Internet Draft
<draft-ietf-ips-auth-mib-00.txt>
Expires August 2002

Mark Bakke
Jim Muchow
Cisco Systems

February 2002

Definitions of Managed Objects for User Identity Authentication

1. Status of this Memo

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

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

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

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

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

1.1. Copyright Notice

2. Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP based internets. In particular it defines objects for managing user identities and the names, addresses, and credentials required to authenticate them, for use with various protocols. This draft was motivated by the need for the configuration of authenticated user identities for the iSCSI protocol [ISCSI], but has been extended to be useful for other protocols that have similar requirements. It is important to note

that this MIB provides only the set of identities and the means to authenticate them; it is the responsibility of other MIBs making use of this one to tie them to authorization lists.

3. Acknowledgments

In addition to the authors, several people contributed to the development of this MIB through discussions of authentication, authorization, and access within the iSCSI MIB and security teams, including John Hufferd, Marjorie Krueger, Keith McCloghrie, Tom McSweeney, Steve Senum, and Josh Tseng.

Thanks especially to Keith McCloghrie for serving as advisor for this MIB.

4. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in STD 16, RFC 1155 [RFC1155], STD 16, RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].
- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].
- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is

described in STD 15, RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].

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

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [RFC2570].

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

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

This MIB will be used to configure and/or look at the configuration of user identities and their authentication information. For the purposes of this MIB, a "user" identity does not need to be an actual person; a user can also be a host, an application, a cluster of hosts, or any other identifiable entity that can be authenticated and granted access to a resource.

Most objects in this MIB have a MAX-ACCESS of read-create; the MIB is intended to allow configuration of user identities and their names, addresses, and credentials. MIN-ACCESS for all objects is read-only for those implementations that configure through other means, but require the ability to monitor user identities.

5. Relationship to Other MIBs

The identity authentication MIB does not directly address objects within other MIBs. The identity address objects contain IPv4, IPv6, or other address types, and as such may be indirectly related to objects within the IPv4 MIB [RFC1213, RFC2011] or IPv6 [RFC2465] MIB.

This MIB does not cover authorization. This should generally be done in MIBs that reference identities in this one. It also does not cover login or authentication failure statistics or notifications, as these are all fairly application-specific, and not generic enough to include here.

The user identity objects within this MIB are typically referenced from other MIBs by a RowPointer within that MIB. A MIB containing resources for which it requires a list of authorized user identities may create such a list, with a single RowPointer within each list element pointing to a user identity within this MIB. This is neither required nor restricted by this MIB.

6. Discussion

This MIB structure is intended to allow the configuration of a list of user identities, each with a list of names, addresses, credentials, and certificates which when combined will authenticate that identity.

The authentication MIB is structured around two primary "objects", the authentication instance, and the identity, which serve as containers for the remainder of the objects. This section contains a brief description of the "object" hierarchy and a description of each object, followed by a discussion of the actual SNMP table structure within the objects.

6.1. Identity Authentication MIB Object Model

The top-level object in this structure is the authentication instance, which "contains" all of the other objects. The indexing hierarchy of this MIB looks like:

ipsAuthInstance

```
   -- A distinct authentication entity within the managed system.
   -- Most implementations will have just one of these.
   ipsAuthCertificate
      -- A public key certificate, which can be pointed to by
      -- an ipsAuthIdentity.
   ipsAuthIdentity
      -- A user identity, consisting of a set of identity names,
      -- addresses, and credentials reflected in the following
      -- objects, as well as a RowPointer to an ipsAuthCertificate.
      ipsAuthIdentityName
         -- A name for a user identity.  A name should be globally
         -- unique, and unchanging over time.  Some protocols may
         -- not require this one.
      ipsAuthIdentityAddress
         -- An address range, typically but not necessarily an
         -- IPv4 or IPv6 address range, at which the identity is
         -- allowed to reside.
      ipsAuthCredential
         -- A single credential, such as a CHAP username/password,
```



```
         -- which can ipsAuthenticate the identity.
         ipsAuthCredChap
            -- CHAP-specific attributes for an ipsAuthCredential
         ipsAuthCredSrp
            -- SRP-specific attributes
         ipsAuthCredSpkm
            -- SPKM-specific attributes
         ipsAuthCredKerberos
            -- Kerberos-specific attributes
```

An identity can contain multiple names, addresses, and credentials.

Work - Add some examples here.

Work - need examples showing how this can work on a client and a server, for mutual authentication.

6.2. ipsAuthInstance

The ipsAuthInstanceAttributesTable is the primary table of the authentication MIB. Every other table entry in this MIB includes the index of an ipsAuthInstanceAttributesEntry as its primary index. An

authentication instance is basically a managed set of identities.

