Network Working Group  K. McCloghrie
Request for Comments: 1447  Hughes LAN Systems
                        J. Galvin
                        Trusted Information Systems
                        April 1993

          Party MIB
          for version 2 of the
          Simple Network Management Protocol (SNMPv2)

Status of this Memo

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

Table of Contents

1 Introduction ........................................... 2
1.1 A Note on Terminology .................................. 2
2 Definitions ............................................ 3
3.1 Textual Conventions .................................. 4
3.2 Administrative Assignments ............................ 7
3.2.1 Initial Party and Context Identifiers .............. 8
3.3 Object Assignments ................................... 16
3.4 The SNMPv2 Party Database Group ..................... 16
3.5 The SNMPv2 Contexts Database Group ................. 29
3.5 The SNMPv2 Access Privileges Database Group ....... 36
3.6 The MIB View Database Group ........................... 40
3.7 Conformance Information ............................. 45
3.7.1 Compliance Statements ............................. 45
3.7.2 Units of Conformance .............................. 47
3 Acknowledgments ....................................... 48
4 References ............................................ 49
5 Security Considerations ............................... 50
6 Authors’ Addresses ..................................... 50
1. Introduction

A network management system contains: several (potentially many) nodes, each with a processing entity, termed an agent, which has access to management instrumentation; at least one management station; and, a management protocol, used to convey management information between the agents and management stations. Operations of the protocol are carried out under an administrative framework which defines both authentication and authorization policies.

Network management stations execute management applications which monitor and control network elements. Network elements are devices such as hosts, routers, terminal servers, etc., which are monitored and controlled through access to their management information.

Management information is viewed as a collection of managed objects, residing in a virtual information store, termed the Management Information Base (MIB). Collections of related objects are defined in MIB modules. These modules are written using a subset of OSI’s Abstract Syntax Notation One (ASN.1) [1], termed the Structure of Management Information (SMI) [2].

The Administrative Model for SNMPv2 document [3] defines the properties associated with SNMPv2 parties, SNMPv2 contexts, and access control policies. It is the purpose of this document, the Party MIB for SNMPv2, to define managed objects which correspond to these properties.

1.1. A Note on Terminology

For the purpose of exposition, the original Internet-standard Network Management Framework, as described in RFCs 1155, 1157, and 1212, is termed the SNMP version 1 framework (SNMPv1). The current framework is termed the SNMP version 2 framework (SNMPv2).
2. Definitions

SNMPv2-PARTY-MIB DEFINITIONS ::= BEGIN

IMPORTS
   MODULE-IDENTITY, OBJECT-TYPE, snmpModules,
   UInteger32
   FROM SNMPv2-SMI
   TEXTUAL-CONVENTION, RowStatus, TruthValue
   FROM SNMPv2-TC
   MODULE-COMPLIANCE, OBJECT-GROUP
   FROM SNMPv2-CONF;

partyMIB MODULE-IDENTITY
   LAST-UPDATED "9304010000Z"
   ORGANIZATION "IETF SNMP Security Working Group"
   CONTACT-INFO
      " Keith McCloghrie
       Postal: Hughes LAN Systems
               1225 Charleston Road
               Mountain View, CA 94043 US
               Tel: +1 415 966 7934
               Fax: +1 415 960 3738
               E-mail: kzm@hls.com"
   DESCRIPTION
      "The MIB module describing SNMPv2 parties."
   ::= { snmpModules 3 }
-- textual conventions

Party ::= TEXTUAL-CONVENTION
   STATUS       current
   DESCRIPTION
      "Denotes a SNMPv2 party identifier.

      Note that agents may impose implementation limitations on the length of OIDs used to identify Parties. As such, management stations creating new parties should be aware that using an excessively long OID may result in the agent refusing to perform the set operation and instead returning the appropriate error response, e.g., noCreation."
   SYNTAX       OBJECT IDENTIFIER

TAddress ::= TEXTUAL-CONVENTION
   STATUS       current
   DESCRIPTION
      "Denotes a transport service address.

      For snmpUDPDomain, a TAddress is 6 octets long, the initial 4 octets containing the IP-address in network-byte order and the last 2 containing the UDP port in network-byte order. Consult [5] for further information on snmpUDPDomain."
   SYNTAX       OCTET STRING
Clock ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"A party’s authentication clock - a non-negative integer which is incremented as specified/allowed by the party’s Authentication Protocol.

For noAuth, a party’s authentication clock is unused and its value is undefined.

For v2md5AuthProtocol, a party’s authentication clock is a relative clock with 1-second granularity."
SYNTAX UIInteger32

Context ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"Denotes a SNMPv2 context identifier.

