INTERNET-DRAFT
draft-ietf-ppvpn-vr-mib-05.txt
Expires: December 2003

Elwin Stelzer
Sam Hancock
Corona Networks, Inc.

Benson Schliesser
SAVVIS Communications

Joseph Laria
Level Stream Research

June 2003

Virtual Router Management Information Base Using SMIv2

1.0 Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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

The list of current Internet-Drafts can be accessed at:
http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at:
http://www.ietf.org/shadow.html.

2.0 Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP based internets. In particular, it defines objects for managing networks using Virtual Routers (VR).

3.0 Table of Contents

4.0 Terminology

This document uses terminology defined in [PPVPN-FW] and [PPVPN-VR].

5.0 Introduction

This memo defines a MIB for the Virtual Router [PPVPN-VR, PPVPN-VR-AS] model of Provider Provisioned VPNs [PPVPN-FW].

Following are the goals, in defining this MIB:

- To have a means for Service Providers to provision VPN service for subscribers, at the PE device.

- To make the agent-side implementation simple, by not modifying the existing standard MIBs.

- Define all the gluing tables that are needed toward this end.

6.0 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 [22].

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.

7.0 Overview of the Virtual Router MIB

7.1 SNMP Contexts for Management for Virtual Routers

There is a need for a single agent to manage multiple Virtual Routers. The Architecture for describing Internet Management Frameworks [1] provides a way to support such cases.

Managing multiple virtual routers requires that the PE management plane be subdivided into logical VR management domains. In the VR model of PPVPNs a single PE device may contain many virtual routers. Different management entities SHOULD be able to manage specific virtual routers and associated services. The Service Provider MUST be able to manage all virtual routers and associated services.

Using SNMP contexts to group a collection of management information provides the following benefits:

(1) Uses a standard framework defined by the IETF, allowing the product to remain flexible to all implementations of virtual router devices.

(a) Use SNMPv2c Community String's

(b) Use SNMPv3 contextName's

(2) Prevents vendors from having to modify the standard MIBs, allowing the implementation to remain standards compliant.

(3) Provides a framework that will work for RIP, OSPF, IS-IS, BGP, IP-FORWARDING, MPLS, and any other MIB that can be administratively grouped with a VR.

The SNMP context for the Virtual Router instance can be specified in the VrConfigTable. The VrContextName columnar object is used to set the SNMPv2c Community String or the SNMPv3 contextName for a given VR.

A virtual router context represents the set of MIB objects that could be administratively grouped within a VR. For example, each VR would maintain its own instance of routing protocol MIB tables. However, the ADMIN context would contain single instances of objects and tables that pertain to system wide configuration such as the Entity, Interfaces, and ATM MIBs.

A management system using the SNMP context of a particular virtual router MUST be able to manage the virtual router without disrupting other virtual routers in the same PE device.

For example, a PE can be subdivided into two 2 VRs running the OSPF routing protocol. Each VR will maintain a unique instance of the OSPF-MIB. Therefore, the ospfAreaTable of VR-A is distinct from the ospfAreaTable of VR-B.

Filtering mechanisms based on the SNMP context of a particular virtual router may implemented to allow different management entities to manage those objects and services provisioned the ôAdminö context.

7.2 VR Indexing

While the standard protocol MIB tables are instantiated in the context specified using SNMP contexts, there may be tables that are defined with the VRID as index.

The VRID is of local significance to a particular PE device, and need not be globally unique. Thus a particular VRID value assigned to a VR in one PE device may indicate a different VR in another PE device.

The VRID has an Unsigned32 value, and this value is assigned by the management station. To aid the management station in assigning a VRID without conflict, the management station can get the 'NextAvailableVRID' from the PE device. A SNMP manager SHOULD NOT assume global significance of any VRID value other than 0.

For those MIB tables instantiated in the virtual router context, indexing can only be assumed unique for that particular VR. However those indices in the "ADMIN" context are unique across the entire system, including all VRs.

7.3 Creation and Deletion of VRs

The VR Config Table is used for this purpose. This is a read-create table and adding an entry into this table will create a VR. Removing an entry from this table marks the deletion of a VR.

VRID 0 is assigned to the Administrative VR, which exists by default, and need not be created. Deletion of the Administrative VR will not be permitted. The VRID of the Administrative VR (VRID 0) should be a reserved VRID number. VRID 0 could be termed the "null VR" and it could be the context that manages the resource pool of unattached interfaces. Routing would then not exist in the context of Administrative VR.