Many implementations will include just one authentication instance row in this table. However, there will be cases where multiple rows in this table may be used:

- A large system may be "partitioned" into multiple, distinct virtual systems, perhaps sharing the SNMP agent but not their lists of identities. Each virtual system would have its own authentication instance.

- A set of stackable systems, each with their own set of identities, may be managed by a common SNMP agent. Each individual system would have its own authentication instance.

- Multiple protocols, each with their own set of identities, may exist within a single system and be managed by a single SNMP agent. In this case, each protocol may have its own authentication instance.

6.3. ipsAuthCertificate

The ipsAuthCertAttributesTable contains a list of certificates which can be used to authenticate user identities within the ipsAuthIdentAttributesTable. Rather than copying each certificate

for each of its uses within the identities, the certificates are instead kept in their own list, and may be pointed to by individual identities. This avoids duplication of certificates that may be used by more than one identity, as well as providing a way to keep track of certificates that are not currently in use by any given identity.

The attribute ipsAuthCert contains the binary certificate, in X.509 format [X.509].

WORK - Need to say which attribute matches the identifier.

WORK - some other references that may be helpful (remove if not):

RFC2538 - Storing Certificates in the Domain Name System

RFC2693 - SPKI Certificate Theory

RFC2797 - Certificate Management Messages over CMS

If the implementation making use of this MIB does not require the use of public key certificates, this table will be empty.

6.4. ipsAuthIdentity

The ipsAuthIdentAttributesTable contains one entry for each configured user identity. The identity contains only a description of what the identity is used for; its attributes are all contained in other tables, since they can have multiple values.

Other MIBs containing lists of users authorized to access a particular resource should generally contain a RowPointer to the ipsAuthIdentAttributesEntry which will, if authenticated, be allowed access.

All other table entries make use of the indices to this table as their primary indices.

6.5. ipsAuthIdentityName

The ipsAuthIdentNameAttributesTable contains a list of UTF-8 names, each of which belong to, and may be used to identify, a particular identity in the authIdentity table.

Implementations making use of the authentication MIB may identify their resources by names, addresses, or both. A name is typically a unique (within the required scope), unchanging identifier for a resource. It will normally meet some or all of the requirements for a Uniform Resource Name [RFC1737], although a name in the context of

this MIB does not need to be a URN. Identifiers that typically change over time should generally be placed into the ipsAuthIdentityAddress table; names that have no uniqueness properties should usually be placed into the description attribute for the identity.

An example of an identity name is the iSCSI Name, defined in [ISCSI].

If this table contains no entries associated with a particular user identity, the implementation does not need to check any name paramenters when authenticating that identity. If the table contains multiple entries associated with a particular user identity, the implementation should consider a match with any one of these entries to be valid.

6.6. ipsAuthIdentityAddress

The ipsAuthIdentAddrAttributesTable contains a list of addresses at which the identity may be authenticated. For example, an identity may be allowed access to a resource only from a certain IP address, or only if its address is in a certain range or set of ranges.

Each entry contains a starting and ending address. If a single address is desired in the list, both starting and ending addresses should be identical.

Each entry contains an AddrType attribute. This attribute contains an enumeration registered as an IANA Address Family type [IANA-AF]. Although many implementations will use IPv4 or IPv6 address types for these entries, any IANA-registered type may be used, as long as it makes sense to the application.

Matching any address within any range within the list associated with a particular identity is considered to be a valid match. If no entries are present in this list for a given identity, its address is not checked during authentication.

WORK: Is it better to make ending == starting for a single address, or should the attribute simply not be returned?

WORK: Is there any point to having a netmask if we have a range?

6.7. ipsAuthCredential

The ipsAuthCredentialAttributesTable contains a list of credentials, each of which may authenticate a particular identity.

Each credential contains an authentication method to be used, such as

CHAP [RFC1994], SRP [RFC2945], Kerberos [RFC1510], or SPKM [RFC2025].

This attribute contains an object identifier instead of an enumerated type, allowing other MIBs to add their own authentication methods, without modifying this MIB.

For each entry in this table, there will exist an entry in another table containing its attributes. The table in which to place the entry depends on the AuthMethod attribute:

CHAP If the AuthMethod is set to the CHAP OID, an entry using the same indices as the ipsAuthCredential will exist in the ipsAuthCredChap table, which contains the CHAP username and password expected.

SRP If the AuthMethod is set to the SRP OID, an entry using the same indices as the ipsAuthCredential will exist in the ipsAuthCredSrp table, which contains the SRP username, password verifier, and salt.

SPKM If the AuthMethod is set to the SPKM OID, an entry using the same indices as the ipsAuthCredential will exist in the ipsAuthCredSpkm table, which contains the indices of the authCertificate entries that are expected.