Note that agents may impose implementation limitations on the length of OIDs used to identify Contexts. As such, management stations creating new contexts should be aware that using an excessively long OID may result in the agent refusing to perform the set operation and instead returning the appropriate error response, e.g., noCreation."
SYNTAX OBJECT IDENTIFIER
StorageType ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION "Describes the memory realization of a conceptual row. A row which is volatile(2) is lost upon reboot. A row which is nonVolatile(3) is backed up by stable storage. A row which is permanent(4) cannot be changed nor deleted."
SYNTAX INTEGER {
other(1), -- eh?
volatile(2), -- e.g., in RAM
nonVolatile(3), -- e.g., in NVRAM
permanent(4) -- e.g., in ROM
}

-- administrative assignments

partyAdmin OBJECT IDENTIFIER ::= { partyMIB 1 }

-- definitions of security protocols

partyProtocols OBJECT IDENTIFIER ::= { partyAdmin 1 }

-- the protocol without authentication
noAuth OBJECT IDENTIFIER ::= { partyProtocols 1 }

-- the protocol without privacy
noPriv OBJECT IDENTIFIER ::= { partyProtocols 2 }

-- the DES Privacy Protocol [4]
desPrivProtocol OBJECT IDENTIFIER ::= { partyProtocols 3 }

-- the MD5 Authentication Protocol [4]
v2md5AuthProtocol OBJECT IDENTIFIER ::= { partyProtocols 4 }

-- definitions of temporal domains

temporalDomains OBJECT IDENTIFIER ::= { partyAdmin 2 }

-- this temporal domain refers to management information
-- at the current time
currentTime OBJECT IDENTIFIER ::= { temporalDomains 1 }

-- this temporal domain refers to management information
-- upon the next re-initialization of the managed device
restartTime OBJECT IDENTIFIER ::= { temporalDomains 2 }

-- the temporal domain { cacheTime N } refers to management
-- information that is cached and guaranteed to be at most
-- N seconds old
cacheTime OBJECT IDENTIFIER ::= { temporalDomains 3 }
-- Definition of Initial Party and Context Identifiers

-- When devices are installed, they need to be configured
-- with an initial set of SNMPv2 parties and contexts. The
-- configuration of SNMPv2 parties and contexts requires (among
-- other things) the assignment of several OBJECT IDENTIFIERs.
-- Any local network administration can obtain the delegated
-- authority necessary to assign its own OBJECT IDENTIFIERs.
-- However, to provide for those administrations who have not
-- obtained the necessary authority, this document allocates a
-- branch of the naming tree for use with the following
-- conventions.

initialPartyId OBJECT IDENTIFIER ::= { partyAdmin 3 }

initialContextId
  OBJECT IDENTIFIER ::= { partyAdmin 4 }

-- Note these are identified as "initial" party and context
-- identifiers since these allow secure SNMPv2 communication
-- to proceed, thereby allowing further SNMPv2 parties to be
-- configured through use of the SNMPv2 itself.

-- The following definitions identify a party identifier, and
-- specify the initial values of various object instances
-- indexed by that identifier. In addition, the SNMPv2
-- context, access control policy, and MIB view information
-- assigned, by convention, are identified.
-- Party Identifiers for use as initial SNMPv2 parties
-- at IP address a.b.c.d

-- Note that for all OBJECT IDENTIFIERS assigned under
-- initialPartyId, the four sub-identifiers immediately
-- following initialPartyId represent the four octets of
-- an IP address. Initial party identifiers for other address
-- families are assigned under a different OBJECT IDENTIFIER,
-- as defined elsewhere.

-- Devices which support SNMPv2 as entities acting in an
-- agent role, and accessed via the snmpUDPDomain transport
-- domain, are required to be configured with the appropriate
-- set of the following as implicit assignments as and when
-- they are configured with an IP address. The appropriate
-- set is all those applicable to the authentication and
-- privacy protocols supported by the device.
-- a noAuth/noPriv party which executes at the agent
-- partyIdentity = { initialPartyId a b c d 1 }
-- partyIndex = 1
-- partyTDomain = snmpUDPDomain
-- partyTAddress = a.b.c.d, 161
-- partyLocal = true (in agent’s database)
-- partyAuthProtocol = noAuth
-- partyAuthClock = 0
-- partyAuthPrivate = ’’H (the empty string)
-- partyAuthPublic = ’’H (the empty string)
-- partyAuthLifetime = 0
-- partyPrivProtocol = noPriv
-- partyPrivPrivate = ’’H (the empty string)
-- partyPrivPublic = ’’H (the empty string)

-- a noAuth/noPriv party which executes at a manager
-- partyIdentity = { initialPartyId a b c d 2 }
-- partyIndex = 2
-- partyTDomain = snmpUDPDomain
-- partyTAddress = assigned by local administration
-- partyLocal = false (in agent’s database)
-- partyAuthProtocol = noAuth
-- partyAuthClock = 0
-- partyAuthPrivate = ’’H (the empty string)
-- partyAuthPublic = ’’H (the empty string)
-- partyAuthLifetime = 0
-- partyPrivProtocol = noPriv
-- partyPrivPrivate = ’’H (the empty string)
-- partyPrivPublic = ’’H (the empty string)
a md5Auth/noPriv party which executes at the agent

-- partyIdentity = { initialPartyId a b c d 3 }
-- partyIndex = 3
-- partyTDomain = snmpUDPDomain
-- partyTAddress = a.b.c.d, 161
-- partyLocal = true (in agent’s database)
-- partyAuthProtocol = v2md5AuthProtocol
-- partyAuthClock = 0
-- partyAuthPrivate = assigned by local administration
-- partyAuthPublic = ’’H (the empty string)
-- partyAuthLifetime = 300
-- partyPrivProtocol = noPriv
-- partyPrivPrivate = ’’H (the empty string)
-- partyPrivPublic = ’’H (the empty string)