7.4 Administrative and Operational Status of VRs

VRs can be administratively turned down. When this is done, no packet forwarding via the VR takes place.

VrOperStatus denotes the operational status of a VR. Currently the VrOperStatus is expected to change along the VrAdminStatus unless an error condition exists.

7.5 Binding interfaces to a VR

Interfaces are bound to a VR, using vrIfConfigTable. This is a read-write table, and note that interfaces are not created through this table. The vrIfConfigTable MIB table is used to indicated the relationship between interfaces and virtual router IDs. For each interface present in the system, this table is used to provide the mapping from IfIndex to a unique VR. The table show which interfaces are ôattached or connectedö to a virtual router. An interface can not be attached to more than one VR.

The "Admin" VR could be used to manage the resource pool of unattached interfaces. However interfaces would not be attached to VRID 0.

7.6 Setting per VR limits

VRs consume resources on a device, and hence the following parameters defined in vrConfigTable are used to specify an upper bound of resource utilization:

VrMaxRoutes -
Specify the maximum number of routes that will be permitted in VR. This includes all routes, such as the statically configured routes, and the routes learnt via dynamic routing protocols.

7.7 Per VR Statistics

In addition to those statistics available through the VR instantiated MIB tables, there are some per-VR statistics available through vrStatTable.

7.8 Traps

This memo defines that VrUp and VrDown traps are generated just after VrOperStatus leaves, or just before it enters, the down state, respectively.

(1) A transition into the down state will occur when an error is detected on a VR instance.

(2) Departing the down state generally indicates that the VR is going to up, which is considered a "healthy" state.

An exception to the above generation of VrUp/VrDown traps on changes in VrOperStatus, occurs when an VR is "flapping", i.e., when it is rapidly oscillating between the up and down states. If traps were generated for each such oscillation, the network and the network management system would be flooded with unnecessary traps. In such a situation, the agent should limit the rate at which it generates traps.

This memo defines that enabling and disabling the VR traps is achieved by setting the VrTrapEnable to true(1) or false(2), respectively. By default, this object should have the value true(1).

8.0 Sample VR MIB Configuration Scenario

8.1 Creation of a VR

Creating VR instances can be achieved using the following example.

(1) Get the next available Virtual Router Id using the NextAvailableVrId, to create a VR:

```
    Using a context with 'read' access for system level entities.
    GetRequest { NextAvailableVrId.0 }
    Response   { NextAvailableVrId.0  =  5555 }
```

(2) In VrConfigTable, create VR Instance using VrRowStatus:

```
    Using a context with 'read-write' access for system level entities
    SetRequest {
        VrRowStatus.5555                createAndGo(4),
        VrName.5555                     "BigTelcoVR",
        VrContextName.5555              "vr5555",
        VrTrapEnable.5555               true(1),
        VrAdminStatus.5555              up(1)
    }
```

