

Network Working Group  
Request for Comments: 4545  
Category: Standards Track

M. Bakke  
Cisco Systems  
J. Muchow  
Qlogic Corp.  
May 2006

## Definitions of Managed Objects for IP Storage User Identity Authorization

### Status of This Memo

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

### Copyright Notice

Copyright (C) The Internet Society (2006).

### 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 manage access control, for use with various protocols. This document was motivated by the need for the configuration of authorized user identities for the iSCSI protocol, but has been extended to be useful for other protocols that have similar requirements. It is important to note that this MIB module provides only the set of identities to be used within access lists; it is the responsibility of other MIB modules making use of this one to tie them to their own access lists or other authorization control methods.

## Table of Contents

1. Introduction .....	3
2. Specification of Requirements .....	3
3. The Internet-Standard Management Framework .....	3
4. Relationship to Other MIB Modules .....	3
5. Relationship to the USM MIB Module .....	4
6. Relationship to SNMP Contexts .....	5
7. Discussion .....	5
7.1. Authorization MIB Object Model .....	5
7.2. ipsAuthInstance .....	6
7.3. ipsAuthIdentity .....	7
7.4. ipsAuthIdentityName .....	7
7.5. ipsAuthIdentityAddress .....	8
7.6. ipsAuthCredential .....	8
7.7. IP, Fibre Channel, and Other Addresses .....	9
7.8. Descriptors: Using OIDs in Place of Enumerated Types .....	10
7.9. Notifications .....	10
8. MIB Definitions .....	11
9. Security Considerations .....	35
9.1. MIB Security Considerations .....	35
9.2. Other Security Considerations .....	38
10. IANA Considerations .....	40
11. Normative References .....	40
12. Informative References .....	41
13. Acknowledgements .....	41

## 1. Introduction

This MIB module will be used to configure and/or look at the configuration of user identities and their credential information. For the purposes of this MIB module, 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 authorized to access a resource.

Most objects in this MIB module have a MAX-ACCESS of read-create; this module 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.

## 2. Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

## 3. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

## 4. Relationship to Other MIB Modules

The IPS-AUTH-MIB module does not directly address objects within other modules. The identity address objects contain IPv4, IPv6, or other address types, and as such they may be indirectly related to objects within the IP [RFC4293] MIB module.

This MIB module does not provide actual authorization or access control lists; it provides a means to identify entities that can be included in other authorization lists. This should generally be done in MIB modules 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 are not generic enough to be included here.

The user identity objects within this module are typically referenced from other modules by a RowPointer within that module. A module 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 module. This is neither required nor restricted by this MIB module.

## 5. Relationship to the USM MIB Module

The User-based Security Model (USM) [RFC3414] also defines the concept of a user, defining authentication and privacy protocols and their credentials. The definition of USM includes the SNMP-USER-BASED-SM-MIB module allows configuration of SNMPv3 user credentials to protect SNMPv3 messages. Although USM's users are not related to the user identities managed by the IPS-AUTH-MIB module defined in this document, USM will often be implemented on the same system as the IPS-AUTH-MIB module, with the SNMP-USER-BASED-SM-MIB module used to manage the security protecting SNMPv3 messages, including those that access the IPS-AUTH-MIB module.

The term "user" in this document is distinct from an SNMPv3 user and is intended to include, but is not limited to, users of IP storage devices. A "user" in this document is a collection of user names (unique identifiers), user addresses, and credentials that can be used together to determine whether an entity should be allowed access to a resource. Each user can have multiple names, addresses, and credentials. As a result, this MIB module is particularly suited to managing users of storage resources, which are typically given access control lists consisting of potentially multiple identifiers, addresses, and credentials. This MIB module provides for authorization lists only and does not include setting of data privacy parameters.

In contrast, an SNMPv3 user as defined in [RFC3414] has exactly one user-name, one authentication protocol, and one privacy protocol, along with their associated information and SNMP-specific information, such as an engine ID. These objects are defined to support exactly the information needed for SNMPv3 security.

For the remainder of this document, the term "user" means an IPS-AUTH-MIB user identity.