a md5Auth/noPriv party which executes at a manager

-- partyIdentity = { initialPartyId a b c d 4 }
-- partyIndex = 4
-- partyTDomain = snmpUDPDomain
-- partyTAddress = assigned by local administration
-- partyLocal = false (in agent’s database)
-- partyAuthProtocol = v2md5AuthProtocol
-- partyAuthClock = 0
-- partyAuthPrivate = assigned by local administration
-- partyAuthPublic = ’’H (the empty string)
-- partyAuthLifetime = 300
-- partyPrivProtocol = noPriv
-- partyPrivPrivate = ’’H (the empty string)
-- partyPrivPublic = ’’H (the empty string)
-- a md5Auth/desPriv party which executes at the agent
-- partyIdentity = { initialPartyId a b c d 5 }
-- partyIndex = 5
-- partyTDomain = snmpUDPDomain
-- partyTAddress = a.b.c.d, 161
-- partyLocal = true (in agent’s database)
-- partyAuthProtocol = v2md5AuthProtocol
-- partyAuthClock = 0
-- partyAuthPrivate = assigned by local administration
-- partyAuthPublic = ‘’H (the empty string)
-- partyAuthLifetime = 300
-- partyPrivProtocol = desPrivProtocol
-- partyPrivPrivate = assigned by local administration
-- partyPrivPublic = ‘’H (the empty string)

-- a md5Auth/desPriv party which executes at a manager
-- partyIdentity = { initialPartyId a b c d 6 }
-- partyIndex = 6
-- partyTDomain = snmpUDPDomain
-- partyTAddress = assigned by local administration
-- partyLocal = false (in agent’s database)
-- partyAuthProtocol = v2md5AuthProtocol
-- partyAuthClock = 0
-- partyAuthPrivate = assigned by local administration
-- partyAuthPublic = ‘’H (the empty string)
-- partyAuthLifetime = 300
-- partyPrivProtocol = desPrivProtocol
-- partyPrivPrivate = assigned by local administration
-- partyPrivPublic = ‘’H (the empty string)
-- the initial SNMPv2 contexts assigned, by convention, are:

-- contextIdentity          = { initialContextId a b c d 1 }
-- contextIndex             = 1
-- contextLocal             = true (in agent’s database)
-- contextViewIndex         = 1
-- contextLocalEntity       = "H"  (the empty string)
-- contextLocalTime         = currentTime
-- contextProxyDstParty     = { 0 0 }
-- contextProxySrcParty     = { 0 0 }
-- contextProxyContext      = { 0 0 }

-- contextIdentity          = { initialContextId a b c d 2 }
-- contextIndex             = 2
-- contextLocal             = true (in agent’s database)
-- contextViewIndex         = 2
-- contextLocalEntity       = "H"  (the empty string)
-- contextLocalTime         = currentTime
-- contextProxyDstParty     = { 0 0 }
-- contextProxySrcParty     = { 0 0 }
-- contextProxyContext      = { 0 0 }
-- The initial access control policy assigned, by
-- convention, is:

-- aclTarget = 1
-- aclSubject = 2
-- aclResources = 1
-- aclPrivileges = 35 (Get, Get-Next & Get-Bulk)

-- aclTarget = 2
-- aclSubject = 1
-- aclResources = 1
-- aclPrivileges = 132 (Response & SNMPv2-Trap)

-- aclTarget = 3
-- aclSubject = 4
-- aclResources = 2
-- aclPrivileges = 43 (Get, Get-Next, Set & Get-Bulk)

-- aclTarget = 4
-- aclSubject = 3
-- aclResources = 2
-- aclPrivileges = 4 (Response)

-- aclTarget = 5
-- aclSubject = 6
-- aclResources = 2
-- aclPrivileges = 43 (Get, Get-Next, Set & Get-Bulk)

-- aclTarget = 6
-- aclSubject = 5
-- aclResources = 2
-- aclPrivileges = 4 (Response)

-- Note that the initial context and access control
-- information assigned above, by default, to the
-- md5Auth/desPriv parties are identical to those assigned to
-- the md5Auth/noPriv parties. However, each administration
-- may choose to have different authorization policies,
-- depending on whether privacy is used.
-- The initial MIB views assigned, by convention, are:

-- viewIndex  = 1
-- viewSubtree = system
-- viewMask = ''H
-- viewType = included

-- viewIndex  = 1
-- viewSubtree = snmpStats
-- viewMask = ''H
-- viewType = included

-- viewIndex  = 1
-- viewSubtree = snmpParties
-- viewMask = ''H
-- viewType = included

-- viewIndex  = 2
-- viewSubtree = internet
-- viewMask = ''H
-- viewType = included

-- Note that full access to the partyTable, contextTable,
-- aclTable, and viewTable gives a manager the ability to
-- configure any parties with any/all capabilities (the
-- equivalent of "root" access). A lesser manager can be
-- given access only to the partyTable so that it can
-- maintain its own parties, but not increase/decrease
-- their capabilities. Such a lesser manager can also
-- create new parties but they are of no use to it.
-- object assignments

partyMIBObjects
   OBJECT IDENTIFIER ::= { partyMIB 2 }

-- the SNMPv2 party database group

snmpParties   OBJECT IDENTIFIER ::= { partyMIBObjects 1 }

partyTable OBJECT-TYPE
   SYNTAX      SEQUENCE OF PartyEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION  
                "The SNMPv2 Party database."
   := { snmpParties 1 }