9.0 Definition of the Virtual Router MIB

```
--
-- VIRTUAL-ROUTER-MIB
--

VIRTUAL-ROUTER-MIB DEFINITIONS ::= BEGIN

    IMPORTS
        InterfaceIndex
            FROM IF-MIB
        InetAddressType, InetAddress
            FROM INET-ADDRESS-MIB
        OBJECT-GROUP, MODULE-COMPLIANCE, NOTIFICATION-GROUP
            FROM SNMPv2-CONF
        experimental, Unsigned32, OBJECT-TYPE,
        MODULE-IDENTITY, TimeTicks, NOTIFICATION-TYPE
            FROM SNMPv2-SMI
        TruthValue, DisplayString, RowStatus, TEXTUAL-CONVENTION
            FROM SNMPv2-TC;

    virtualRouterMIB MODULE-IDENTITY
        LAST-UPDATED "200301311200Z"
        ORGANIZATION
            "IETF PPVPN WG"
        CONTACT-INFO
            "
            Elwin Stelzer Eliazer
            Corona Networks, Inc.
            630 Alder Drive
            Milpitas, CA 95035
            USA
            Phone: +1-408-519-3832
            Email: elwinietf@yahoo.com
```

```
            Samuel Hancock
            Corona Networks, Inc.
            630 Alder Drive
            Milpitas, CA 95035
            USA
            Phone: +1-408-519-3800 Ext 421
            Email: hancoc_s@yahoo.com

            Benson Schliesser
            SAVVIS Communications
            1 Savvis Parkway
            Town and Country, MO 63017
            USA
            Phone: +1-314-628-7036
            Email: bensons@savvis.net

            Joseph Laria
            Level Stream Research
            Wilmington, MA  01887
            USA
            Phone: +1-978-884-3537
            Emain: jlaria@levelstream.com
            "
        DESCRIPTION
            "The MIB is the definition of the managed
            objects for the Virtual Router."
        REVISION "200306011200Z"
        DESCRIPTION
            "VR-MIB Draft of the IETF PPVPN WG"
        ::= { experimental xxxx } -- To be assigned

--
-- Textual conventions
--

    VrId ::= TEXTUAL-CONVENTION
        STATUS current
        DESCRIPTION
            "Virtual Router Identifier.
             VRID 0 is reserved for the Administrative VR
             and cannot be used to create VR's.
            "
        SYNTAX Unsigned32

    VpnIdentifier ::= TEXTUAL-CONVENTION
        STATUS current
        DESCRIPTION
            "RFC2685:  The global VPN Identifier format is:
            3 octet VPN authority Organizationally Unique Identifier
            followed by
            4 octet VPN index identifying VPN according to OUI"
```

```
        SYNTAX OCTET STRING(SIZE (0..7))

--
-- Node definitions
--

    vrMIBObjects OBJECT IDENTIFIER ::= { virtualRouterMIB 1 }

    vrConfig OBJECT IDENTIFIER ::= { vrMIBObjects 1 }

    vrConfigScalars OBJECT IDENTIFIER ::= { vrConfig 1 }

    vrConfigNextAvailableVrId OBJECT-TYPE
        SYNTAX VrId
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "The next available Virtual Router Id (index).
            This object provides a hint for the vrID value
            to use when administratively creating a new
            vrConfigEntry.

            A GET of this object returns the next available vrId
            value to be used to create an entry in the associated
            vrConfigTable; or zero, if no valid vrId
            value is available. A value of zero(0) indicates that
            it is not possible to create a new vrConfigEntry
            This object also returns a value of zero when it is the
            lexicographic successor of a varbind presented in an
            SNMP GETNEXT or GETBULK request, for which circumstance
            it is assumed that ifIndex allocation is unintended.

            Successive GETs will typically return different
            values, thus avoiding collisions among cooperating
            management clients seeking to create table entries
            simultaneously.

            Unless specified otherwise by its MAX-ACCESS and
            DESCRIPTION clauses, an object of this type is read-only,
            and a SET of such an object returns a notWritable error."
        ::= { vrConfigScalars 1 }

    vrConfigTable OBJECT-TYPE
        SYNTAX SEQUENCE OF VrConfigEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
            "This table is for creating the new Virtual Routers."
        ::= { vrConfig 2 }

    vrConfigEntry OBJECT-TYPE
        SYNTAX VrConfigEntry
```

```
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The entries in this table can be added/deleted
        using the vrRowStatus."
    INDEX { vrId }
    ::= { vrConfigTable 1 }

VrConfigEntry ::=
    SEQUENCE {
        vrId
            VrId,
        vrRowStatus
            RowStatus,
        vrName
            DisplayString,
        vrContextName
            DisplayString,
        vrTrapEnable
            TruthValue,
        vrMaxRoutes
            Unsigned32,
        vrAdminStatus
            INTEGER,
        vrVpnId
            VpnIdentifier,
        vrRpTrigger
            Unsigned32
     }

vrId OBJECT-TYPE
    SYNTAX VrId
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The unique id of this virtual router instance. A Virtual
         Router cannot not be created with vrId = 0.
         VRID 0 is reserved for the Administrative VR.
        "
::= { vrConfigEntry 1 }

vrRowStatus OBJECT-TYPE
    SYNTAX RowStatus
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The status column has three defined values:

        - `active', which indicates that the conceptual row is
        available for use by the managed device;
```

```
        - `createAndGo', which is supplied by a management
        station wishing to create a new instance of a
        conceptual row and to have its status automatically set
        to active, making it available for use by the managed
        device;

        - `destroy', which is supplied by a management station
        wishing to delete all of the instances associated with
        an existing conceptual row."
::= { vrConfigEntry 2 }

vrName OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The Name of the Virtual Router..
         "
    ::= { vrConfigEntry 3 }

vrContextName OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The SNMPv2 Community String or SNMPv3 contextName
        denotes the VR 'context' and is used to logically
        separate the MIB management.
        RFC2571 and RFC2737 describe this approach."
    ::= { vrConfigEntry 4 }

vrTrapEnable OBJECT-TYPE
    SYNTAX TruthValue
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "This objects is used to enable the generation
        of the VrUp and VrDown traps.
            true(1)     - VR Traps Enabled
            false(2)    - VR Traps Disabled"
    DEFVAL { true }
    ::= { vrConfigEntry 5 }

vrMaxRoutes OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "This object specifies the maximum number of routes that
        this VR can support. The default value is 4 Gig (meaning
        unlimited)."
```

```
    DEFVAL { 4294967295 }
    ::= { vrConfigEntry 6 }

vrAdminStatus OBJECT-TYPE
    SYNTAX  INTEGER {
             up(1),
             down(2),
             testing(3),
             unknown(4)
            }
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The administrative state of the Virtual Router."
    DEFVAL { down }
    ::= { vrConfigEntry 7 }

vrVpnId OBJECT-TYPE
    SYNTAX  VpnIdentifier
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The Virtual Private Network Identifier of the Virtual
         Router."
    ::= { vrConfigEntry 8 }

vrRpTrigger OBJECT-TYPE
    SYNTAX  Unsigned32
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The Routing Protocol Triggers on the Virtual Router.
        This can be used to initiate or shutdown routing protocols
        on a VR.
        The 32 bits are divided into:
            16 bits of RP bitmap,
            15 bits reserved (0), and 1 bit of action-code.


                     0 1 2 3 4 5 6 7
                    +-+-+-+-+-+-+-+-+
                    |RP bitmap (MSB)|
                    +-+-+-+-+-+-+-+-+
                    |RP bitmap (LSB)|
                    +-+-+-+-+-+-+-+-+
                    |   Reserved    |
                    +-+-+-+-+-+-+-+-+
                    |*|  Reserved   |
                    +-+-+-+-+-+-+-+-+    *=action-code
```

```
        The RP bitmap specify the RP that is to be initiated or
        shutdown. Multiple RPs can be acted on simultaneously.
        Also, individual RPs can be brought up in steps, which
        should not affect the RPs that were running.
        Action-code specify what needs to be done for the RPs
        in the RP bitmap.  The actions are: initiate or shutdown.

        The running status of the RP shall be available in the
        VR stats table's vrRpStatus, which has a similar
        format, but represent the status."
    ::= { vrConfigEntry 9 }

vrStat OBJECT IDENTIFIER ::= { vrMIBObjects 2 }

vrStatScalars OBJECT IDENTIFIER ::= { vrStat 1 }

vrConfiguredVRs OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of VRs configured on this network element."
    ::= { vrStatScalars 1 }

vrActiveVRs OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of VRs that are active on the network element.
        These are VRs for which the
        vrStatOperationalStatus  = up(1)"
    ::= { vrStatScalars 2 }

vrStatTable OBJECT-TYPE
    SYNTAX SEQUENCE OF VrStatEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "This table contains statistics for the Virtual Router."
    ::= { vrStat 2 }

vrStatEntry OBJECT-TYPE
    SYNTAX VrStatEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Entries in this table a per vrId."
    INDEX { vrId }
    ::= { vrStatTable 1 }
```

```
VrStatEntry ::=
    SEQUENCE {
        vrStatRouteEntries
            Unsigned32,
        vrStatFIBEntries
            Unsigned32,
        vrStatUpTime
            TimeTicks,
        vrOperStatus
            INTEGER,
        vrRpStatus
            Unsigned32,
        vrRouterAddressType
            InetAddressType,
        vrRouterAddress
            InetAddress
     }

vrStatRouteEntries OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Total number of routes for this VR."
    ::= { vrStatEntry 1 }

vrStatFIBEntries OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Total number of FIB Entries for this VR."
    ::= { vrStatEntry 2 }

vrStatUpTime OBJECT-TYPE
    SYNTAX TimeTicks
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The time in (in hundredths of a second) since
        this VR entry has been operational."
    ::= { vrStatEntry 3 }

vrOperStatus OBJECT-TYPE
    SYNTAX  INTEGER {
             up(1),
             down(2),
             unknown(3)
            }
    MAX-ACCESS read-only
    STATUS current
```

```
    DESCRIPTION
