

Access Control Model for version 3 of the  
Simple Network Management Protocol (SNMPv3)

18 June 1997

**Bert Wijnen**  
**IBM T. J. Watson Research**  
**wijnen@vnet.ibm.com**

Randy Presuhn  
BMC Software, Inc.  
rpresuhn@bmc.com

Keith McCloghrie  
Cisco Systems, Inc.  
kzm@cisco.com

[<draft-ietf-snmpv3-acm-00.txt>](#)

Status of this Memo

This document is an Internet-Draft. 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.''

To learn the current status of any Internet-Draft, please check the ``1id-abstracts.txt' listing contained in the Internet- Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

Abstract

This document describes the Access Control Model (ACM) for SNMP version 3 for use in the SNMP architecture [[SNMP-ARCH](#)]. This document defines the Elements of Procedure for applying access control to management information. This document also includes a MIB for remotely monitoring/managing the configuration parameters for this ACM.

## **0.1 Issues**

- Where do we do the miId to groupName mapping
- Should we use UTF-8 for human readable names like contextName, viewName, groupName etc.  
We now use a SnmpAdminString TC which still needs to be defined.
- acknowledgements needs expansion
- Do we want to mandate a standard out-of-the-box configuration.
- How do we return a proper indication of the error-counter to be used in a possible reportPDU.
- Do we keep the statistics (error counters) here or in MPC

## **0.2 Change Log**

[version 3.1] - This is the June 18 version.

- remove old (resolved) issues
- list new issues
- corrections/additions by myself (bert)
- corrections based on dbh comments
- removed change log of before 1st interim meeting.

[version 3.0] - this is the first ACM doc (June 12 version).

- Modifications as agreed at 1st Interim Meeting
  - Make Access Control Module a separate document
  - Use viewName as index instead of an integer
  - add notify\_view
  - use SnmpAdminString
- Other Modification
  - use miId and secModel
  - add groupTable
  - add/rename Stats counters

## **1. Introduction**

The Architecture for describing Internet Management Frameworks is composed of multiple subsystems:

- 1) a message processing and control subsystem,
- 2) a security subsystem,
- 3) an access control subsystem, and
- 4) orangelets.

It is important to understand the SNMP architecture and the terminology of the architecture to understand where the model described in this document fits into the architecture and interacts with other subsystems within the architecture. The reader is expected to have read and understood the description of the SNMP architecture, as defined in [[SNMP-ARCH](#)].

The Access Control subsystem of an SNMP engine provides services to orangelets so that these orangelets can check if access to an object is allowed or not.

An Access Control model has the responsibility for checking if a specific type of access (read, write, notify) to a particular object (instance) is allowed.

It is the purpose of this document to define a specific model of the Access Control subsystem, designated the SNMP version 3 Access Control model.

### **1.2 Access Control**

Access Control occurs (either implicit or explicit) in an SNMP engine acting in an agent role when processing SNMP request messages from an SNMP engine acting in a manager role. These request messages include these types of operations: GetRequest, GetNextRequest, GetBulkRequest, and SetRequest operations.

Access Control also occurs in an SNMP engine when an SNMP notification message is generated. These notification messages include these types of operations: InformRequest and SNMPv2-trap

operations.

Access Control via the Access Control module only occurs if the orangelet that processes or generates the operation explicitly calls upon the access control service for checking of access rights. So it is the responsibility of an orangelet to make the proper calls for access checking.

### **1.3 Local Configuration Datastore**

To implement the model described in this document, each SNMP engine needs to retain its own set of information about access

Wijnen/Presuhn/McCloghrie Expires December 1997 [Page 3]

Draft Access Control Model (ACM) for SNMPv3 June 1997

rights and policies, and the like. This set of information is called the SNMP engine's Local Configuration Datastore (LCD) because it is locally-stored information.

In order to allow an SNMP engine's LCD to be remotely configured, portions of the LCD need to be accessible as managed objects. A MIB module, the SNMPv3 Access Control Model Configuration MIB, which defines these managed object types is included in this document.

## **2. Elements of the Model**

This section contains definitions to realize the access control applied by this Access Control Model.

### **2.1 Groups**

A groupName identifies a group (set) of zero or more securityIdentities on whose behalf SNMP management objects can be accessed. The Access Control module assumes the securityIdentity has already been authenticated as needed and provides no authentication by itself.

This SNMPv3 Access Control model requires the securityModel and the securityIdentity to be passed as input to the Access Control module when a request is made to check for access rights.

### **2.2 Level of Security (LoS)**

Different access rights can be defined for different Levels of Security. The LoS identifies the Level of Security that will be assumed when checking for access rights.

This Access Control Model requires the LoS to be passed as input to the Access Control module when a request is made to check access rights.

### **2.3 Contexts**

An SNMP context is a collection of management information accessible by an SNMP agent. An item of management information may exist in more than one context. An SNMP agent potentially has access to many contexts.

## **2.4 Access Policy**

This Access Control model determines the access rights of groups (representing zero, one or more securityIdentities which have the same access rights). For a particular context (contextName) to which a group (groupName) has access using a particular Level of Security (LoS), that group's access rights are given by a read-view, a write-view and a notify-view.

The read-view is the set of object instances authorized for the group when reading objects. Reading objects occurs when processing a retrieval (Get, GetNext, GetBulk) operation.

The write-view is the set of object instances authorized for the group when writing objects. Writing objects occurs when processing a Set operation.

Wijnen/Presuhn/McCloghrie Expires December 1997 [Page 5]

Draft Access Control Model (ACM) for SNMPv3 June 1997

The notify-view is the set of object instances authorized for the group when sending objects in a notification. Such occurs when sending a notification (Inform or Trap).