partyEntry OBJECT-TYPE
   SYNTAX      PartyEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION  
                "Locally held information about a particular
                SNMPv2 party."
   INDEX       { IMPLIED partyIdentity }
   ::= { partyTable 1 }
PartyEntry ::= SEQUENCE {
  partyIdentity Party,
  partyIndex INTEGER,
  partyTDomain OBJECT IDENTIFIER,
  partyTAddress TAddress,
  partyMaxMessageSize INTEGER,
  partyLocal TruthValue,
  partyAuthProtocol OBJECT IDENTIFIER,
  partyAuthClock Clock,
  partyAuthPrivate OCTET STRING,
  partyAuthPublic OCTET STRING,
  partyAuthLifetime INTEGER,
  partyPrivProtocol OBJECT IDENTIFIER,
  partyPrivPrivate OCTET STRING,
  partyPrivPublic OCTET STRING,
  partyCloneFrom Party,
  partyStorageType StorageType,
  partyStatus RowStatus
}

partyIdentity OBJECT-TYPE
SYNTAX Party
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
  "A party identifier uniquely identifying a particular SNMPv2 party."
 ::= { partyEntry 1 }

partyIndex OBJECT-TYPE
SYNTAX INTEGER (1..65535)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
  "A unique value for each SNMPv2 party. The value for each SNMPv2 party must remain constant at least from one re-initialization of the entity’s network management system to the next re-initialization."
 ::= { partyEntry 2 }
partyTDomain OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
MAX-ACCESS read-create
STATUS current
DESCRIPTION "Indicates the kind of transport service by which
the party receives network management traffic."
DEFVAL { snmpUDPDoman }
::= { partyEntry 3 }

partyTAddress OBJECT-TYPE
SYNTAX TAddress
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The transport service address by which the party
receives network management traffic, formatted
according to the corresponding value of
partyTDomain. For snmpUDPDoman, partyTAddress is
formatted as a 4-octet IP Address concatenated
with a 2-octet UDP port number."
DEFVAL { ‘000000000000’H }
::= { partyEntry 4 }

partyMaxMessageSize OBJECT-TYPE
SYNTAX INTEGER (484..65507)
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The maximum length in octets of a SNMPv2 message
which this party will accept. For parties which
execute at an agent, the agent initializes this
object to the maximum length supported by the
agent, and does not let the object be set to any
larger value. For parties which do not execute at
the agent, the agent must allow the manager to set
this object to any legal value, even if it is
larger than the agent can generate."
DEFVAL { 484 }
::= { partyEntry 5 }
partyLocal OBJECT-TYPE
SYNTAX     TruthValue
MAX-ACCESS read-create
STATUS     current
DESCRIPTION "An indication of whether this party executes at this SNMPv2 entity. If this object has a value of true(1), then the SNMPv2 entity will listen for SNMPv2 messages on the partyTAddress associated with this party. If this object has the value false(2), then the SNMPv2 entity will not listen for SNMPv2 messages on the partyTAddress associated with this party."
DEFVAL     { false }
::= { partyEntry 6 }

partyAuthProtocol OBJECT-TYPE
SYNTAX     OBJECT IDENTIFIER
MAX-ACCESS read-create
STATUS     current
DESCRIPTION "The authentication protocol by which all messages generated by the party are authenticated as to origin and integrity. The value noAuth signifies that messages generated by the party are not authenticated.

Once an instance of this object is created, its value can not be changed."
DEFVAL     { v2md5AuthProtocol }
::= { partyEntry 7 }
partyAuthClock  OBJECT-TYPE
   SYNTAX     Clock
   MAX-ACCESS read-create
   STATUS     current
   DESCRIPTION
      "The authentication clock which represents the
       local notion of the current time specific to the
       party.  This value must not be decremented unless
       the party’s private authentication key is changed
       simultaneously."
   DEFVAL     { 0 }
   ::= { partyEntry 8 }
partyAuthPrivate OBJECT-TYPE
SYNTAX OCTET STRING
-- for v2md5AuthProtocol: (SIZE (16))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"An encoding of the party’s private authentication key which may be needed to support the authentication protocol. Although the value of this variable may be altered by a management operation (e.g., a SNMPv2 Set-Request), its value can never be retrieved by a management operation: when read, the value of this variable is the zero length OCTET STRING.

The private authentication key is NOT directly represented by the value of this variable, but rather it is represented according to an encoding. This encoding is the bitwise exclusive-OR of the old key with the new key, i.e., of the old private authentication key (prior to the alteration) with the new private authentication key (after the alteration). Thus, when processing a received protocol Set operation, the new private authentication key is obtained from the value of this variable as the result of a bitwise exclusive-OR of the variable’s value and the old private authentication key. In calculating the exclusive-OR, if the old key is shorter than the new key, zero-valued padding is appended to the old key. If no value for the old key exists, a zero-length OCTET STRING is used in the calculation."
DEFVAL { ''H } -- the empty string
::= { partyEntry 9 }
partyAuthPublic OBJECT-TYPE
SYNTAX OCTET STRING
-- for v2md5AuthProtocol: (SIZE (0..16))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"A publically-readable value for the party.

Depending on the party’s authentication protocol, this value may be needed to support the party’s authentication protocol. Alternatively, it may be used by a manager during the procedure for altering secret information about a party. (For example, by altering the value of an instance of this object in the same SNMPv2 Set-Request used to update an instance of partyAuthPrivate, a subsequent Get-Request can determine if the Set-Request was successful in the event that no response to the Set-Request is received, see [4].)