Kerberos If the AuthMethod is set to the Kerberos OID, an entry using the same indices as the ipsAuthCredential will exist in the ipsAuthCredKerberos table. Contents are TBD.

Other If the AuthMethod is set to any OID not defined in this MIB, an entry using the same indices as the ipsAuthCredential entry should be placed in the other MIB that define whatever attributes are needed for that type of credential.

6.8. IP and Other Addresses

WORK: Re-write based on address family types.

The IP addresses in this MIB are represented by two attributes, one of type InetAddressType, and the other of type InetAddress. These are taken from [IPV6MIB], which is an update to [RFC2851] specifying how to support addresses that may be either IPv4 or IPv6.

6.9. Descriptors: Using OIDs in Place of Enumerated Types

Some attributes, particularly the authentication method attribute, would normally require an enumerated type. However, implementations will likely need to add new authentication method types of their own, without extending this MIB. To make this work, the MIB defines a set

of object identities within ipsAuthDescriptors. Each of these object identities is basically an enumerated type.

Attributes that make use of these object identities have a value which is an OID instead of an enumerated type. These OIDs can either indicate the object identities defined in this MIB, or object identities defined elsewhere, such as in an enterprise MIB. Those implementations that add their own authentication methods should also define a corresponding object identity for each of these methods within their own enterprise MIB, and return its OID whenever one of these attributes is using that method.

6.10. Notifications

Monitoring of authentication failures and other notification events are outside the scope of this MIB, as they are generally application-specific. No notifications are provided or required.

7. MIB Definitions

```
IPS-AUTH-MIB DEFINITIONS  ::= BEGIN
-- 2/21-2002 Initial version

-- still some work to do (editor search for "Work")

    IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, OBJECT-IDENTITY, NOTIFICATION-TYPE,
    Unsigned32,
    experimental
    FROM SNMPv2-SMI

    TEXTUAL-CONVENTION, RowStatus,
    AutonomousType
    FROM SNMPv2-TC

    MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
    FROM SNMPv2-CONF

    SnmpAdminString
    FROM SNMP-FRAMEWORK-MIB -- RFC 2571

--  These are from draft-ietf-ops-rfc2851-update-06.txt
--  You will have to work out the details with your own
--  compiler being because they are so new.
    InetAddressType, InetAddress
    FROM INET-ADDRESS-MIB
    ;

ipsAuthModule MODULE-IDENTITY
    LAST-UPDATED  "200202210000Z"
    ORGANIZATION  "IETF IPS Working Group"
    CONTACT-INFO
    "
    Mark Bakke
    Postal: Cisco Systems, Inc
```