### 3. Elements of Procedure

This section describes the procedures followed by the Access Control module that implements this Access Control Model when checking access rights as requested by an orangelet.

The abstract service interface into the access control service is:

```
Boolean is_access_allowed ( secModel, miId, LoS,  
                           viewType, contextName,  
                           variableName  
                           )
```

Where:

Boolean	- FALSE if no access is allowed. TRUE if access is allowed.
secModel	- security model to which the miId belongs.

miId           - security model independent ID (securityIdentity).  
LoS           - Level of Security  
viewType       - view to be checked (read, write or notify).  
contextName    - context in which the variable\_name is accessed.  
variableName   - variable that is accessed.

### **3.1 Processing the is\_access\_allowed service request**

This section describes the procedure followed by the Access Control module whenever it receives a request to check if access is allowed.

- (1) The LCD (snmpV3AcContextTable) is consulted for information about the SNMP context identified by the contextName. If information about this SNMP context is absent from the LCD, then the snmpV3AcStatsUnknownContexts counter is incremented, and FALSE is returned to the caller.
- (2) The LCD (snmpV3AcGroupTable) is consulted for information about the security model (secModel) and securityIdentity (miId). If information about this combination is absent from the LCD, then the snmpV3AcStatsNoGroups counter is incremented, and FALSE is returned to the caller.
- (3) The LCD (snmpV3AcTable) is consulted for information about the contextName, groupName and LoS. If information about this combination is absent from the LCD, then the snmpV3AcStatsNoViews counter is incremented, and FALSE is returned to the caller.
- (4) If the SNMPv2 viewType is the read, then the read-view is used for checking if the variableName is accessible.  
If access is allowed, then TRUE is returned to the caller.  
Otherwise the snmpV3AcStatsUnauthorizedAccesses counter is incremented and FALSE is returned to the caller.

Wijnen/Presuhn/McCloghrie           Expires December 1997           [Page 7]

Draft                   Access Control Model (ACM) for SNMPv3           June 1997

- (5) If the SNMPv2 viewType is the write, then the write-view is used for checking if the variableName is accessible.  
If access is allowed, then TRUE is returned to the caller.  
Otherwise the snmpV3AcStatsUnauthorizedAccesses counter is incremented and FALSE is returned to the caller.
- (6) If the SNMPv2 viewType is the notify, then the notify-view is used for checking if the variableName is accessible.  
If access is allowed, then TRUE is returned to the caller.  
Otherwise the snmpV3AcStatsUnauthorizedAccesses counter is



incremented and FALSE is returned to the caller.

Editor's note:

We decided that a boolean would be returned. Maybe it is better to return a status\_code which can have one of these values:

- otherError
- accessAllowed
- unknownContext
- noGroup
- noView
- accessNotAllowed

Then the caller can generate the appropriate reportPDU (or tell the MPC to generate the appropriate reportPDU).

End Editor's note

Wijnen/Presuhn/McCloghrie Expires December 1997 [Page 8]

Draft Access Control Model (ACM) for SNMPv3 June 1997

#### **4. Definitions**

SNMPV3-AC-MIB DEFINITIONS ::= BEGIN

IMPORTS

Counter32, Unsigned32, BITS,

MODULE-IDENTITY, OBJECT-TYPE, snmpModules FROM SNMPv2-SMI  
TEXTUAL-CONVENTION, TestAndIncr,  
RowStatus, StorageType, FROM SNMPv2-TC  
MODULE-COMPLIANCE, OBJECT-GROUP FROM SNMPv2-CONF  
SnmpAdminString,  
SnmpLoS,  
SnmpSecurityModel FROM SNMPv3-MIB;

snmpV3AcMIB MODULE-IDENTITY

LAST-UPDATED "9706180000Z" -- 18 June 1997, midnight  
ORGANIZATION "SNMPv3 Working Group"  
CONTACT-INFO "WG-email: snmpv3@tis.com  
Subscribe: majordomo@tis.com  
In msg body: subscribe snmpv3

Chair: Russ Mundy  
Trusted Information Systems  
postal: 3060 Washington Rd  
Glenwood MD 21738  
email: mundy@tis.com  
phone: 301-854-6889

Co-editor: Bert Wijnen  
IBM T.J. Watson Research  
postal: Schagen 33  
3461 GL Linschoten  
Netherlands  
email: wijnen@vnet.ibm.com  
phone: +31-348-412-498

Co-editor: Randy Presuhn  
BMC Software, Inc  
postal: 1190 Saratoga Avenue, Suite 130  
San Jose, CA 95129-3433  
USA  
email: rpresuhn@bmc.com  
phone: +1-408-556-0720

Co-editor: Keith McCloghrie  
Cisco Systems, Inc.  
postal: 170 West Tasman Drive  
San Jose, CA 95134-1706  
USA  
email: kzm@cisco.com  
phone: +1-408-526-5260

```

"

DESCRIPTION "The management information definitions for the
             SNMPv3 Access Control Model.
"

::= { snmpModules 99 }

-- Administrative assignments *****

snmpV3AcMIBObjects      OBJECT IDENTIFIER ::= { snmpV3AcMIB 1 }
snmpV3AcMIBConformance OBJECT IDENTIFIER ::= { snmpV3AcMIB 2 }

-- Statistics for Access Control Checking *****

snmpV3AcStats OBJECT IDENTIFIER ::= { snmpV3AcMIBObjects 1 }

snmpV3AcStatsUnknownContexts OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The total number of packets received by the SNMP
                engine which were dropped because they referenced a
                context that was not known to the engine.
"
    ::= { snmpV3AcStats 1 }

snmpV3AcStatsNoGroups OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The total number of packets received by the SNMP
                engine which were dropped because the security model
                independent ID (securityIdentity, miId) did not map
                a group in the snmpV3AcGroupTable.
"
    ::= { snmpV3AcStats 2 }

snmpV3AcStatsNoViews OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The total number of packets received by the SNMP
                engine which were dropped because the combination
                of contextName, groupName and LoS does not have
                an entry in the snmpV3AcTable at all.
"
    ::= { snmpV3AcStats 3 }

snmpV3AcStatsUnauthorizedAccesses OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only