The length of the value is dependent on the party’s authentication protocol. If not used by the authentication protocol, it is recommended that agents support values of any length up to and including the length of the corresponding partyAuthPrivate object."
DEFVAL { ''H } -- the empty string
::= { partyEntry 10 }
partyAuthLifetime OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
UNITS "seconds"
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The lifetime (in units of seconds) which represents an administrative upper bound on acceptable delivery delay for protocol messages generated by the party.

Once an instance of this object is created, its value can not be changed."
DEFVAL { 300 }
::= { partyEntry 11 }

partyPrivProtocol OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The privacy protocol by which all protocol messages received by the party are protected from disclosure. The value noPriv signifies that messages received by the party are not protected.

Once an instance of this object is created, its value can not be changed."
DEFVAL { noPriv }
::= { partyEntry 12 }
partyPrivPrivate OBJECT-TYPE
SYNTAX OCTET STRING
-- for desPrivProtocol: (SIZE (16))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"An encoding of the party’s private encryption key which may be needed to support the privacy protocol. Although the value of this variable may be altered by a management operation (e.g., a SNMPv2 Set-Request), its value can never be retrieved by a management operation: when read, the value of this variable is the zero length OCTET STRING.

The private encryption key is NOT directly represented by the value of this variable, but rather it is represented according to an encoding. This encoding is the bitwise exclusive-OR of the old key with the new key, i.e., of the old private encryption key (prior to the alteration) with the new private encryption key (after the alteration). Thus, when processing a received protocol Set operation, the new private encryption key is obtained from the value of this variable as the result of a bitwise exclusive-OR of the variable’s value and the old private encryption key. In calculating the exclusive-OR, if the old key is shorter than the new key, zero-valued padding is appended to the old key. If no value for the old key exists, a zero-length OCTET STRING is used in the calculation."

DEFVAL { ''H } -- the empty string
::= { partyEntry 13 }
partyPrivPublic OBJECT-TYPE
SYNTAX OCTET STRING
-- for desPrivProtocol: (SIZE (0..16))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"A publically-readable value for the party.

Depending on the party’s privacy protocol, this value may be needed to support the party’s privacy protocol. Alternatively, it may be used by a manager as a part of its procedure for altering secret information about a party. (For example, by altering the value of an instance of this object in the same SNMPv2 Set-Request used to update an instance of partyPrivPrivate, a subsequent Get-Request can determine if the Set-Request was successful in the event that no response to the Set-Request is received, see [4].)

The length of the value is dependent on the party’s privacy protocol. If not used by the privacy protocol, it is recommended that agents support values of any length up to and including the length of the corresponding partyPrivPrivate object."

DEFVAL { ''H } -- the empty string
::= { partyEntry 14 }
partyCloneFrom OBJECT-TYPE
SYNTAX      Party
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
 "The identity of a party to clone authentication and privacy parameters from. When read, the value ( 0 0 ) is returned.

This value must be written exactly once, when the associated instance of partyStatus either does not exist or has the value ‘notReady’. When written, the value identifies a party, the cloning party, whose status column has the value ‘active’. The cloning party is used in two ways.

One, if instances of the following objects do not exist for the party being created, then they are created with values identical to those of the corresponding objects for the cloning party:

   partyAuthProtocol
   partyAuthPublic
   partyAuthLifetime
   partyPrivProtocol
   partyPrivPublic

Two, instances of the following objects are updated using the corresponding values of the cloning party:

   partyAuthPrivate
   partyPrivPrivate

   (e.g., the value of the cloning party’s instance of the partyAuthPrivate object is XOR’d with the value of the partyAuthPrivate instances of the party being created.)"
::= { partyEntry 15 }
partyStorageType OBJECT-TYPE
SYNTAX      StorageType
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
 "The storage type for this conceptual row in the
partyTable."
DEFVAL      { nonVolatile }
 ::= { partyEntry 16 }

partyStatus OBJECT-TYPE
SYNTAX      RowStatus
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
 "The status of this conceptual row in the
partyTable.

A party is not qualified for activation until
instances of all columns of its partyEntry row
have an appropriate value. In particular:

A value must be written to the Party’s
partyCloneFrom object.

If the Party’s partyAuthProtocol object has the
value md5AuthProtocol, then the corresponding
instance of partyAuthPrivate must contain a
secret of the appropriate length. Further, at
least one management protocol set operation
updating the value of the party’s
partyAuthPrivate object must be successfully
processed, before the partyAuthPrivate column is
considered appropriately configured.

If the Party’s partyPrivProtocol object has the
value desPrivProtocol, then the corresponding
instance of partyPrivPrivate must contain a
secret of the appropriate length. Further, at
least one management protocol set operation
updating the value of the party’s
partyPrivPrivate object must be successfully
processed, before the partyPrivPrivate column is
considered appropriately configured.
Until instances of all corresponding columns are appropriately configured, the value of the corresponding instance of the partyStatus column is 'notReady'.