## 6. Relationship to SNMP Contexts

Each non-scalar object in the IPS-AUTH-MIB module is indexed first by an instance. Each instance is a collection of identities that can be used to authorize access to a resource. The use of an instance works well with partitionable or hierarchical devices and fits in logically with other management schemes. Instances do not replace SNMP contexts; however, they do provide a very simple way to assign a collection of identities within a device to one or more SNMP contexts, without having to do so for each identity's row.

## 7. Discussion

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

The IPS-AUTH-MIB module is structured around two primary "objects", the authorization 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.

### 7.1. Authorization MIB Object Model

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

```
ipsAuthInstance
-- A distinct authorization entity within the managed system.
-- Most implementations will have just one of these.
ipsAuthIdentity
-- A user identity, consisting of a set of identity names,
-- addresses, and credentials reflected in the following
-- objects:
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, IPv6, or Fibre Channel address range, at which
-- the identity is allowed to reside.
ipsAuthCredential
-- A single credential, such as a CHAP username,
```

```
-- which can be used to verify the identity.
ipsAuthCredChap
    -- CHAP-specific attributes for an ipsAuthCredential
ipsAuthCredSrp
    -- SRP-specific attributes
ipsAuthCredKerberos
    -- Kerberos-specific attributes
```

Each identity contains the information necessary to identify a particular end-point that wishes to access a service, such as iSCSI.

An identity can contain multiple names, addresses, and credentials. Each of these names, addresses, and credentials exists in its own row. If multiple rows of one of these three types are present, they are treated in an "OR" fashion; an entity to be authorized need only match one of the rows. If rows of different types are present (e.g., a name and an address), these are treated in an "AND" fashion; an entity to be authorized must match at least one row from each category. If there are no rows present of a category, this category is ignored.

For example, if an ipsAuthIdentity contains two rows of ipsAuthIdentityAddress, one row of ipsAuthCredential, and no rows of ipsAuthIdentityName, an entity must match the Credential row and at least one of the two Address rows to match the identity.

Index values such as ipsAuthInstIndex and ipsAuthIdentIndex are referenced in multiple tables, and rows can be added and deleted. An implementation should therefore attempt to keep all index values persistent across reboots; index values for rows that have been deleted must not be reused before a reboot.

## 7.2. ipsAuthInstance

The ipsAuthInstanceAttributesTable is the primary table of the IPS-AUTH-MIB module. Every other table entry in this module includes the index of an ipsAuthInstanceAttributesEntry as its primary index. An authorization instance is basically a managed set of identities.

Many implementations will include just one authorization 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 authorization instance.

- A set of stackable systems, each with its own set of identities, may be represented by a common SNMP agent. Each individual system would have its own authorization instance.
- Multiple protocols, each with its own set of identities, may exist within a single system and be represented by a single SNMP agent. In this case, each protocol may have its own authorization instance.

An entry in this table is often referenced by its name (`ipsAuthInstDescr`), which should be displayed to the user by the management station. When an implementation supports only one entry in this table, the description may be returned as a zero-length string.

### 7.3. `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 each have multiple values.

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

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

### 7.4. `ipsAuthIdentityName`

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

Implementations making use of the IPS-AUTH-MIB module 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 module 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 [RFC3720]. Any other MIB module defining names to be used as `ipsAuthIdentityName` objects should specify how its names are unique, and the domain within which they are unique.

If this table contains no entries associated with a particular user identity, the implementation does not need to check any name parameters when verifying 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.

#### 7.5. `ipsAuthIdentityAddress`

The `ipsAuthIdentAddrAttributesTable` contains a list of addresses at which the identity may reside. 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 must 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 a valid match. If no entries are present in this list for a given identity, its address is automatically assumed to match the identity.

Netmasks are not supported, since an address range can express the same thing with more flexibility. An application specifying addresses using network masks may do so, and convert to and from address ranges when reading or writing this MIB module.

#### 7.6. `ipsAuthCredential`

The `ipsAuthCredentialAttributesTable` contains a list of credentials, each of which may be used to verify a particular identity.