```



```

STATUS      current
DESCRIPTION "The total number of packets received by the SNMP
            engine which were dropped because the type of access
            requested is invalid or not authorized.
            "
 ::= { snmpV3AcStats 4 }

```

```
-- Information about Mapping of miId into a group *****
```

```

-- Editor's question:
--   I have included the mapping table for the miId into a
--   groupName into this MIB. I think that keeps the acces
--   control nicely grouped together. Comments?
-- End Editor's question.

```

```

snmpV3AcGroupTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SnmpV3AcGroupEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION "The table that maps the Security Model Independent ID
                into a groupName which defines an acces control
                policy for a group of security identities.
                "
    ::= { snmpV3AcMIBObjects 2 }

```

```

snmpV3AcGroupEntry OBJECT-TYPE
    SYNTAX      SnmpV3AcGroupEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION "An entry in this table maps a miId into a groupName."
    INDEX       { snmpV3AcSecModel,
                  snmpV3AcMiId
                }
    ::= { snmpV3AcGroupTable 1 }

```

```

SnmpV3AcGroupEntry ::= SEQUENCE
{
    snmpV3AcSecModel      SnmpV3SecurityModel,
    snmpV3AcMiId          SnmpV3AdminString,
    snmpV3AcGroupName     SnmpV3AdminString,
    snmpV3AcGroupStorageType StorageType,
    snmpV3AcGroupStatus   RowStatus
}

```

```

snmpV3AcSecModel OBJECT-TYPE
    SYNTAX      SnmpV3SecurityModel
    MAX-ACCESS  not-accessible
    STATUS      current

```

DESCRIPTION "The security model, which is the first index in this  
table.  
"

Wijnen/Presuhn/McCloghrie

Expires December 1997

[Page 11]

```
::= { snmpV3AcGroupEntry 1 }
```

```
snmpV3AcMiId      OBJECT-TYPE
    SYNTAX         SnmpV3AdminString
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION    "The Security Model Independent ID (miId) for a
                    particular security identity. It is used as a second
                    index in this table.
    "
```

```
::= { snmpV3AcGroupEntry 2 }
```

```
snmpV3AcGroupName OBJECT-TYPE
    SYNTAX         SnmpV3AdminString
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION    "The groupName to which this miId belongs. This
                    groupName represents a access control policy and is
                    used as an index in the snmpV3AcTable.
    "
```

```
::= { snmpV3AcGroupEntry 3 }
```

```
snmpV3AcGroupStorageType OBJECT-TYPE
    SYNTAX         StorageType
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION    "The storage type for this conceptual row.
                    Conceptual rows having the value 'permanent'
                    need not allow write-access to any columnar
                    objects in the row.
    "
```

```
DEFVAL           { nonVolatile }
```

```
::= { snmpV3AcGroupEntry 6 }
```

```
snmpV3AcGroupStatus OBJECT-TYPE
    SYNTAX         RowStatus
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION    "The status of this conceptual row.

                    For those columnar objects which permit write-access,
                    their value in an existing conceptual row can be
                    changed irrespective of the value of
                    snmpV3AcGroupStatus for that row.
    "
```

```
::= { snmpV3AcGroupEntry 7 }
```

```
-- Information about Local Contexts *****
```

snmpV3AcContextTable OBJECT-TYPE  
SYNTAX SEQUENCE OF SnmpV3AcContextEntry

Wijnen/Presuhn/McCloghrie

Expires December 1997

[Page 12]



MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION "The table of locally available contexts. If a context is listed in this table that does not mean that access to this context has been defined in the snmpV3AcTable. It just means that the context exists and that MIB objects may exist in this context.

This table must be made accessible via the default context.

This table is read-only meaning that SNMP engines in a manager role cannot configure contexts.

Instead the table is meant to provide input to SNMP engines in a manager role such that they can properly configure the snmpV3AcTable to control access to all contexts in an SNMP engine operating in an agent role.

"

::= { snmpV3AcMIBObjects 3 }

snmpV3AcContextEntry OBJECT-TYPE

SYNTAX SnmpV3AcContextEntry  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION "Information about a particular context."  
 INDEX { snmpV3AcContextName }  
 ::= { snmpV3AcContextTable 1 }

SnmpV3AcContextEntry ::= SEQUENCE

```
{
    snmpV3AcContextName      SnmpV3AdminString
}
```

snmpV3AcContextName OBJECT-TYPE

SYNTAX SnmpV3AdminString (SIZE(0..32))  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION "A textual name uniquely identifying a particular context on a particular agent."  
 "

::= { snmpV3AcContextEntry 1 }

-- Information about Access Rights \*\*\*\*\*

snmpV3AcTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnmpV3AcEntry  
 MAX-ACCESS not-accessible

STATUS current

DESCRIPTION "The table of group access rights configured in the

Wijnen/Presuhn/McCloghrie

Expires December 1997

[Page 13]

local configuration datastore (LCD).

Each entry is indexed by a contextName, a GroupName and a Level of Security (LoS). When checking if access is allowed, then one entry from this table needs to be selected and the proper viewName from that entry must be used for access control checking.

To select the proper entry, first a match must be found for the contextName. The procedure for this process depends on the value of snmpV3AcContextMatch:

- exact

In this case, the snmpV3AcContextName represents an exact contextName, and so the name must match exactly.