6450 Wedgwood Road, Suite 130
Maple Grove, MN
USA 55311

Tel: +1 763-398-1000
Fax: +1 763-398-1001

E-mail: mbakke@cisco.com"


```
    DESCRIPTION
        "The IP Storage Authorization MIB module."

    REVISION "200202210000Z" -- February 21, 2001
    DESCRIPTION
        "Initial revision published as RFC xxxx."

--::= { mib-2 xx } -- to be assigned by IANA.
::= { experimental 99999 } -- in case you want to COMPILE

ipsAuthObjects OBJECT IDENTIFIER ::= { ipsAuthModule 1 }
ipsAuthNotifications OBJECT IDENTIFIER ::= { ipsAuthModule 2 }
ipsAuthConformance OBJECT IDENTIFIER ::= { ipsAuthModule 3 }

-- Textual Conventions

-------------------------------------------------------------------------

ipsAuthDescriptors OBJECT IDENTIFIER ::= { ipsAuthObjects 1 }

ipsAuthMethodTypes OBJECT IDENTIFIER ::= { ipsAuthDescriptors 1 }

ipsAuthMethodNone OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
     "The authoritative identifier when no authentication
     method is used."
    REFERENCE "iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 1 }

ipsAuthMethodSrp OBJECT-IDENTITY
    STATUS      current
```

```
    DESCRIPTION
     "The authoritative identifier when the authentication
     method is SRP."
    REFERENCE "iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 2 }

ipsAuthMethodChap OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
     "The authoritative identifier when the authentication
     method is CHAP."
    REFERENCE "iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 3 }

ipsAuthMethodKrb5 OBJECT-IDENTITY
    STATUS      current
```



```
    DESCRIPTION
     "The authoritative identifier when the authentication
     method is KRB-5."
    REFERENCE "iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 4 }

ipsAuthMethodSpkm1 OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
     "The authoritative identifier when the authentication
     method is SPKM-1."
    REFERENCE "iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 5 }

ipsAuthMethodSpkm2 OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
     "The authoritative identifier when the authentication
     method is SPKM-2."
    REFERENCE "iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 6 }

--------------------------------------------------------------------

ipsAuthInstance OBJECT IDENTIFIER ::= { ipsAuthObjects 2 }
```

```
-- Instance Attributes Table

ipsAuthInstanceAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthInstanceAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of iSCSI instances present on the system."
::= { ipsAuthInstance 2 }

ipsAuthInstanceAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthInstanceAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing managment information applicable
        to a particular iSCSI instance."
    INDEX { ipsAuthInstIndex }
::= { ipsAuthInstanceAttributesTable 1 }

IpsAuthInstanceAttributesEntry ::= SEQUENCE {
    ipsAuthInstIndex                Unsigned32,
```



```
    ipsAuthInstDescr                SnmpAdminString
}

ipsAuthInstIndex OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a particular
        authentication instance."
::= { ipsAuthInstanceAttributesEntry 1 }

ipsAuthInstDescr OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "An octet string, determined by the implementation to describe
```

```
     the authentication instance.  When only a single instance is present,
     this object may be set to the zero-length string; with multiple
     authentication instances, it may be used in an implementation-dependent
     manner to describe the purpose of the respective instance."
::= { ipsAuthInstanceAttributesEntry 2 }


ipsAuthCertificate OBJECT IDENTIFIER ::= { ipsAuthObjects 3 }

-- Authorized Certificate Attributes Table

ipsAuthCertAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthCertAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of certificates that may be used to authenticate
     user identities."
::= { ipsAuthCertificate 1 }

ipsAuthCertAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthCertAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a certificate which may be used to authenticate
        a user identity within an authentication instance."
    INDEX { ipsAuthInstIndex, ipsAuthCertIndex }
::= { ipsAuthCertAttributesTable  1 }
```



```
IpsAuthCertAttributesEntry ::= SEQUENCE {
    ipsAuthCertIndex                  Unsigned32,
    ipsAuthCertDescription            SnmpAdminString,
    ipsAuthCertIdentity               OCTET STRING,
    ipsAuthCert                       OCTET STRING,
    ipsAuthCertRowStatus              RowStatus
}

ipsAuthCertIndex OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    not-accessible
```

```
    STATUS        current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a particular
        certificate instance within an authentication instance present
        on the node."
::= { ipsAuthCertAttributesEntry 1 }


ipsAuthCertDescription OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string describing this certificate."
::= { ipsAuthCertAttributesEntry 2 }

ipsAuthCertIdentity OBJECT-TYPE
    SYNTAX        OCTET STRING
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string, which is either a copy of the XXX attribute
     from the certificate, or an empty string.  If this attribute
     is not empty, it MUST match value of the XXX attribute from
     the certificate."
::= { ipsAuthCertAttributesEntry 3 }

ipsAuthCert OBJECT-TYPE
    SYNTAX        OCTET STRING
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The certificate, encoded in X.509 format."
::= { ipsAuthCertAttributesEntry 4 }

ipsAuthCertRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
```



```
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
```

```
     removed from this table via SNMP."
::= { ipsAuthCertAttributesEntry 5 }


ipsAuthIdentity OBJECT IDENTIFIER ::= { ipsAuthObjects 4 }

-- iSCSI User Identity Attributes Table

ipsAuthIdentAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthIdentAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
     "A list of user identities, each belonging to a particular
     ipsAuthInstance."
::= { ipsAuthIdentity 1 }

ipsAuthIdentAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthIdentAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        describing a user identity
        within an authentication instance on this node."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex }
::= { ipsAuthIdentAttributesTable  1 }

IpsAuthIdentAttributesEntry ::= SEQUENCE {
    ipsAuthIdentIndex                   Unsigned32,
    ipsAuthIdentDescription             SnmpAdminString,
    ipsAuthIdentRowStatus               RowStatus
}

ipsAuthIdentIndex OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a particular
        identity instance within an authentication instance present
     on the node."
::= { ipsAuthIdentAttributesEntry 1 }
```

```
ipsAuthIdentDescription OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string describing this particular identity."
::= { ipsAuthIdentAttributesEntry 2 }

ipsAuthIdentRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthIdentAttributesEntry 3 }


ipsAuthIdentityName OBJECT IDENTIFIER ::= { ipsAuthObjects 5 }

-- iSCSI User Initiator Name Attributes Table

ipsAuthIdentNameAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthIdentNameAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of unique names that can be used to positively
     identify a particular user identity."
::= { ipsAuthIdentityName 1 }

ipsAuthIdentNameAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthIdentNameAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a unique identity name which can be used
     to uniquely identify a user identity within a particular
     authentication instance."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthIdentNameIndex }
::= { ipsAuthIdentNameAttributesTable  1 }

IpsAuthIdentNameAttributesEntry ::= SEQUENCE {
    ipsAuthIdentNameIndex                   Unsigned32,
    ipsAuthIdentName                        SnmpAdminString,
    ipsAuthIdentNameRowStatus               RowStatus
```

```
}
```



```
ipsAuthIdentNameIndex OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a particular
        identity name instance within an ipsAuthIdentity within an
     authentication instance."
::= { ipsAuthIdentNameAttributesEntry 1 }

ipsAuthIdentName OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "A character string which is the unique name of an
     identity that may be used to identify this
     ipsAuthIdent entry."
::= { ipsAuthIdentNameAttributesEntry 2 }

ipsAuthIdentNameRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthIdentNameAttributesEntry 3 }


ipsAuthIdentityAddress OBJECT IDENTIFIER ::= { ipsAuthObjects 6 }

-- iSCSI User Initiator Address Attributes Table

-- Work: Add the FC stuff here and IANA Address family
ipsAuthIdentAddrAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthIdentAddrAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
```

```
        "A list of address ranges that are allowed to serve
     as the endpoint addresses of a particular identity.
     An address range includes a starting and ending address
     and an optional netmask, and an address type indicator,
     which can specify whether the address is IPv4, IPv6,
     FC-WWPN, or FC-WWNN."
::= { ipsAuthIdentityAddress 1 }
```



```
ipsAuthIdentAddrAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthIdentAddrAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to an address range which is used as part
     of the authentication of an identity
        within an authentication instance on this node."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthIdentAddrIndex }
::= { ipsAuthIdentAddrAttributesTable  1 }

IpsAuthIdentAddrAttributesEntry ::= SEQUENCE {
    ipsAuthIdentAddrIndex                   Unsigned32,
    ipsAuthIdentAddrType                    InetAddressType,
    ipsAuthIdentAddrStart                   InetAddress,
    ipsAuthIdentAddrEnd                     InetAddress,
    ipsAuthIdentAddrMask                    InetAddress,
    ipsAuthIdentAddrRowStatus               RowStatus
}

ipsAuthIdentAddrIndex OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a particular
        ipsAuthIdentAddress instance within an ipsAuthIdentity within an
     authentication instance present on the node."
::= { ipsAuthIdentAddrAttributesEntry 1 }

ipsAuthIdentAddrType OBJECT-TYPE
    SYNTAX        InetAddressType
```

```
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The type of Address in the ipsAuthIdentAddress start, end,
     and mask fields.  This type is taken from the IANA address
     family types; more types may be registered independently
     of this MIB."
::= { ipsAuthIdentAddrAttributesEntry 2 }

ipsAuthIdentAddrStart OBJECT-TYPE
    SYNTAX        InetAddress
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The starting address of the allowed address range."
```



```
::= { ipsAuthIdentAddrAttributesEntry 3 }

ipsAuthIdentAddrEnd OBJECT-TYPE
    SYNTAX        InetAddress
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The ending address of the allowed address range.  If the
         ipsAuthIdentAddrEntry specifies a single address, this shall
         match the ipsAuthIdentAddrStart."
::= { ipsAuthIdentAddrAttributesEntry 4 }


-- Work: Need to think through whether we need a mask.

ipsAuthIdentAddrMask OBJECT-TYPE
    SYNTAX        InetAddress
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The Address mask. -- NEED TO SPECIFY EXACTLY HOW USED W/RANGE"
::= { ipsAuthIdentAddrAttributesEntry 5 }

ipsAuthIdentAddrRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
```

```
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthIdentAddrAttributesEntry 6 }


ipsAuthCredential OBJECT IDENTIFIER ::= { ipsAuthObjects 7 }

-- Identity Credential Attributes Table

ipsAuthCredentialAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthCredentialAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of credentials related to user identities
     that are allowed as valid authenticators of the
     particular identity."
::= { ipsAuthCredential 1 }

ipsAuthCredentialAttributesEntry OBJECT-TYPE
```



```
    SYNTAX        IpsAuthCredentialAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a credential which authenticates a user
        identity within an authentication instance."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredentialAttributesTable  1 }

IpsAuthCredentialAttributesEntry ::= SEQUENCE {
    ipsAuthCredIndex                    Unsigned32,
    ipsAuthCredAuthMethod               AutonomousType,
    ipsAuthCredUserName                 SnmpAdminString,
    ipsAuthCredRowStatus                RowStatus
}

ipsAuthCredIndex OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