Each credential contains an authentication method to be used, such as CHAP [RFC1994], SRP [RFC2945], or Kerberos [RFC4120]. This attribute contains an object identifier instead of an enumerated type, allowing other MIB modules to add their own authentication methods, without modifying this MIB module.

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

An additional credential type can be added to this MIB module by defining a new OID in the ipsAuthMethodTypes subtree, and defining a new table specific to that credential type.

#### 7.7. IP, Fibre Channel, and Other Addresses

The IP addresses in this MIB module are represented by two attributes, one of type AddressFamilyNumbers, and the other of type AuthAddress. Each address can take on any of the types within the list of address family numbers; the most likely being IPv4, IPv6, or one of the Fibre Channel address types.

The type AuthAddress is an octet string. If the address family is IPv4 or IPv6, the format is taken from the InetAddress specified in [RFC4001]. If the address family is one of the Fibre Channel types, the format is identical to the FcNameIdOrZero type defined in [RFC4044].

### 7.8. 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 module. To make this work, this module 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 that is an OID instead of an enumerated type. These OIDs can either indicate the object identities defined in this module, or object identities defined elsewhere, such as in an enterprise MIB module. 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 module, and return its OID whenever one of these attributes is using that method.

### 7.9. Notifications

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

## 8. MIB Definitions

```
IPS-AUTH-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
MODULE-IDENTITY, OBJECT-TYPE, OBJECT-IDENTITY, Unsigned32,  
mib-2
```

```
FROM SNMPv2-SMI
```

```
TEXTUAL-CONVENTION, RowStatus, AutonomousType, StorageType  
FROM SNMPv2-TC
```

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

```
SnmpAdminString
```

```
FROM SNMP-FRAMEWORK-MIB -- RFC 3411
```

```
AddressFamilyNumbers
```

```
FROM IANA-ADDRESS-FAMILY-NUMBERS-MIB
```

```
;
```

```
ipsAuthMibModule MODULE-IDENTITY
```

```
LAST-UPDATED "200605220000Z" -- May 22, 2006
```

```
ORGANIZATION "IETF IPS Working Group"
```

```
CONTACT-INFO
```

```
"
```

```
Mark Bakke
```

```
Postal: Cisco Systems, Inc
```

```
7900 International Drive, Suite 400
```

```
Bloomington, MN
```

```
USA 55425
```

```
E-mail: mbakke@cisco.com
```

```
James Muchow
```

```
Postal: Qlogic Corp.
```

```
6321 Bury Dr.
```

```
Eden Prairie, MN
```

```
USA 55346
```

```
E-Mail: james.muchow@qlogic.com"
```

```
DESCRIPTION
```

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

```
Copyright (C) The Internet Society (2006). This version of  
this MIB module is part of RFC 4545; see the RFC itself for  
full legal notices."
```

REVISION "200605220000Z" -- May 22, 2006

DESCRIPTION

"Initial version of the IP Storage Authentication MIB module,  
published as RFC 4545"

::= { mib-2 141 }

ipsAuthNotifications OBJECT IDENTIFIER ::= { ipsAuthMibModule 0 }  
ipsAuthObjects OBJECT IDENTIFIER ::= { ipsAuthMibModule 1 }  
ipsAuthConformance OBJECT IDENTIFIER ::= { ipsAuthMibModule 2 }

-- Textual Conventions

IpsAuthAddress ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"IP Storage requires the use of address information that uses not only the InetAddress type defined in the INET-ADDRESS-MIB, but also Fibre Channel type defined in the Fibre Channel Management MIB. Although these address types are recognized in the IANA Address Family Numbers MIB, the addressing mechanisms have not been merged into a well-known, common type. This data type, the IpsAuthAddress, performs the merging for this MIB module.

The formats of objects of this type are determined by a corresponding object with syntax AddressFamilyNumbers, and thus every object defined using this TC must identify the object with syntax AddressFamilyNumbers that specifies its type.