- prefix

In this case, the snmpV3AcContextName represents a prefix of a contextName, so that (a limited form of) wildcarding is possible. The value of snmpV3AcContextName must match with the first part of the contextName to which access is requested.

For example, if we use a prefix contextName 'repeater', then both contexts named 'repeater1' and 'repeater2' are accessible.

In case multiple entries match, then the entry with the longest snmpV3AcContextName wins.

The second match to make is for the groupName. Here an exact match must be found.

The 3rd match to make is for the LoS. Here an exact match must be found.

"

```
-- Editors Question to Keith:
-- I have removed snmpV3AcContextName from the AcTable.... I was
-- thinking that it has the same semantics as snmpV3AcContextName
-- in the SnmpV3AcContextTable above. But now that we also allow
-- for wildcarding here, now I am not so sure that the semantics
-- are indeed the same. Should I define a snmpV2AcContextPrefix
-- instead?
-- End Editors Question
 ::= { snmpV3AcMIBObjects 4 }
```

snmpV3AcEntry	OBJECT-TYPE
SYNTAX	SnmpV3AcEntry
MAX-ACCESS	not-accessible

STATUS current  
DESCRIPTION "An access right configured in the local configuration  
datastore (LCD) authorizing access to an SNMP context."

```

"
INDEX      { snmpV3AcContextName,
              snmpV3AcGroupName,
              snmpV3AcLoS
            }
 ::= { snmpV3AcTable 1 }

```

SnmpV3AcEntry ::= SEQUENCE

```

{
    snmpV3AcLoS           SnmpV3LoS,
    snmpV3AcContextMatch  INTEGER,
    snmpV3AcReadViewName  SnmpV3AdminString,
    snmpV3AcWriteViewName SnmpV3AdminString,
    snmpV3AcNotifyViewName SnmpV3AdminString,
    snmpV3AcStorageType   StorageType,
    snmpV3AcStatus        RowStatus
}

```

```

snmpV3AcLoS      OBJECT-TYPE
SYNTAX            SnmpV3LoS
MAX-ACCESS        not-accessible
STATUS            current
DESCRIPTION       "The minimum level of security required in order to
                  gain the access rights allowed by this conceptual
                  row."
"
 ::= { snmpV3AcEntry 1 }

```

snmpV3AcContextMatch OBJECT-TYPE

```

SYNTAX            INTEGER
                  { exact (0),      -- exact match of context Name
                    prefix (1)     -- Only match to this prefix
                  }
MAX-ACCESS        read-create
STATUS            current
DESCRIPTION       "If exact is set, then the contextName of the
                  index part snmpV3AcContextName of this entry in this
                  table represents a full contextName.

                  If prefix is set, then the contextName of the
                  index part snmpV3AcContextName of this entry in this
                  table represents a partial contextName which acts
                  as a prefix so that a simple form of wilddcarding
                  is possible."
"
 ::= { snmpV3AcEntry 2 }

```

snmpV3AcReadViewName OBJECT-TYPE

SYNTAX	SnmpV3AdminString
MAX-ACCESS	read-create
STATUS	current

DESCRIPTION "The value of an instance of this object identifies the MIB view of the SNMP context to which this conceptual row authorizes read access.

The identified MIB view is that for which snmpV3AcViewName has the same value as the instance of this object; if the value is the empty string or if there is no active MIB view having this value of snmpV3AcViewName, then no access is granted.

Otherwise, this object is ignored and can take any value at the Access Control module's discretion, e.g., the empty string.

"

DEFVAL { 'H } -- the empty string  
::= { snmpV3AcEntry 3 }

snmpV3AcWriteViewName OBJECT-TYPE

SYNTAX SnmpV3AdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The value of an instance of this object identifies the MIB view of the SNMP context to which this conceptual row authorizes write access.

The identified MIB view is that for which snmpV3AcViewName has the same value as the instance of this object; if the value is the empty string or if there is no active MIB view having this value of snmpV3AcViewName, then no access is granted.

Otherwise, this object is ignored and can take any value at the Access Control module's discretion, e.g., the empty string.

"

DEFVAL { 'H } -- the empty string  
::= { snmpV3AcEntry 4 }

snmpV3AcNotifyViewName OBJECT-TYPE

SYNTAX SnmpV3AdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The value of an instance of this object identifies the MIB view of the SNMP context to which this conceptual row authorizes access for notifications.

The identified MIB view is that for which snmpV3AcViewName has the same value as the instance

of this object; if the value is the empty string or if there is no active MIB view having this value of snmpV3AcViewName, then no access is granted.



Otherwise, this object is ignored and can take any value at the Access Control module's discretion, e.g., the empty string.

"

DEFVAL { 'H } -- the empty string  
::= { snmpV3AcEntry 5 }

snmpV3AcStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The storage type for this conceptual row.

Conceptual rows having the value 'permanent' need not allow write-access to any columnar objects in the row.

"

DEFVAL { nonVolatile }  
::= { snmpV3AcEntry 6 }

snmpV3AcStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The status of this conceptual row.

For those columnar objects which permit write-access, their value in an existing conceptual row can be changed irrespective of the value of snmpV3AcStatus for that row.

"

::= { snmpV3AcEntry 7 }

-- Information about MIB views \*\*\*\*\*

-- Support for views having instance-level granularity is optional

snmpV3AcViewTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnmpV3AcViewEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION "The table of locally defined MIB views.

When an SNMP engine in the manager role wants to create a new MIB view, then it must first create an entry in the snmpV3AcViewTable, using a non-existing index-value for snmpV3AcViewName. If the creation of such an entry is successful, the SNMP engine in the manager role can then start

creating entries in the snmpV3AcSubtreeFamilyTable.