        "The operational state of the Virtual Router."
    ::= { vrStatEntry 4 }

vrRpStatus OBJECT-TYPE
    SYNTAX  Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "List of Routing Protocols on this VR."
    ::= { vrStatEntry 5 }

vrRouterAddressType OBJECT-TYPE
    SYNTAX  InetAddressType
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Router Address Type of this VR."
    ::= { vrStatEntry 6 }

vrRouterAddress OBJECT-TYPE
    SYNTAX  InetAddress
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Router Address of this VR.  It is derived from one of the
        interfaces. If loopback interface is present, the loopback
        interface address can be used. However, loopback interface
        is optional."
    ::= { vrStatEntry 7 }

vrIfConfig OBJECT IDENTIFIER ::= { vrMIBObjects 3 }

vrIfConfigScalars OBJECT IDENTIFIER ::= { vrIfConfig 1 }

vrIfConfigTable OBJECT-TYPE
    SYNTAX SEQUENCE OF VrIfConfigEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "This table is for configuring VR Interfaces."
    ::= { vrIfConfig 2 }

vrIfConfigEntry OBJECT-TYPE
    SYNTAX VrIfConfigEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Entries in this table correspond to the entries in
        the ifTable that apply to the Virtual Router."
    INDEX { vrId,
```

```
                vrIfId }
        ::= { vrIfConfigTable 1 }

    VrIfConfigEntry ::=
        SEQUENCE {
            vrIfId
               InterfaceIndex,
            vrIfConfigRowStatus
               RowStatus
         }

    vrIfId OBJECT-TYPE
        SYNTAX InterfaceIndex
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
            "Virtual Router Interface Index."
        ::= { vrIfConfigEntry 1 }

     vrIfConfigRowStatus  OBJECT-TYPE
        SYNTAX RowStatus
        MAX-ACCESS read-create
        STATUS current
        DESCRIPTION
            " This object is used to create, delete or
            modify a row in this table."
        ::= { vrIfConfigEntry 2 }


-- ****************************************************************
-- Module Traps/Notifications
-- ****************************************************************

    vrNotificationsPrefix OBJECT IDENTIFIER ::= { vrMIBObjects 4 }

    vrNotifications OBJECT IDENTIFIER ::= { vrNotificationsPrefix 0 }

    vrUp NOTIFICATION-TYPE
        OBJECTS { vrRowStatus }
        STATUS current
        DESCRIPTION
            "This notification is generated when the specified
            VR is about to initialized or change the status from
            down to up."
        ::= { vrNotifications 1 }

    vrDown NOTIFICATION-TYPE
        OBJECTS { vrRowStatus }
        STATUS current
        DESCRIPTION
            "This notification is generated when the specified
```

```
            VR is about to go down."
        ::= { vrNotifications 2 }

    vrMaxRoutesExceeded NOTIFICATION-TYPE
        OBJECTS { vrRowStatus, vrMaxRoutes, vrStatRouteEntries }
        STATUS current
        DESCRIPTION
            "This notification is generated when the specified VR has
            exceeded the maximum number of routes specified"
        ::= { vrNotifications 3 }


-- ****************************************************************
-- Module Compliance/Conformance Statements
-- ****************************************************************

    vrConformance OBJECT IDENTIFIER ::= { virtualRouterMIB 2 }

    vrCompliances OBJECT IDENTIFIER ::= { vrConformance 1 }

    vrMIBCompliance MODULE-COMPLIANCE
        STATUS current
        DESCRIPTION
            "The compliance statement for entities that implement the
            VIRTUAL-ROUTER-MIB.  Implementation of this MIB
            is strongly recommended for any platform targeted for a
            carrier-class environment."
        MODULE -- this module
            MANDATORY-GROUPS { vrConfigGroup, vrStatGroup,
                               vrIfGroup, vrNotificationGroup }
        ::= { vrCompliances 1 }

    vrGroups OBJECT IDENTIFIER ::= { vrConformance 2 }

    vrConfigGroup OBJECT-GROUP
        OBJECTS { vrRowStatus, vrName,
                  vrContextName, vrTrapEnable,
                  vrMaxRoutes, vrAdminStatus,
                  vrVpnId, vrRpTrigger,
                  vrConfigNextAvailableVrId }
            STATUS current
            DESCRIPTION
                "A collection of attributes that support provisioning
                of a virtual router."
            ::= { vrGroups 1 }

    vrStatGroup OBJECT-GROUP
        OBJECTS { vrConfiguredVRs, vrActiveVRs,
                  vrStatRouteEntries, vrStatFIBEntries, vrStatUpTime,
                  vrOperStatus, vrRpStatus, vrRouterAddress,
                  vrRouterAddressType }
```

```
        STATUS current
        DESCRIPTION
            "A collection of attributes that contain stats about the
            virtual router."
        ::= { vrGroups 2 }

    vrIfGroup OBJECT-GROUP
        OBJECTS { vrIfConfigRowStatus  }
        STATUS current
        DESCRIPTION
            "A collection of attributes that support provisioning of a
            virtual router interfaces."
        ::= { vrGroups 3 }

    vrNotificationGroup NOTIFICATION-GROUP
        NOTIFICATIONS { vrUp, vrDown, vrMaxRoutesExceeded }
        STATUS current
        DESCRIPTION
            "A collection of traps that are supported by the VR."
        ::= { vrGroups 4 }

END
```