::= { partyEntry 17 }
--- the SNMPv2 contexts database group

snmpContexts OBJECT IDENTIFIER ::= { partyMIBObjects 2 }

contextTable OBJECT-TYPE
SYNTAX    SEQUENCE OF ContextEntry
MAX-ACCESS not-accessible
STATUS    current
DESCRIPTION
   "The SNMPv2 Context database."
::= { snmpContexts 1 }

contextEntry OBJECT-TYPE
SYNTAX    ContextEntry
MAX-ACCESS not-accessible
STATUS    current
DESCRIPTION
   "Locally held information about a particular
   SNMPv2 context."
INDEX     { IMPLIED contextIdentity }
::= { contextTable 1 }

ContextEntry ::= SEQUENCE {
   contextIdentity         Context,
   contextIndex            INTEGER,
   contextLocal            TruthValue,
   contextViewIndex        INTEGER,
   contextLocalEntity      OCTET STRING,
   contextLocalTime        OBJECT IDENTIFIER,
   contextProxyDstParty    Party,
   contextProxySrcParty    Party,
   contextProxyContext     OBJECT IDENTIFIER,
   contextStorageType      StorageType,
   contextStatus           RowStatus
}

Galvin & McCloghrie
contextIdentity OBJECT-TYPE
    SYNTAX    Context
    MAX-ACCESS not-accessible
    STATUS    current
    DESCRIPTION
        "A context identifier uniquely identifying a
         particular SNMPv2 context."
    ::= { contextEntry 1 }

contextIndex OBJECT-TYPE
    SYNTAX    INTEGER (1..65535)
    MAX-ACCESS read-only
    STATUS    current
    DESCRIPTION
        "A unique value for each SNMPv2 context.  The
         value for each SNMPv2 context must remain constant
         at least from one re-initialization of the
         entity’s network management system to the next
         re-initialization."
    ::= { contextEntry 2 }

contextLocal OBJECT-TYPE
    SYNTAX    TruthValue
    MAX-ACCESS read-create
    STATUS    current
    DESCRIPTION
        "An indication of whether this context is realized
         by this SNMPv2 entity."
    DEFVAL    { true }
    ::= { contextEntry 3 }
contextViewIndex OBJECT-TYPE
SYNTAX INTEGER (0..65535)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"If the value of an instance of this object is zero, then this corresponding conceptual row in the contextTable refers to a SNMPv2 context which identifies a proxy relationship; the values of the corresponding instances of the contextProxyDstParty, contextProxySrcParty, and contextProxyContext objects provide further information on the proxy relationship.

Otherwise, if the value of an instance of this object is greater than zero, then this corresponding conceptual row in the contextTable refers to a SNMPv2 context which identifies a MIB view of a locally accessible entity; the value of the instance identifies the particular MIB view which has the same value of viewIndex; and the value of the corresponding instances of the contextLocalEntity and contextLocalTime objects provide further information on the local entity and its temporal domain."

::= { contextEntry 4 }
contextLocalEntity OBJECT-TYPE
SYNTAX OCTET STRING
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"If the value of the corresponding instance of the contextViewIndex is greater than zero, then the value of an instance of this object identifies the local entity whose management information is in the SNMPv2 context’s MIB view. The empty string indicates that the MIB view contains the SNMPv2 entity’s own local management information; otherwise, a non-empty string indicates that the MIB view contains management information of some other local entity, e.g., ‘Repeater1’.

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

contextLocalTime OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"If the value of the corresponding instance of the contextViewIndex is greater than zero, then the value of an instance of this object identifies the temporal context of the management information in the MIB view."

DEFVAL { currentTime }
::= { contextEntry 6 }
contextProxyDstParty OBJECT-TYPE
SYNTAX Party
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"If the value of the corresponding instance of the contextViewIndex is equal to zero, then the value of an instance of this object identifies a SNMPv2 party which is the proxy destination of a proxy relationship.

If the value of the corresponding instance of the contextViewIndex is greater than zero, then the value of an instance of this object is { 0 0 }."  
::= { contextEntry 7 }

contextProxySrcParty OBJECT-TYPE
SYNTAX Party
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"If the value of the corresponding instance of the contextViewIndex is equal to zero, then the value of an instance of this object identifies a SNMPv2 party which is the proxy source of a proxy relationship.

Interpretation of an instance of this object depends upon the value of the transport domain associated with the SNMPv2 party used as the proxy destination in this proxy relationship.

If the value of the corresponding instance of the contextViewIndex is greater than zero, then the value of an instance of this object is { 0 0 }."  
::= { contextEntry 8 }
contextProxyContext OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"If the value of the corresponding instance of the contextViewIndex is equal to zero, then the value of an instance of this object identifies the context of a proxy relationship.

Interpretation of an instance of this object depends upon the value of the transport domain associated with the SNMPv2 party used as the proxy destination in this proxy relationship.

If the value of the corresponding instance of the contextViewIndex is greater than zero, then the value of an instance of this object is { 0 0 }." ::= { contextEntry 9 }

currentStorageType OBJECT-TYPE
SYNTAX StorageType
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The storage type for this conceptual row in the contextTable." DEFVAL { nonVolatile }
 ::= { contextEntry 10 }
contextStatus OBJECT-TYPE
SYNTAX RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The status of this conceptual row in the contextTable.

A context is not qualified for activation until instances of all corresponding columns have the appropriate value. In particular, if the context’s contextViewIndex is greater than zero, then the viewStatus column of the associated conceptual row(s) in the viewTable must have the value ‘active’. Until instances of all corresponding columns are appropriately configured, the value of the corresponding instance of the contextStatus column is ‘notReady’.