When deleting MIB views, it is strongly advised that first the related `snmpV3AcSubtreeFamilyEntries` are deleted from the `snmpV3AcSubtreeFamilyTable` and that only upon completion of that deletion process the `snmpV3AcViewEntry` is deleted from the `snmpV3AcViewTable`.

Furthermore, a manager should take great care to delete all the 'included' family entries before deleting any of the 'excluded' ones.

Following these procedures there should be no collisions when multiple managers try to update the MIB views at an SNMP engine in an agent role.

If managers do not follow these procedures then it is agent-implementation dependent as to what the result of possible collisions will be.

"

::= { snmpV3AcMIBObjects 5 }

`snmpV3AcViewEntry` OBJECT-TYPE

SYNTAX            `SnmpV3AcViewEntry`  
MAX-ACCESS       not-accessible  
STATUS            current  
DESCRIPTION       "Information on a particular local MIB view."  
INDEX             { `snmpV3AcViewName` }  
::= { `snmpV3AcViewTable` 1 }

`SnmpV3AcViewEntry` ::= SEQUENCE

{  
    `snmpV3AcViewName`                `SnmpV3AdminString`,  
    `snmpV3AcViewStorageType`       `StorageType`,  
    `snmpV3AcViewStatus`             `RowStatus`  
}

`snmpV3AcViewName` OBJECT-TYPE

SYNTAX            `SnmpV3AdminString` (SIZE(1..32))  
MAX-ACCESS       not-accessible  
STATUS            current  
DESCRIPTION       "An unique viewName that uniquely identifies a MIB  
                    viewEntry in this table."  
                    "

::= { `snmpV3AcViewEntry` 1 }

`snmpV3AcViewStorageType` OBJECT-TYPE

SYNTAX            `StorageType`  
MAX-ACCESS       read-create

STATUS           current  
DESCRIPTION "The storage type for this conceptual row.

Wijnen/Presuhn/McCloghrie

Expires December 1997

[Page 18]

Conceptual rows having the value 'permanent' need not allow write-access to any columnar objects in the row.

"

DEFVAL { nonVolatile }  
::= { snmpV3AcViewEntry 2 }

snmpV3AcViewStatus OBJECT-TYPE

SYNTAX RowStatus  
MAX-ACCESS read-create  
STATUS current  
DESCRIPTION "The status of this conceptual row.

For those columnar objects which permit write-access, their value in an existing conceptual row can be changed irrespective of the value of snmpV3AcViewStatus for that row.

"

::= { snmpV3AcViewEntry 3 }

-- Subtree families of MIB views \*\*\*\*\*

snmpV3AcSubtreeFamilyTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnmpV3AcSubtreeFamilyEntry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION "Locally held information about families of subtrees within MIB views.

Each MIB view is defined by two collections of view subtrees: the included view subtrees, and the excluded view subtrees.

Every such subtree, both included and excluded, is defined in this table.

To determine if a particular object instance is in a particular MIB view, compare the object instance's OBJECT IDENTIFIER with each of the MIB view's active entries in this table. If none match, then the object instance is not in the MIB view. If one or more match, then the object instance is included in, or excluded from, the MIB view according to the value of snmpV3AcSubtreeFamilyType in the entry whose value of snmpV3AcSubtreeFamilySubtree has the most sub-identifiers. If multiple entries match and have the same number of sub-identifiers, then the lexicographically greatest instance of snmpV3AcSubtreeFamilyType determines the inclusion

or exclusion.

An object instance's OBJECT IDENTIFIER X matches an

active entry in this table when the number of sub-identifiers in X is at least as many as in the value of `snmpV3AcSubtreeFamilySubtree` for the entry, and each sub-identifier in the value of `snmpV3AcSubtreeFamilySubtree` matches its corresponding sub-identifier in X. Two sub-identifiers match either if the corresponding bit of `snmpV3AcSubtreeFamilyMask` is zero (the 'wild card' value), or if they are equal.

A 'family' of view subtrees is the set of subtrees defined by a particular combination of values of `snmpV3AcSubtreeFamilySubtree` and `snmpV3AcSubtreeFamilyMask`.

In the case where no 'wild card' is defined in `snmpV3AcSubtreeFamilyMask`, the family of view subtrees reduces to a single view subtree.

When an SNMP engine in the manager role wants to create a new MIB view, then it should first create an entry in the `snmpV3AcViewTable`, using a non-existing index-value for `snmpV3AcViewName`. If the creation of such an entry is successful, the SNMP engine in the manager role can then start creating entries in the `snmpV3AcSubtreeFamilyTable`.

When deleting MIB views, it is strongly advised that first the related `snmpV3AcSubtreeFamilyEntries` are deleted from the `snmpV3AcSubtreeFamilyTable` and that only upon completion of that deletion process the `snmpV3AcViewEntry` is deleted from the `snmpV3AcViewTable`.

Following these procedures there should be no collisions when multiple managers try to update the MIB views at an SNMP engine in an agent role.

"

::= { `snmpV3AcMIBObjects` 6 }

`snmpV3AcSubtreeFamilyEntry` OBJECT-TYPE

SYNTAX `SnmpV3AcSubtreeFamilyEntry`

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION "Information on a particular family of view subtrees included in or excluded from a particular SNMP context's MIB view. The MIB view must exist (i.e., be represented by a conceptual row in the `snmpV3AcViewTable`) before any subtree families can

be defined for it.

Implementations must not restrict the number of



families of view subtrees for a given MIB view, except as dictated by resource constraints on the overall number of entries in the snmpV3AcSubtreeFamilyTable.

The value of snmpV3AcViewName in this INDEX clause of this table identifies the MIB view in which this subtree family exists.