The syntax and semantics of this object depend on the identified AddressFamilyNumbers object as follows:

AddressFamilyNumbers	this object
=====	=====
ipV4(1)	restricted to the same syntax and semantics as the InetAddressIPv4 TC.
ipV6(2)	restricted to the same syntax and semantics as the InetAddressIPv6 TC.
fibresChannelWWPN (22)	
& fibresChannelWWNN(23)	restricted to the same syntax and semantics as the FcNameIdOrZero TC.

Types other than the above should not be used unless

the corresponding format of the IpsAuthAddress object is further specified (e.g., in a future revision of this TC)."

## REFERENCE

"IANA-ADDRESS-FAMILY-NUMBERS-MIB;  
INET-ADDRESS-MIB (RFC 4001);  
FC-MGMT-MIB (RFC 4044)."

SYNTAX OCTET STRING (SIZE(0..255))

--\*\*\*\*\*

ipsAuthDescriptors OBJECT IDENTIFIER ::= { ipsAuthObjects 1 }

ipsAuthMethodTypes OBJECT-IDENTITY

STATUS current

## DESCRIPTION

"Registration point for Authentication Method Types."

REFERENCE "RFC 3720, iSCSI Protocol Specification."

::= { ipsAuthDescriptors 1 }

ipsAuthMethodNone OBJECT-IDENTITY

STATUS current

## DESCRIPTION

"The authoritative identifier when no authentication method is used."

REFERENCE "RFC 3720, iSCSI Protocol Specification."

::= { ipsAuthMethodTypes 1 }

ipsAuthMethodSrp OBJECT-IDENTITY

STATUS current

## DESCRIPTION

"The authoritative identifier when the authentication method is SRP."

REFERENCE "RFC 3720, iSCSI Protocol Specification."

::= { ipsAuthMethodTypes 2 }

ipsAuthMethodChap OBJECT-IDENTITY

STATUS current

## DESCRIPTION

"The authoritative identifier when the authentication method is CHAP."

REFERENCE "RFC 3720, iSCSI Protocol Specification."

::= { ipsAuthMethodTypes 3 }

ipsAuthMethodKerberos OBJECT-IDENTITY

STATUS current

## DESCRIPTION

"The authoritative identifier when the authentication method is Kerberos."

```

REFERENCE "RFC 3720, iSCSI Protocol Specification."
::= { ipsAuthMethodTypes 4 }

--*****

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 Authorization instances present on the system."
    ::= { ipsAuthInstance 2 }

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

IpsAuthInstanceAttributesEntry ::= SEQUENCE {
    ipsAuthInstIndex      Unsigned32,
    ipsAuthInstDescr      SnmpAdminString,
    ipsAuthInstStorageType StorageType
}

ipsAuthInstIndex OBJECT-TYPE
    SYNTAX          Unsigned32 (1..4294967295)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An arbitrary integer used to uniquely identify a
         particular authorization instance. This index value
         must not be modified or reused by an agent unless
         a reboot has occurred. An agent should attempt to
         keep this value persistent across reboots."
    ::= { ipsAuthInstanceAttributesEntry 1 }

ipsAuthInstDescr OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-write

```

STATUS current

DESCRIPTION

"A character string, determined by the implementation to describe the authorization instance. When only a single instance is present, this object may be set to the zero-length string; with multiple authorization instances, it must be set to a unique value in an implementation-dependent manner to describe the purpose of the respective instance. If this is deployed in a master agent with more than one subagent implementing this MIB module, the master agent is responsible for ensuring that this object is unique across all subagents."

::= { ipsAuthInstanceAttributesEntry 2 }

ipsAuthInstStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The storage type for all read-write objects within this row. Rows in this table are always created via an external process, and may have a storage type of readOnly or permanent. Conceptual rows having the value 'permanent' need not allow write access to any columnar objects in the row.

If this object has the value 'volatile', modifications to read-write objects in this row are not persistent across reboots. If this object has the value 'nonVolatile', modifications to objects in this row are persistent.

An implementation may choose to allow this object to be set to either 'nonVolatile' or 'volatile', allowing the management application to choose this behavior."