 ::= { contextEntry 11 }
-- the SNMPv2 access privileges database group

snmpAccess OBJECT IDENTIFIER ::= { partyMIBObjects 3 }

aclTable OBJECT-TYPE
SYNTAX Sequence OF AclEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "The access privileges database."
::= { snmpAccess 1 }

aclEntry OBJECT-TYPE
SYNTAX AclEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "The access privileges for a particular subject SNMPv2 party when asking a particular target SNMPv2 party to access a particular SNMPv2 context."
INDEX { aclTarget, aclSubject, aclResources }
::= { aclTable 1 }

AclEntry ::= SEQUENCE {
    aclTarget        INTEGER,
    aclSubject       INTEGER,
    aclResources     INTEGER,
    aclPrivileges    INTEGER,
    aclStorageType   StorageType,
    aclStatus        RowStatus
}
aclTarget OBJECT-TYPE
SYNTAX INTEGER (1..65535)
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"The value of an instance of this object
identifies a SNMPv2 party which is the target of
an access control policy, and has the same value
as the instance of the partyIndex object for that
party."
 ::= { aclEntry 1 }

aclSubject OBJECT-TYPE
SYNTAX INTEGER (1..65535)
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"The value of an instance of this object
identifies a SNMPv2 party which is the subject of
an access control policy, and has the same value
as the instance of the partyIndex object for that
SNMPv2 party."
 ::= { aclEntry 2 }

aclResources OBJECT-TYPE
SYNTAX INTEGER (1..65535)
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"The value of an instance of this object
identifies a SNMPv2 context in an access control
policy, and has the same value as the instance of
the contextIndex object for that SNMPv2 context."
 ::= { aclEntry 3 }
aclPrivileges OBJECT-TYPE
SYNTAX      INTEGER (0..255)
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION "The access privileges which govern what
management operations a particular target party
may perform with respect to a particular SNMPv2
context when requested by a particular subject
party. These privileges are specified as a sum of
values, where each value specifies a SNMPv2 PDU
type by which the subject party may request a
permitted operation. The value for a particular
PDU type is computed as 2 raised to the value of
the ASN.1 context-specific tag for the appropriate
SNMPv2 PDU type. The values (for the tags defined
in [5]) are defined in [3] as:

Get         :   1
GetNext     :   2
Response    :   4
Set         :   8
unused      :  16
GetBulk     :  32
Inform      :  64
SNMPv2-Trap : 128

The null set is represented by the value zero."
DEFVAL      { 35 }      -- Get, Get-Next & Get-Bulk
::= { aclEntry 4 }

aclStorageType OBJECT-TYPE
SYNTAX      StorageType
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION "The storage type for this conceptual row in the
aclTable."
DEFVAL      { nonVolatile }
 ::= { aclEntry 5 }
aclStatus OBJECT-TYPE
SYNTAX    RowStatus
MAX-ACCESS read-create
STATUS    current
DESCRIPTION
  "The status of this conceptual row in the
    aclTable."
 ::= { aclEntry 6 }
-- the MIB view database group

snmpViews OBJECT IDENTIFIER ::= { partyMIBObjects 4 }

viewTable OBJECT-TYPE
SYNTAX SEQUENCE OF ViewEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

"Locally held information about the MIB views known to this SNMPv2 entity.

Each SNMPv2 context which is locally accessible has a single MIB view which 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 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 viewType in the entry whose value of viewSubtree has the most sub-identifiers. If multiple entries match and have the same number of sub-identifiers, then the lexicographically greatest instance of viewType determines the inclusion or exclusion.

An object instance’s OBJECT IDENTIFIER X matches an entry in this table when the number of sub-identifiers in X is at least as many as in the value of viewSubtree for the entry, and each sub-identifier in the value of viewSubtree matches its corresponding sub-identifier in X. Two sub-identifiers match either if the corresponding bit of viewMask is zero (the ‘wild card’ value), or if they are equal.

Due to this ‘wild card’ capability, we introduce
the term, a ‘family’ of view subtrees, to refer to the set of subtrees defined by a particular combination of values of viewSubtree and viewMask. In the case where no ‘wild card’ is defined in viewMask, the family of view subtrees reduces to a single view subtree.

::= { snmpViews 1 }

viewEntry OBJECT-TYPE
SYNTAX       ViewEntry
MAX-ACCESS   not-accessible
STATUS       current
DESCRIPTION  "Information on a particular family of view subtrees included in or excluded from a particular SNMPv2 context’s MIB view.

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 viewTable."

INDEX       { viewIndex, IMPLIED viewSubtree }
::= { viewTable 1 }

ViewEntry ::==
SEQUENCE {
  viewIndex        INTEGER,
  viewSubtree      OBJECT IDENTIFIER,
  viewMask         OCTET STRING,
  viewType         INTEGER,
  viewStorageType  StorageType,
  viewStatus       RowStatus
}

Galvin & McCloghrie                                  [Page 41]
viewIndex OBJECT-TYPE
SYNTAX       INTEGER (1..65535)
MAX-ACCESS   not-accessible
STATUS       current
DESCRIPTION  
"A unique value for each MIB view. The value for 
each MIB view must remain constant at least from 
one re-initialization of the entity’s network 
management system to the next re-initialization."
 ::= { viewEntry 1 }

viewSubtree OBJECT-TYPE
SYNTAX       OBJECT IDENTIFIER
MAX-ACCESS   not-accessible
STATUS       current
DESCRIPTION  
"A MIB subtree."
 ::= { viewEntry 2 }

viewMask 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 viewSubtree, defines a 
family of view subtrees.