A MIB view for which there are no conceptual rows in this table is the empty set of view subtrees.

"

```
INDEX      { snmpV3AcViewName,
             IMPLIED snmpV3AcSubtreeFamilySubtree
           }
 ::= { snmpV3AcSubtreeFamilyTable 1 }
```

SnmpV3AcSubtreeFamilyEntry ::= SEQUENCE

```
{
    snmpV3AcSubtreeFamilySubtree    OBJECT IDENTIFIER,
    snmpV3AcSubtreeFamilyMask       OCTET STRING,
    snmpV3AcSubtreeFamilyType       INTEGER,
    snmpV3AcSubtreeFamilyStorageType StorageType,
    snmpV3AcSubtreeFamilyStatus     RowStatus
}
```

snmpV3AcSubtreeFamilySubtree OBJECT-TYPE

```
SYNTAX      OBJECT IDENTIFIER
MAX-ACCESS   not-accessible
STATUS       current
DESCRIPTION  "The MIB subtree which when combined with the
              corresponding instance of snmpV3AcSubtreeFamilyMask
              defines a family of view subtrees.
"
 ::= { snmpV3AcSubtreeFamilyEntry 1 }
```

snmpV3AcSubtreeFamilyMask OBJECT-TYPE

```
SYNTAX      OCTET STRING (SIZE (0..16))
MAX-ACCESS   read-create
STATUS       current
DESCRIPTION  "The bit mask which,
              in combination with the corresponding instance of
              snmpV3AcSubtreeFamilySubtree, defines a family of
              view subtrees.
```

Each bit of this bit mask corresponds to a sub-identifier of snmpV3AcSubtreeFamilySubtree, with the most significant bit of the i-th octet

of this octet string value (extended if necessary,  
see below) corresponding to the  $(8*i - 7)$ -th  
sub-identifier, and the least significant bit of

the  $i$ -th octet of this octet string corresponding to the  $(8*i)$ -th sub-identifier, where  $i$  is in the range 1 through 16.

Each bit of this bit mask specifies whether or not the corresponding sub-identifiers must match when determining if an OBJECT IDENTIFIER is in this family of view subtrees; a '1' indicates that an exact match must occur; a '0' indicates 'wild card', i.e., any sub-identifier value matches.

Thus, the OBJECT IDENTIFIER  $X$  of an object instance is contained in a family of view subtrees if, for each sub-identifier of the value of `snmpV3AcSubtreeFamilySubtree`, either:

the  $i$ -th bit of `snmpV3AcSubtreeFamilyMask` is 0, or

the  $i$ -th sub-identifier of  $X$  is equal to the  $i$ -th sub-identifier of the value of `snmpV3AcSubtreeFamilySubtree`.

If the value of this bit mask is  $M$  bits long and there are more than  $M$  sub-identifiers in the corresponding instance of `snmpV3AcSubtreeFamilySubtree`, then the bit mask is extended with 1's to be the required length.

Note that when the value of this object is the zero-length string, this extension rule results in a mask of all-1's being used (i.e., no 'wild card'), and the family of view subtrees is the one view subtree uniquely identified by the corresponding instance of `snmpV3AcSubtreeFamilySubtree`.

"

```
DEFVAL      { 'H' }
::= { snmpV3AcSubtreeFamilyEntry 2 }
```

`snmpV3AcSubtreeFamilyType` OBJECT-TYPE

```
SYNTAX      INTEGER { included(1), excluded(2) }
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION "The indication of whether the corresponding instances
            of snmpV3AcSubtreeFamilySubtree and
            snmpV3AcSubtreeFamilyMask define a family of view
            subtrees which is included in or excluded from the
            MIB view.
```

"

```
DEFVAL      { included }  
::= { snmpV3AcSubtreeFamilyEntry 3 }
```

## snmpV3AcSubtreeFamilyStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The storage type for this conceptual row.

Conceptual rows having the value 'permanent' need not allow write-access to any columnar objects in the row.

An SNMP engine in the manager role is advised to use the same value for this row as the value used in the corresponding row in the snmpV3AcViewTable.

"

DEFVAL { nonVolatile }

::= { snmpV3AcSubtreeFamilyEntry 4 }

## snmpV3AcSubtreeFamilyStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION "The status of this conceptual row.

For those columnar objects which permit write-access, their value in an existing conceptual row can be changed irrespective of the value of snmpV3AcSubtreeFamilyStatus for that row.

An SNMP engine in the manager role is advised to use the same value for this row as the value used in the corresponding row in the snmpV3AcViewTable.

"

::= { snmpV3AcSubtreeFamilyEntry 5 }

-- Conformance information \*\*\*\*\*

## snmpV3AcMIBCompliances

OBJECT IDENTIFIER ::= { snmpV3AcMIBConformance 1 }

## snmpV3AcMIBGroups

OBJECT IDENTIFIER ::= { snmpV3AcMIBConformance 2 }

-- Compliance statements \*\*\*\*\*

## snmpV3AcMIBCompliance MODULE-COMPLIANCE

STATUS current

DESCRIPTION "The compliance statement for SNMP engines which implement the SNMPv3 ACM configuration MIB.

"

MODULE -- this module

MANDATORY-GROUPS { snmpV3AcBasicGroup }

OBJECT	snmpV3AcContextMatch
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcReadViewName
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcWriteViewName
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcNotifyViewName
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcStorageType
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcStatus
MIN-ACCESS	read-only
DESCRIPTION	"Create access to the snmpV3AcViewTable is not required. "
OBJECT	snmpV3AcViewStorageType
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcViewStatus
MIN-ACCESS	read-only
DESCRIPTION	"Create access to the snmpV3AcViewTable is not required. "
OBJECT	snmpV3AcSubtreeFamilyMask
WRITE-SYNTAX	OCTET STRING (SIZE (0))
MIN-ACCESS	read-only
DESCRIPTION	"Support for configuration via SNMP of subtree families defined using wild-cards is not required. "
OBJECT	snmpV3AcSubtreeFamilyType
MIN-ACCESS	read-only
DESCRIPTION	"Write access is not required."
OBJECT	snmpV3AcSubtreeFamilyStorageType

MIN-ACCESS read-only  
DESCRIPTION "Write access is not required."