DEFVAL { volatile }

::= { ipsAuthInstanceAttributesEntry 3 }

ipsAuthIdentity OBJECT IDENTIFIER ::= { ipsAuthObjects 3 }

-- 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 authorization
        instance on this node."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex }
 ::= { ipsAuthIdentAttributesTable 1 }

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

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 authorization
        instance present on the node. This index value
        must not be modified or reused by an agent unless
        a reboot has occurred. An agent should attempt to
        keep this value persistent across reboots."
 ::= { ipsAuthIdentAttributesEntry 1 }

ipsAuthIdentDescription OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "A character 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. When adding a row to this table, all non-Index/RowStatus objects must be set. Rows may be discarded using RowStatus. The value of ipsAuthIdentityDescription may be set while ipsAuthIdentityRowStatus is 'active'."

```
::= { ipsAuthIdentityAttributesEntry 3 }
```

## ipsAuthIdentityStorageType OBJECT-TYPE

SYNTAX           StorageType  
MAX-ACCESS       read-create  
STATUS           current

## DESCRIPTION

"The storage type for all read-create objects in this row. Rows in this table that were created through an external process may have a storage type of readOnly or permanent. Conceptual rows having the value 'permanent' need not allow write access to any columnar objects in the row."

DEFVAL           { nonVolatile }

```
::= { ipsAuthIdentityAttributesEntry 4 }
```

```
ipsAuthIdentityName OBJECT IDENTIFIER ::= { ipsAuthObjects 4 }
```

## -- User Initiator Name Attributes Table

## ipsAuthIdentityNameAttributesTable OBJECT-TYPE

SYNTAX           SEQUENCE OF IpsAuthIdentityNameAttributesEntry  
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 }
```

## ipsAuthIdentityNameAttributesEntry OBJECT-TYPE

SYNTAX           IpsAuthIdentityNameAttributesEntry  
MAX-ACCESS       not-accessible  
STATUS           current

## DESCRIPTION

"An entry (row) containing management information applicable to a unique identity name, which can be used to identify a user identity within a particular authorization instance."

INDEX { ipsAuthInstIndex, ipsAuthIdentityIndex,  
          ipsAuthIdentityNameIndex }

```
::= { ipsAuthIdentityNameAttributesTable 1 }
```

```

IpsAuthIdentNameAttributesEntry ::= SEQUENCE {
    ipsAuthIdentNameIndex      Unsigned32,
    ipsAuthIdentName          SnmpAdminString,
    ipsAuthIdentNameRowStatus  RowStatus,
    ipsAuthIdentNameStorageType StorageType
}

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 authorization instance.
        This index value must not be modified or reused by
        an agent unless a reboot has occurred. An agent
        should attempt to keep this value persistent across
        reboots."
    ::= { ipsAuthIdentNameAttributesEntry 1 }

ipsAuthIdentName OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "A character string that 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. When adding a row to
        this table, all non-Index/RowStatus objects must be set.
        Rows may be discarded using RowStatus. The value of
        ipsAuthIdentName may be set when this value is 'active'."
    ::= { ipsAuthIdentNameAttributesEntry 3 }

ipsAuthIdentNameStorageType OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION

```

"The storage type for all read-create objects in this row. Rows in this table that were created through an external process may have a storage type of readOnly or permanent. Conceptual rows having the value 'permanent' need not allow write access to any columnar objects in the row."

```

DEFVAL      { nonVolatile }
::= { ipsAuthIdentNameAttributesEntry 4 }

ipsAuthIdentityAddress OBJECT IDENTIFIER ::= { ipsAuthObjects 5 }

-- User Initiator Address Attributes Table

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 that is used as part
         of the authorization of an identity
         within an authorization instance on this node."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex,
            ipsAuthIdentAddrIndex }
    ::= { ipsAuthIdentAddrAttributesTable 1 }

IpsAuthIdentAddrAttributesEntry ::= SEQUENCE {
    ipsAuthIdentAddrIndex      Unsigned32,
    ipsAuthIdentAddrType       AddressFamilyNumbers,
    ipsAuthIdentAddrStart      IpsAuthAddress,
    ipsAuthIdentAddrEnd        IpsAuthAddress,
    ipsAuthIdentAddrRowStatus   RowStatus,
    ipsAuthIdentAddrStorageType StorageType
}

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 authorization instance present on the node.

