB's 'AttributeNumber' textual conventions use RTFM Attribute names and numbers from the RTFM Architecture document [RTFM-ARC], or other numbers allocated according to that document's IANA Considerations section:

FlowAttributeNumber	Have values stored in a flow table row
RuleAttributeNumber	May be tested in a rule

The MIB's ActionNumber textual convention uses RTFM PME Opcode names and numbers from the RTFM Architecture document [RTFM-ARC], or other numbers allocated according to that document's IANA Considerations section.

7 Appendix A: Changes Introduced Since RFC 2064

The first version of the Meter MIB was published as RFC 2064 in January 1997. The most significant changes since then are summarised below.

- TEXTUAL CONVENTIONS: Greater use is made of textual conventions to describe the various types of addresses used by the meter.

- PACKET MATCHING ATTRIBUTES: Computed attributes (e.g. FlowClass and FlowKind) may now be tested. This allows one to use these variables to store information during packet matching.

  A new attribute, MatchingStoD, has been added. Its value is 1 while a packet is being matched with its adresses in 'wire' (source-to-destination) order.

- FLOOD MODE: This is now a read-write variable. Setting it to false(2) switches the meter out of flood mode and back to normal operation.

- CONTROL TABLES: Several variables have been added to the RuleSet, Reader and Manager tables to provide more effective control of the meter's activities.

- FLOW TABLE: 64-bit counters are used for octet and PDU counts. This reduces the problems caused by the wrap-around of 32-bit counters in earlier versions.
  flowDataRuleSet is now used as an index to the flow table. This allows a meter reader to collect only those flow table rows created by a specified RuleSet.

- DATA PACKAGES: This is a new table, allowing a meter reader to retrieve values for a list of attributes from a flow as a single object (a BER-encoded sequence [ASN-1, ASN-BER]). It provides an efficient way to recover flow data, particularly when used with SNMP GetBulk requests.

  Earlier versions had a 'Column Activity Table'; using this it was difficult to collect all data for a flow efficiently in a single SNMP request.

8 Acknowledgements

An early draft of this document was produced under the auspices of the IETF's Accounting Working Group with assistance from the SNMP Working Group and the Security Area Advisory Group. Particular thanks are due to Jim Barnes, Sig Handelman and Stephen Stibler for their support and their assistance with checking early versions of the MIB.

Stephen Stibler shared the development workload of producing the MIB changes summarized in chapter 5 (above).

9 Intellectual Property Notice

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 implementers 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 References

[ACT-BKG] Mills, C., Hirsch, G. and G. Ruth, "Internet Accounting Background", RFC 1272, November 1991.

[ASG-NBR] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, ISI, October 1994.

[ASN-1] Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), International Organization for Standardization, International Standard 8824, December 1987.

[ASN-BER] Information processing systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Notation One (ASN.1), International Organization for Standardization, International Standard 8825, December 1987.

[ENET-OBJ] Kastenholz, F., "Definitions of Managed Objects for the Ethernet-like Interface Types", RFC 1643, July 1994.

[FDDI-MIB] Case, J. and A. Rijsinghani, "FDDI Management Information Base", RFC 1512, September 1993.

[IPPM-FRM] Paxson, V., Almes, G., Mahdavi, J. and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.

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

[RFC1155] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990

[RFC1157] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.

[RFC1212] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[RFC1215] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991

[RFC1901] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.

[RFC1905] 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.

[RFC1906] 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.

[RFC1908] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Coexistence between version 1 and version 2 of the Internet-standard Network Management Framework", RFC 1908, January 1996.

[RFC2570] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.

[RFC2571] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.

[RFC2572] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.

[RFC2573] Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.

[RFC2574] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.

[RFC2575] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.

[RFC2578] 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.

[RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RMON-MIB] Waldbusser, S., "Remote Network Monitoring Management Information Base", RFC 1757, February 1995.

[RMON2-MIB] Waldbusser, S., "Remote Network Monitoring Management Information Base Version 2 using SMIv2", RFC 2021, January 1997.

[RTFM-ARC] Brownlee, N., Mills, C. and Ruth, G., "Traffic Flow Measurement: Architecture", RFC 722, October 1999.

[UTF-8] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.

[V6-ADDR] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

11 Author's Address

Nevil Brownlee
Information Technology Systems & Services
The University of Auckland
Private Bag 92-019
Auckland, New Zealand

Phone: +64 9 373 7599 x8941
EMail: n.brownlee@auckland.ac.nz

12 Full Copyright Statement

Copyright (C) The Internet Society (1999). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

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