Wijnen/Presuhn/McCloghrie

Expires December 1997

[Page 24]



```
OBJECT      snmpV3AcSubtreeFamilyStatus
MIN-ACCESS  read-only
DESCRIPTION "Create access to the snmpV3AcSubtreeFamilyTable
            is not required.
            "
```

```
::= { snmpV3AcMIBCompliances 1 }
```

```
-- Units of conformance *****
```

```
snmpV3AcBasicGroup OBJECT-GROUP
```

```
OBJECTS { snmpV3AcStatsUnknownContexts,
          snmpV3AcStatsNoGroups,
          snmpV3AcStatsNoViews,
          snmpV3AcStatsUnauthorizedAccesses, -- length 33
          snmpV3AcGroupName,
          snmpV3AcGroupStorageType,
          snmpV3AcGroupStatus,
          snmpV3AcContextName,
          snmpV3AcReadViewName,
          snmpV3AcWriteViewName,
          snmpV3AcNotifyViewName,
          snmpV3AcStorageType,
          snmpV3AcStatus,
          snmpV3AcViewStorageType,
          snmpV3AcViewStatus,
          snmpV3AcSubtreeFamilyMask,
          snmpV3AcSubtreeFamilyType,
          snmpV3AcSubtreeFamilyStorageType, -- length 32
          snmpV3AcSubtreeFamilyStatus
        }
```

```
STATUS      current
```

```
DESCRIPTION "A collection of objects providing for remote
            configuration of an SNMP engine which implements
            the SNMPv3 Access Control Model (ACM).
            "
```

```
::= { snmpV3AcMIBGroups 1 }
```

```
END
```



## **5. Security Considerations**

### **5.1 Recommended Practices**

This document is meant for use in the SNMP architecture. The Access Control Model (ACM) described in this document controls access rights to management information based on:

- contextName, representing a set of management information at the managed system where the Access Control module is running.
- groupName, representing a group or set of zero, one or more securityIdentities. These securityIdentities are mapped into one or more groups in the SNMPv3 Access Control subsystem.
- Level of Security (LoS) used for the transmission of an SNMP message.

When the Access Control module (ACM) is called for checking access rights, it is assumed that the calling module has ensured the authentication and privacy aspects as specified by the Level of Security (LoS) that is being passed.

### **5.2 Defining Groups**

GroupNames are used to give access to a group of zero, one or more securityIdentities. Within the ACM, a groupName is considered to exist if that groupName is used (as an index) in a row in the snmpV3AcTable.

By mapping a securityIdentity into a group, a Management System can add/delete securityIdentities to/from a group.

### **5.3 Conformance**

Conformance rules are described in the SNMP architecture document [[SNMP-ARCH](#)].



## **6. Editor's Addresses**

Co-editor: Bert Wijnen  
IBM T. J. Watson Research  
postal: Schagen 33  
3461 GL Linschoten  
Netherlands  
email: wijnen@vnet.ibm.com  
phone: +31-348-432-794

Co-editor: Randy Presuhn  
BMC Software, Inc  
postal: 1190 Saratoga Avenue, Suite 130  
San Jose, CA 95129-3433  
USA  
email: rpresuhn@bmc.com  
phone: +1-408-556-0720

Co-editor: Keith McCloghrie  
Cisco Systems, Inc.  
postal: 170 West Tasman Drive  
San Jose, CA 95134-1706  
USA  
email: kzm@cisco.com  
phone: +1-408-526-5260

## **7. Acknowledgements**

This document describes the work of the SNMP Security and Administrative Framework Evolution team, comprised of

David Harrington (Cabletron Systems Inc.)  
Jeff Johnson (Cisco)  
David Levi (SNMP Research Inc.)  
John Linn (Openvision)  
Russ Mundy (Trusted Information Systems) chair  
Shawn Routhier (Epilogue)  
Glenn Waters (Nortel)  
Bert Wijnen (IBM T. J. Watson Research)



## 8. References

- [RFC1902] The SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1905](#), January 1996.
- [RFC1905] The SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1905](#), January 1996.
- [RFC1906] The SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1906](#), January 1996.
- [RFC1907] The SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1907](#), January 1996.
- [RFC1908] The SNMPv2 Working Group, 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.
- [SNMP-ARCH] The SNMPv3 Working Group, Harrington, D., Wijnen, B., "An Architecture for describing Internet Management Frameworks", [draft-ietf-snmpv3-next-gen-arch-02.txt](#), June 1997.
- [SNMPv3-ACM] The SNMPv3 Working Group, Wijnen, B., Harrington, D., "Access Control Model for Version 3 of the Simple Network Management Protocol (SNMPv3)", [draft-ietf-snmpv3-acm-00.txt](#), June 1997.
- [SNMPv3-MPC] The SNMPv3 Working Group, Wijnen, B., Harrington, D., "Message Processing and Control Model for version 3 of the Simple Network Management Protocol (SNMPv3)", [draft-ietf-snmpv3-mpc-00.txt](#), March 1997.
- [SNMPv3-USEC] The SNMPv3 Working Group, Blumenthal, U., Wijnen, B., "The User-Based Security Model for Version 3 of the Simple Network Management Protocol (SNMPv3)", [draft-ietf-snmpv3-usec-01.txt](#), June 1997.