This index value must not be modified or reused by an agent unless a reboot has occurred. An agent should attempt to keep this value persistent across reboots."

::= { ipsAuthIdentAddrAttributesEntry 1 }

## ipsAuthIdentAddrType OBJECT-TYPE

SYNTAX            AddressFamilyNumbers  
 MAX-ACCESS       read-create  
 STATUS           current

## DESCRIPTION

"The address types used in the ipsAuthIdentAddrStart and ipsAuthAddrEnd objects. This type is taken from the IANA address family types."

::= { ipsAuthIdentAddrAttributesEntry 2 }

## ipsAuthIdentAddrStart OBJECT-TYPE

SYNTAX            IpsAuthAddress  
 MAX-ACCESS       read-create  
 STATUS           current

## DESCRIPTION

"The starting address of the allowed address range. The format of this object is determined by ipsAuthIdentAddrType."

::= { ipsAuthIdentAddrAttributesEntry 3 }

## ipsAuthIdentAddrEnd OBJECT-TYPE

SYNTAX            IpsAuthAddress  
 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. The format of this object is determined by ipsAuthIdentAddrType."

::= { ipsAuthIdentAddrAttributesEntry 4 }

## 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.  When adding a row to
    this table, all non-Index/RowStatus objects must be set.
    Rows may be discarded using RowStatus.  The values of
    ipsAuthIdentAddrStart and ipsAuthIdentAddrEnd may be set
    when this value is 'active'.  The value of
    ipsAuthIdentAddrType may not be set when this value is
    'active'."
::= { ipsAuthIdentAddrAttributesEntry 5 }

ipsAuthIdentAddrStorageType OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The storage type for all read-create objects in this row.
        Rows in this table that were created through an external
        process may have a storage type of readOnly or permanent.
        Conceptual rows having the value 'permanent' need not
        allow write access to any columnar objects in the row."
    DEFVAL      { nonVolatile }
::= { ipsAuthIdentAddrAttributesEntry 6 }

ipsAuthCredential OBJECT IDENTIFIER ::= { ipsAuthObjects 6 }

-- 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 that verifies a user
        identity within an authorization instance."

```

To provide complete information in this MIB for a credential, the management station must not only create the row in this table but must also create a row in another table, where the other table is determined by the value of ipsAuthCredAuthMethod, e.g., if ipsAuthCredAuthMethod has the value ipsAuthMethodChap, a row must be created in the ipsAuthCredChapAttributesTable."

```
INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredentialAttributesTable 1 }
```

```
IpsAuthCredentialAttributesEntry ::= SEQUENCE {
    ipsAuthCredIndex          Unsigned32,
    ipsAuthCredAuthMethod     AutonomousType,
    ipsAuthCredRowStatus      RowStatus,
    ipsAuthCredStorageType    StorageType
}
```

ipsAuthCredIndex OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An arbitrary integer used to uniquely identify a particular Credential instance within an instance present on the node.

This index value must not be modified or reused by an agent unless a reboot has occurred. An agent should attempt to keep this value persistent across reboots."

```
::= { ipsAuthCredentialAttributesEntry 1 }
```

ipsAuthCredAuthMethod OBJECT-TYPE

SYNTAX AutonomousType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object contains an OBJECT IDENTIFIER that identifies the authentication method used with this credential.

When a row is created in this table, a corresponding row must be created by the management station in a corresponding table specified by this value.

When a row is deleted from this table, the corresponding row must be automatically deleted by the agent in the corresponding table specified by this value.

If the value of this object is `ipsAuthMethodNone`, no corresponding rows are created or deleted from other tables.

Some standardized values for this object are defined within the `ipsAuthMethodTypes` subtree."