10.0 Summary for Sub-IP Area

This document defines a MIB that provides a way to provision VPNs at the PE devices employing the Virtual Router model.

10.1 Where does it fit in the Picture of the Sub-IP Work

This work fits in the PPVPN Working Group.

10.2 Why is it Targeted at this WG

The WG is chartered with developing Provider Provisioned VPN solutions. This draft contributes to this.

10.3 Justification

The WG should consider this document since it provides a means to configure and manage Virtual Router based PPVPNs.

11.0 Security Considerations

The Administrative VR provides visibility into and control over multiple VPNs. As such, security considerations for implementations of the Administrative VR and associated control plane(s) are critical to the security of the VPNs supported on each PE device.

Use of any vrContextName MUST be allowed in the Administrative VR. Additional authentication and security mechanisms SHOULD be used for SNMP access in the Administrative VR.

VRs other than the Administrative VR MUST NOT have access to other VR® s Instantiated MIBs, and MAY have access to their own instantiated MIBs.

In VRs other than the Administrative VR, access to that VR® s instantiated MIBs MAY be permitted via that VR® s vrContextName. Use of any vrContextName other than that assigned to the accessed VR MUST result in an error, and implementations SHOULD provide a logging mechanism for such events.

12.0 Acknowledgments

Special thanks to Joan Cucchiara for providing valuable comments on this MIB.

13.0 References

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

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

[10] 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

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

[17] Ouldbrahim's VR draft, "Network Based IP VPN Architecture Using Virtual Routers", draft-ouldbrahim-vpn-vr-01.txt

[18] RFC 2685, "Virtual Private Networks Identifier"

[19] RFC 2764, "A Framework for IP Based Vitual Private Networks"

[20] RFC 2547bis, "BGP/MPLS VPNs", draft-rosen-rfc2547bis-03.txt

[21] "BGP/IPsec VPN", draft-declercq-bgp-ipsec-vpn-00.txt

[22] RFC 2667, "IP Tunnel MIB"

[PPVPN-FW] R. Callon, et al., "A Framework for Layer 3 Provider Provisioned Virtual Private Networks", draft-ietf-ppvpn-framework-06.txt, October 2002.

[PPVPN-VR] P. Knight, et al., "Network based IP VPN Architecture using Virtual Routers", draft-ietf-ppvpn-vpn-vr-03.txt, July 2002.

[PPVPN-VR-AS] A. Nagarajan, et al., "Applicability Statement for Virtual Router-based Layer 3 PPVPN approaches", draft-ietf-ppvpn-as-vr-00.txt, August 2002.

14.0 Authors' Addresses

Elwin Stelzer Eliazer
Corona Networks, Inc.
630 Alder Drive
Milpitas, CA 95035
USA
Phone: +1-408-519-3832
Email: elwinietf@yahoo.com

Samuel Hancock
Corona Networks, Inc.
630 Alder Drive
Milpitas, CA 95035
USA

Phone: +1-408-519-3800 Ext 421
Email: hancoc_s@yahoo.com

Benson Schliesser
SAVVIS Communications
1 **Savvis Parkway**
Town and Country, MO 63017
USA
Phone: +1-314-628-7036
Email: bensons@savvis.net

Joseph Laria
Level Stream Research
Wilmington, MA 01887
USA
Phone: +1-978-884-3537
Emain: jlaria@levelstream.com