## APPENDIX A - Installation

**A.1. Installation Parameters**

During installation, an SNMPv3 engine acting in an authoritative role is configured with several parameters. These include for the Access Control module:

## (1) A security posture

The choice of security posture determines the extent of the view configured for unauthenticated access. One of three possible choices is selected:

```

    minimum-secure,
    semi-secure, or
    very-secure.

```

## (2) A default context

One entry in the `snmpV3AcContextTable` with a `contextName` of "" (the empty string, representing the default context.

Editor's note:

If we do keep the `groupTable`, then we also need an entry in the `groupTable` for group `public`. It should have a `miId` of "public" for USEC that maps into `groupName` "public"

End Editor's note.

(3) Three entries in the `snmpV3AcTable` as follows:

- One entry to be used for unauthenticated access:

	no privacy support	privacy support
	-----	-----
<code>snmpV3AcContextName</code>	""	""
<code>snmpV3AcGroupName</code>	"public"	"public"
<code>snmpV3AcLoS</code>	noAuth/noPriv	noAuth/noPriv
<code>snmpV3AcReadViewName</code>	"restricted"	"restricted"
<code>snmpV3AcWriteViewName</code>	""	""
<code>snmpV3AcNotifyViewName</code>	"restricted"	"restricted"
<code>snmpV3AcStorageType</code>	permanent	permanent
<code>snmpV3AcStatus</code>	active	active

- One entry to be used for authenticated access but without privacy:

	no privacy support	privacy support
	-----	-----
<code>snmpV3AcContextName</code>	""	""

snmpV3AcGroupName  
snmpV3AcLoS

"public"  
Auth/noPriv

"public"  
Auth/noPriv

Wijnen/Presuhn/McCloghrie

Expires December 1997

[Page 29]

snmpV3AcReadViewName	"all"	"all"
snmpV3AcWriteViewName	"all"	"all"
snmpV3AcNotifyViewName	"all"	"all"
snmpV3AcStorageType	permanent	permanent
snmpV3AcStatus	active	active

- One entry to be used for authenticated access with privacy:

	no privacy support	privacy support
	-----	-----
snmpV3AcContextName		""
snmpV3AcGroupName		"public"
snmpV3AcLoS		Auth/Priv
snmpV3AcReadViewName		"all"
snmpV3AcWriteViewName		"all"
snmpV3AcNotifyViewName		"all"
snmpV3AcStorageType		permanent
snmpV3AcStatus		active

(4) Two views depending on the security posture.

- One view (the <all> view) for authenticated access:
  - the <all> MIB view is the following subtree:
 

"internet"	[ <a href="#">RFC1902</a> ]
------------	-----------------------------

Editor's note:

I picked this up from the [RFC1910](#).

I have experience myself that MIBs were defined outside the internet subtree, so maybe this should just be

"iso"

End Editor's note.

- A second view (the <restricted> view) for unauthenticated access. This view is configured according to the selected security posture:

- For the "very-secure" posture:

the <restricted> MIB view is the union of these subtrees:

"snmp"	[ <a href="#">RFC1907</a> ]
"snmpEngine"	[ <a href="#">SNMPv3-USEC</a> ]
"snmpV3Stats"	[ <a href="#">SNMPv3-MPC</a> ]
"snmpV3AcStats"	[ <a href="#">SNMPv3-ACM</a> ]

- For the "semi-secure" posture:

the <restricted> MIB view is the union of these subtrees:

"snmp"	[ <a href="#">RFC1907</a> ]
--------	-----------------------------

"snmpEngine"      [[SNMPv3-USEC](#)]  
"snmpV3Stats"    [[SNMPv3-MPC](#)]

"snmpV3AcStats"    [[SNMPv3-ACM](#)]  
"system"            [[RFC1907](#)]

- For the "minimum-secure" posture:

the <restricted> MIB view is the following subtree.

"internet"          [[RFC1902](#)]

- Access rights to allow:

- read-notify access for LoS "noAuth" on behalf of security entities that belong to the group "public" to the <restricted> MIB view in the context with contextName "".
- read-write-notify access for LoS "auth" on behalf of security entities that belong to the group "public" to the <all> MIB view in the context with contextName "".
- if privacy is supported,  
read-write-notify access for LoS "auth" on behalf of security entities that belong to the group "public" to the <all> MIB view in the context with contextName "".

-- Editor's note:

    If we find it useful (I do) then I will also work out  
    the entries in the viewTable and viewSubtreeFamilyTable  
    so that we have the above views defined.

-- End Editor's note



## Table of Contents

<a href="#">0.1</a>	Issues	2
<a href="#">0.2</a>	Change Log	2
<a href="#">1.</a>	Introduction	3
<a href="#">1.2</a>	Access Control	3
<a href="#">1.3</a>	Local Configuration Datastore	3
<a href="#">2.</a>	Elements of the Model	5
<a href="#">2.1</a>	Groups	5
<a href="#">2.2</a>	Level of Security (LoS)	5
<a href="#">2.3</a>	Contexts	5
<a href="#">2.4</a>	Access Policy	5
<a href="#">3.</a>	Elements of Procedure	7
<a href="#">3.1</a>	Processing the is_access_allowed service request	7
<a href="#">4.</a>	Definitions	9
<a href="#">5.</a>	Security Considerations	26
<a href="#">5.1</a>	Recommended Practices	26
<a href="#">5.2</a>	Defining Groups	26
<a href="#">5.3</a>	Conformance	26
<a href="#">6.</a>	Editor's Addresses	27
<a href="#">7.</a>	Acknowledgements	27
<a href="#">8.</a>	References	28
<a href="#">A.1.</a>	Installation Parameters	29