```

```
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a particular
        iSCSI Credential instance within an iSCSI instance present on the
        node."
::= { ipsAuthCredentialAttributesEntry 1 }

ipsAuthCredAuthMethod OBJECT-TYPE
    SYNTAX        AutonomousType
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
     "This object contains an OBJECT IDENTIFIER
     which identifies the authentication method
     used with this credential.

     Some standardized values for this object are defined
     within the ipsAuthMethods subtree."
::= { ipsAuthCredentialAttributesEntry 2 }

ipsAuthCredUserName OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string containing the user name for this credential,
     if it is applicable to the ipsAuthCredAuthMethod."
::= { ipsAuthCredentialAttributesEntry 3 }
```



```
ipsAuthCredRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthCredentialAttributesEntry 4 }


ipsAuthCredChap OBJECT IDENTIFIER ::= { ipsAuthObjects 8 }
```

```
-- Credential Chap-Specific Attributes Table

ipsAuthCredChapAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthCredChapAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of CHAP attributes for credentials that
     have their ipsAuthCredAuthMethod == ipsAuthMethodChap."
::= { ipsAuthCredChap 1 }

ipsAuthCredChapAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthCredChapAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a credential which has the ipsAuthCredAuthMethod
     set to the OID of ipsAuthMethodChap."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredChapAttributesTable  1 }

IpsAuthCredChapAttributesEntry ::= SEQUENCE {
    ipsAuthCredChapUserName            SnmpAdminString,
    ipsAuthCredChapPassword            SnmpAdminString,
    ipsAuthCredChapRowStatus           RowStatus
}

ipsAuthCredChapUserName OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string containing the CHAP user name for this
     credential."
```



```
::= { ipsAuthCredChapAttributesEntry 1 }

ipsAuthCredChapPassword OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
```

```
    STATUS        current
    DESCRIPTION
        "An octet string containing the password for this
     credential.  If written, it changes the password for
     the credential.  If read, it returns a zero-length
     string."
::= { ipsAuthCredChapAttributesEntry 2 }

ipsAuthCredChapRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthCredChapAttributesEntry 3 }


ipsAuthCredSrp OBJECT IDENTIFIER ::= { ipsAuthObjects 9 }

-- Credential Srp-Specific Attributes Table

ipsAuthCredSrpAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthCredSrpAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of SRP-specific attributes for credentials that
     have their ipsAuthCredAuthMethod == ipsAuthMethodSrp."
::= { ipsAuthCredSrp 1 }

ipsAuthCredSrpAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthCredSrpAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a credential which has the ipsAuthCredAuthMethod
     set to the OID of ipsAuthMethodSrp."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredSrpAttributesTable  1 }

IpsAuthCredSrpAttributesEntry ::= SEQUENCE {
```

```
    ipsAuthCredSrpUserName             SnmpAdminString,
    ipsAuthCredSrpPasswordVerifier     SnmpAdminString,
    ipsAuthCredSrpSalt                 SnmpAdminString,
    ipsAuthCredSrpRowStatus            RowStatus
}

ipsAuthCredSrpUserName OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string containing the CHAP user name for this
     credential."
::= { ipsAuthCredSrpAttributesEntry 1 }

ipsAuthCredSrpPasswordVerifier OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string containing the SRP password verifier
     for this credential."
::= { ipsAuthCredSrpAttributesEntry 2 }


-- Work: what is the size of Salt?  Should it be an integer?

ipsAuthCredSrpSalt OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string containing the salt value related to
     this credential."
::= { ipsAuthCredSrpAttributesEntry 3 }

ipsAuthCredSrpRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthCredSrpAttributesEntry 4 }


ipsAuthCredSpkm OBJECT IDENTIFIER ::= { ipsAuthObjects 10 }
```

```
-- Credential Spkm-Specific Attributes Table

ipsAuthCredSpkmAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthCredSpkmAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of SPKM-specific attributes for credentials that
     have their ipsAuthCredAuthMethod == ipsAuthMethodSpkm."
::= { ipsAuthCredSpkm 1 }

ipsAuthCredSpkmAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthCredSpkmAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a credential which has the ipsAuthCredAuthMethod
     set to the OID of ipsAuthMethodSpkm."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredSpkmAttributesTable  1 }

-- Work: Do we need to split out the cert identity here, or in
-- the certificate object?

IpsAuthCredSpkmAttributesEntry ::= SEQUENCE {
    ipsAuthCredSpkmPeerIdentity        OCTET STRING,
    ipsAuthCredSpkmPeerCert            Unsigned32,
    ipsAuthCredSpkmMyCert              Unsigned32,
    ipsAuthCredSpkmRowStatus           RowStatus
}

-- Work: Should this go here, or with the cert, or both?

ipsAuthCredSpkmPeerIdentity OBJECT-TYPE
    SYNTAX        OCTET STRING
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The identity to be authenticated by the public
     key certificate.  If ipsAuthCredSpkmPeerCert is not
     zero, this identity much match the XXXXXXX attribute
```

```
     within the certificate referenced by PeerCert."
::= { ipsAuthCredSpkmAttributesEntry 1 }

ipsAuthCredSpkmPeerCert OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    read-create
```



```
    STATUS        current
    DESCRIPTION
        "The index of the ipsAuthCertificateEntry that contains
     the certificate for the peer that is expected for
     this credential to be authenticated, or zero if this
     attribute is not used."
::= { ipsAuthCredSpkmAttributesEntry 2 }

-- Work: I'm not sure that the following belongs here, yet.

ipsAuthCredSpkmMyCert OBJECT-TYPE
    SYNTAX        Unsigned32 (1..4294967295)
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The index of the ipsAuthCertificateEntry that contains
     the certificate that will be provided to the other
     system when this this credential to be authenticated,
     or zero if this attribute is not used."
::= { ipsAuthCredSpkmAttributesEntry 3 }

ipsAuthCredSpkmRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthCredSpkmAttributesEntry 4 }


ipsAuthCredKerberos OBJECT IDENTIFIER ::= { ipsAuthObjects 11 }

-- Credential Kerberos-Specific Attributes Table
```

```
ipsAuthCredKerbAttributesTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF IpsAuthCredKerbAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A list of SRP-specific attributes for credentials that
     have their ipsAuthCredAuthMethod == ipsAuthMethodKerberos."
::= { ipsAuthCredKerberos 1 }

ipsAuthCredKerbAttributesEntry OBJECT-TYPE
    SYNTAX        IpsAuthCredKerbAttributesEntry
    MAX-ACCESS    not-accessible
    STATUS        current
```



```
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a credential which has the ipsAuthCredAuthMethod
     set to the OID of ipsAuthMethodKerberos."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredKerbAttributesTable  1 }

IpsAuthCredKerbAttributesEntry ::= SEQUENCE {
    ipsAuthCredKerbAttribute           SnmpAdminString,
    ipsAuthCredKerbRowStatus           RowStatus
}

-- Work:  The following is a placeholder attribute, since I
-- haven't figured out what to configure for Kerberos.

ipsAuthCredKerbAttribute OBJECT-TYPE
    SYNTAX        SnmpAdminString
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "An octet string containing a Kerberos attribute
     for this credential."
::= { ipsAuthCredKerbAttributesEntry 1 }

ipsAuthCredKerbRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
```

```
    DESCRIPTION
        "This field allows entries to be dynamically added and
     removed from this table via SNMP."
::= { ipsAuthCredKerbAttributesEntry 2 }




-----------------------------------------------------------------------
-- Notifications

-- There are no notifications necessary in this MIB.

-----------------------------------------------------------------------

-- Conformance Statements

ipsAuthGroups OBJECT IDENTIFIER ::= { ipsAuthConformance 1 }

ipsAuthInstanceAttributesGroup OBJECT-GROUP
```



```
    OBJECTS {
        ipsAuthInstDescr
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
     authentication instances."
::= { ipsAuthGroups 1 }

ipsAuthIdentCertAttributesGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthCertDescription,
    ipsAuthCert,
    ipsAuthCertIdentity,
    ipsAuthCertRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
     certicates within an authentication instance."
::= { ipsAuthGroups 2 }
```

```
ipsAuthIdentAttributesGroup OBJECT-GROUP
    OBJECTS {
        ipsAuthIdentDescription,
    ipsAuthIdentRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
     user identities within an authentication instance."
::= { ipsAuthGroups 3 }

ipsAuthIdentNameAttributesGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthIdentName,
    ipsAuthIdentNameRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
     user names within user identities within an authentication
     instance."
::= { ipsAuthGroups 4 }

ipsAuthIdentAddrAttributesGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthIdentAddrType,
```



```
    ipsAuthIdentAddrStart,
    ipsAuthIdentAddrEnd,
    ipsAuthIdentAddrMask,
    ipsAuthIdentAddrRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
     address ranges within user identities within an authentication
     instance."
::= { ipsAuthGroups 5 }

ipsAuthIdentCredAttributesGroup OBJECT-GROUP
    OBJECTS {
```

```
    ipsAuthCredAuthMethod,
    ipsAuthCredUserName,
    ipsAuthCredRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
     credentials within user identities within an authentication
     instance."
::= { ipsAuthGroups 6 }

ipsAuthIdentChapAttrGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthCredChapUserName,
    ipsAuthCredChapPassword,
    ipsAuthCredChapRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about CHAP
     credentials within user identities within an authentication
     instance."
::= { ipsAuthGroups 7 }

ipsAuthIdentSrpAttrGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthCredSrpUserName,
    ipsAuthCredSrpPasswordVerifier,
    ipsAuthCredSrpSalt,
    ipsAuthCredSrpRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about SRP
```



```
     credentials within user identities within an authentication
     instance."
::= { ipsAuthGroups 8 }

ipsAuthIdentSpkmAttrGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthCredSpkmPeerIdentity,
```

```
    ipsAuthCredSpkmPeerCert,
    ipsAuthCredSpkmMyCert,
    ipsAuthCredSpkmRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about SPKM
     credentials within user identities within an authentication
     instance."
::= { ipsAuthGroups 9 }

ipsAuthIdentKerberosAttrGroup OBJECT-GROUP
    OBJECTS {
    ipsAuthCredKerbAttribute,
    ipsAuthCredKerbRowStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about Kerberos
     credentials within user identities within an authentication
     instance."
::= { ipsAuthGroups 10 }

-- Work need to add the rest of the groups

--------------------------------------------------------------------------

ipsAuthCompliances OBJECT IDENTIFIER ::= { ipsAuthConformance 2 }

ipsAuthComplianceV1 MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "Initial version of compliance statement based on
        initial version of MIB.

     The Instance and Identity groups are mandatory;
     at least one of the other groups (Name, Address,
     Credential, Certificate) is also mandatory for
     any given implementation."
    MODULE       -- this module
    MANDATORY-GROUPS {
```

```
        ipsAuthInstanceAttributesGroup,
        ipsAuthIdentAttributesGroup
    }

    -- Conditionally mandatory groups to be included with
    -- the mandatory groups when necessary.

    GROUP ipsAuthIdentNameAttributesGroup
    DESCRIPTION
        "This group is mandatory for all implementations
     that make use of unique identity names."

    GROUP ipsAuthIdentAddrAttributesGroup
    DESCRIPTION
        "This group is mandatory for all implementations
     that use addresses to help authenticate identities."

    GROUP ipsAuthIdentCredAttributesGroup
    DESCRIPTION
        "This group is mandatory for all implementations
     that use credentials to help authenticate identities."

    GROUP ipsAuthIdentCertAttributesGroup
    DESCRIPTION
        "This group is mandatory for all implementations
     that make use of public key certificates."

::= { ipsAuthCompliances 1 }

END
```