```
::= { ipsAuthCredentialAttributesEntry 2 }
```

```
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. When adding a row to this table, all non-Index/RowStatus objects must be set. Rows may be discarded using RowStatus. The value of `ipsAuthCredAuthMethod` must not be changed while this row is 'active'."

```
::= { ipsAuthCredentialAttributesEntry 3 }
```

```
ipsAuthCredStorageType OBJECT-TYPE
```

```
SYNTAX          StorageType
```

```
MAX-ACCESS      read-create
```

```
STATUS          current
```

```
DESCRIPTION
```

"The storage type for all read-create objects in this row. Rows in this table that were created through an external process may have a storage type of `readOnly` or `permanent`. Conceptual rows having the value 'permanent' need not allow write access to any columnar objects in the row."

```
DEFVAL          { nonVolatile }
```

```
::= { ipsAuthCredentialAttributesEntry 4 }
```

```
ipsAuthCredChap OBJECT IDENTIFIER ::= { ipsAuthObjects 7 }
```

```
-- 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 use `ipsAuthMethodChap` as their `ipsAuthCredAuthMethod`.

A row in this table can only exist when an instance of the `ipsAuthCredAuthMethod` object exists (or is created

simultaneously) having the same instance identifiers  
and a value of '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 that uses  
ipsAuthMethodChap as their ipsAuthCredAuthMethod.

When a row is created in ipsAuthCredentialAttributesTable  
with ipsAuthCredAuthMethod = ipsAuthCredChap, the  
management station must create a corresponding row  
in this table.

When a row is deleted from ipsAuthCredentialAttributesTable  
with ipsAuthCredAuthMethod = ipsAuthCredChap, the  
agent must delete the corresponding row (if any) in  
this table."

INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }  
 ::= { ipsAuthCredChapAttributesTable 1 }

IpsAuthCredChapAttributesEntry ::= SEQUENCE {  
ipsAuthCredChapUserName SnmpAdminString,  
ipsAuthCredChapRowStatus RowStatus,  
ipsAuthCredChapStorageType StorageType  
}

ipsAuthCredChapUserName OBJECT-TYPE

SYNTAX SnmpAdminString  
MAX-ACCESS read-create  
STATUS current  
DESCRIPTION

"A character string containing the CHAP user name for this  
credential."

REFERENCE

"W. Simpson, RFC 1994: PPP Challenge Handshake  
Authentication Protocol (CHAP), August 1996"

::= { ipsAuthCredChapAttributesEntry 1 }

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. When adding a row to this table, all non-Index/RowStatus objects must be set. Rows may be discarded using RowStatus. The value of ipsAuthCredChapUserName may be changed while this row is 'active'."

```
 ::= { ipsAuthCredChapAttributesEntry 2 }
```

ipsAuthCredChapStorageType OBJECT-TYPE

SYNTAX	StorageType
MAX-ACCESS	read-create
STATUS	current

DESCRIPTION

"The storage type for all read-create objects in this row. Rows in this table that were created through an external process may have a storage type of readOnly or permanent. Conceptual rows having the value 'permanent' need not allow write access to any columnar objects in the row."

DEFVAL	{ nonVolatile }
--------	-----------------

```
 ::= { ipsAuthCredChapAttributesEntry 3 }
```

ipsAuthCredSrp OBJECT IDENTIFIER ::= { ipsAuthObjects 8 }

-- Credential Srp-Specific Attributes Table

ipsAuthCredSrpAttributesTable OBJECT-TYPE

SYNTAX	SEQUENCE OF IpsAuthCredSrpAttributesEntry
MAX-ACCESS	not-accessible
STATUS	current

DESCRIPTION

"A list of SRP attributes for credentials that use ipsAuthMethodSrp as its ipsAuthCredAuthMethod.

A row in this table can only exist when an instance of the ipsAuthCredAuthMethod object exists (or is created simultaneously) having the same instance identifiers and a value of '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 that uses ipsAuthMethodSrp as their ipsAuthCredAuthMethod."

When a row is created in ipsAuthCredentialAttributesTable with ipsAuthCredAuthMethod = ipsAuthCredSrp, the management station must create a corresponding row in this table.

When a row is deleted from ipsAuthCredentialAttributesTable with ipsAuthCredAuthMethod = ipsAuthCredSrp, the agent must delete the corresponding row (if any) in this table."