Each bit of this bit mask corresponds to a sub-
identifier of viewSubtree, 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 the following criteria are met:

for each sub-identifier of the value of viewSubtree, either:

the i-th bit of viewMask is 0, or

the i-th sub-identifier of X is equal to the i-th sub-identifier of the value of viewSubtree.

If the value of this bit mask is M bits long and there are more than M sub-identifiers in the corresponding instance of viewSubtree, 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 viewSubtree.

DEFVAL { ''H }
 ::= { viewEntry 3 }
viewType OBJECT-TYPE
SYNTAX     INTEGER  {
             included(1),
             excluded(2)
          }
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
 "The status of a particular family of view
subtrees within the particular SNMPv2 context’s
MIB view. The value ‘included(1)’ indicates that
the corresponding instances of viewSubtree and
viewMask define a family of view subtrees included
in the MIB view. The value ‘excluded(2)’
indicates that the corresponding instances of
viewSubtree and viewMask define a family of view
subtrees excluded from the MIB view."
DEFVAL     { included }
 ::= { viewEntry 4 }

viewStorageType OBJECT-TYPE
SYNTAX     StorageType
MAX-ACCESS read-create
STATUS     current
DESCRIPTION
 "The storage type for this conceptual row in the
viewTable."
DEFVAL     { nonVolatile }
 ::= { viewEntry 5 }

viewStatus OBJECT-TYPE
SYNTAX     RowStatus
MAX-ACCESS read-create
STATUS     current
DESCRIPTION
 "The status of this conceptual row in the
viewTable."
 ::= { viewEntry 6 }
-- conformance information

partyMIBConformance
   OBJECT IDENTIFIER ::= { partyMIB 3 }

partyMIBCompliances
   OBJECT IDENTIFIER ::= { partyMIBConformance 1 }

partyMIBGroups
   OBJECT IDENTIFIER ::= { partyMIBConformance 2 }

-- compliance statements

unSecurableCompliance MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION
      "The compliance statement for SNMPv2 entities
      which implement the Party MIB, but do not support
      any authentication or privacy protocols (i.e.,
      only the noAuth and noPriv protocols are
      supported)."
   MODULE -- this module
      MANDATORY-GROUPS { partyMIBGroup }
      ::= { partyMIBCompliances 1 }

partyNoPrivacyCompliance MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION
      "The compliance statement for SNMPv2 entities
      which implement the Party MIB, and support an
      authentication protocol, but do not support any
      privacy protocols (i.e., only the noAuth,
      v2md5AuthProtocol, and noPriv protocols are
      supported)."
   MODULE -- this module
      MANDATORY-GROUPS { partyMIBGroup }
      ::= { partyMIBCompliances 2 }
The compliance statement for SNMPv2 entities which implement the Party MIB, support an authentication protocol, and support a privacy protocol ONLY for the purpose of accessing security parameters.

For all aclTable entries authorizing a subject and/or target SNMPv2 party whose privacy protocol is desPrivProtocol, to be used in accessing an SNMPv2 context, the MIB view for that SNMPv2 context shall include only those objects subordinate to partyMIBObjects, or a subset thereof, e.g.,

```
viewSubtree = { partyMIBObjects }
viewMask    = ''H
viewType    = { included }
```

Any attempt to configure an entry in the partyTable, the contextTable, the aclTable or the viewTable such that a party using the desPrivProtocol would be authorized for use in accessing objects outside of the partyMIBObjects subtree shall result in the appropriate error response (e.g., wrongValue or inconsistentValue).
fullPrivacyCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
"The compliance statement for SNMPv2 entities which implement the Party MIB, support an authentication protocol, and support a privacy protocol without restrictions on its use."

MODULE -- this module
MANDATORY-GROUPS { partyMIBGroup }
::= { partyMIBCompliances 4 }

-- units of conformance

partyMIBGroup OBJECT-GROUP
OBJECTS { partyIndex, partyTDomain, partyTAddress, partyMaxMessageSize, partyLocal, partyAuthProtocol, partyAuthClock, partyAuthPrivate, partyAuthPublic, partyAuthLifetime, partyPrivProtocol, partyPrivPrivate, partyPrivPublic, partyStorageType, partyStatus, partyCloneFrom, contextIndex, contextLocal, contextViewIndex, contextLocalEntity, contextLocalTime, contextStorageType, contextStatus, aclTarget, aclSubject, aclPrivileges, aclStorageType, aclStatus, viewMask, viewType, viewStorageType, viewStatus }
STATUS current
DESCRIPTION
"The collection of objects allowing the description and configuration of SNMPv2 parties. Note that objects which support proxy relationships are not included in this conformance group."

::= { partyMIBGroups 1 }

END
3. Acknowledgments

This document is based, almost entirely, on RFC 1353.
4. References


5. Security Considerations

Security issues are not discussed in this memo.

6. Authors’ Addresses

Keith McCloghrie
Hughes LAN Systems
1225 Charleston Road
Mountain View, CA  94043
US

Phone: +1 415 966 7934
Email: kzm@hls.com

James M. Galvin
Trusted Information Systems, Inc.
3060 Washington Road, Route 97
Glenwood, MD 21738

Phone: +1 301 854-6889
EMail: galvin@tis.com