8. Security Considerations

WORK: Need some text about all the bad things that can happen when someone gains write access to this MIB.

WORK: Considerations for read only.

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

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model RFC 2574 [RFC2574] and the View-

based Access Control Model RFC 2575 [RFC2575] is recommended.

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

9. References

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

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

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

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

[RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

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

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

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

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

[RFC1906] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.

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

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

[RFC1905] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.

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

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

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

[RFC2012] McCloghrie, K., "SNMPv2 Management Information Base for the Transmission Control Protocol using SMIv2", RFC 2012, November 1996.

[RFC2851] Daniele, M., et. al., "Textual Conventions for Internet Network Addresses", RFC 2851, June 2000.

[IPV6MIB] Daniele, M., et. al., "Textual Conventions for Internet Network Addresses", draft-ietf-ops-rfc2851-update-06.txt, February 2001

[IANA-AF] IANA, "WORK: something about assigned enum types for address families", http://www.iana.org/something.

[RFC1213] K. McCloghrie, M.T. Rose, "Management Information Base for Network Management of TCP/IP-based internets:MIB-II", March 1991.

[RFC2011] K. McCloghrie, "SNMPv2 Management Information Base for the Internet Protocol using SMIv2", November 1996.

[RFC1994] W. Simpson, "PPP Challenge Handshake Authentication Protocol (CHAP)", August 1996.

[RFC1510] J. Kohl, C. Neuman, "The Kerberos Network Authentication Service (V5)", September 1993.

[RFC2025] C. Adams, "The Simple Public-Key GSS-API Mechanism (SPKM)", October 1996.

[RFC2945] T. Wu, "The SRP Authentication and Key Exchange System", September 2000.

[RFC2465] D. Haskin, S. Onishi, "Management Information Base for IP Version 6: Textual Conventions and General Group", December 1998.

[ISCSI] Satran, J., et. al., "iSCSI", draft-ietf-ips-iSCSI-10, Febrary 2002.

[RFC1737] K. Sollins, L. Masinter, "Functional Requirements for Uniform Resource Names", December 1994.

[X.509] ITU-T Recommendation X.509 (1997 E), "Information Technology - Open Systems Interconnection - The Directory: Authentication Framework", June 1997.

10. Authors' Addresses

Mark Bakke
Postal: Cisco Systems, Inc
6450 Wedgwood Road, Suite 130
Maple Grove, MN

USA 55311

Tel: +1 763-398-1000
Fax: +1 763-398-1001

E-mail: mbakke@cisco.com

Jim Muchow
Postal: Cisco Systems, Inc
6450 Wedgwood Road, Suite 130
Maple Grove, MN
USA 55311

Tel: +1 763-398-1000
Fax: +1 763-398-1001

E-mail: jmuchow@cisco.com"