```
INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
::= { ipsAuthCredSrpAttributesTable 1 }
```

```
IpsAuthCredSrpAttributesEntry ::= SEQUENCE {
    ipsAuthCredSrpUserName      SnmpAdminString,
    ipsAuthCredSrpRowStatus     RowStatus,
    ipsAuthCredSrpStorageType   StorageType
}
```

ipsAuthCredSrpUserName OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"A character string containing the SRP user name for this credential."

REFERENCE

"T. Wu, RFC 2945: The SRP Authentication and Key Exchange System, September 2000"

```
::= { ipsAuthCredSrpAttributesEntry 1 }
```

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. When adding a row to this table, all non-Index/RowStatus objects must be set. Rows may be discarded using RowStatus. The value of ipsAuthCredSrpUserName may be changed while the status of this row is 'active'."

```
::= { ipsAuthCredSrpAttributesEntry 2 }
```

ipsAuthCredSrpStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The storage type for all read-create objects in this row. Rows in this table that were created through an external process may have a storage type of readOnly or permanent. Conceptual rows having the value 'permanent' need not allow write access to any columnar objects in the row."

```

DEFVAL          { nonVolatile }
::= { ipsAuthCredSrpAttributesEntry 3 }

ipsAuthCredKerberos OBJECT IDENTIFIER ::= { ipsAuthObjects 9 }

-- Credential Kerberos-Specific Attributes Table

ipsAuthCredKerbAttributesTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF IpsAuthCredKerbAttributesEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "A list of Kerberos attributes for credentials that
        use ipsAuthMethodKerberos as their ipsAuthCredAuthMethod.

        A row in this table can only exist when an instance of
        the ipsAuthCredAuthMethod object exists (or is created
        simultaneously) having the same instance identifiers
        and a value of 'ipsAuthMethodKerb'."
    ::= { ipsAuthCredKerberos 1 }

ipsAuthCredKerbAttributesEntry OBJECT-TYPE
    SYNTAX          IpsAuthCredKerbAttributesEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry (row) containing management information
        applicable to a credential that uses
        ipsAuthMethodKerberos as its ipsAuthCredAuthMethod.

        When a row is created in ipsAuthCredentialAttributesTable
        with ipsAuthCredAuthMethod = ipsAuthCredKerberos, the
        management station must create a corresponding row
        in this table.

        When a row is deleted from ipsAuthCredentialAttributesTable
        with ipsAuthCredAuthMethod = ipsAuthCredKerberos, the
        agent must delete the corresponding row (if any) in
        this table."
    INDEX { ipsAuthInstIndex, ipsAuthIdentIndex, ipsAuthCredIndex }
    ::= { ipsAuthCredKerbAttributesTable 1 }

IpsAuthCredKerbAttributesEntry ::= SEQUENCE {

```

```

    ipsAuthCredKerbPrincipal      SnmpAdminString,
    ipsAuthCredKerbRowStatus      RowStatus,
    ipsAuthCredKerbStorageType    StorageType
}

ipsAuthCredKerbPrincipal OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "A character string containing a Kerberos principal
        for this credential."
    REFERENCE
        "C. Neuman, S. Hartman, and K. Raeburn, RFC 4120:
        The Kerberos Network Authentication Service (V5),
        July 2005"
    ::= { 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. When adding a row to
        this table, all non-Index/RowStatus objects must be set.
        Rows may be discarded using RowStatus. The value of
        ipsAuthCredKerbPrincipal may be changed while this row
        is 'active'."
    ::= { ipsAuthCredKerbAttributesEntry 2 }

ipsAuthCredKerbStorageType OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "The storage type for all read-create objects in this row.
        Rows in this table that were created through an external
        process may have a storage type of readOnly or permanent.
        Conceptual rows having the value 'permanent' need not
        allow write access to any columnar objects in the row."
    DEFVAL      { nonVolatile }
    ::= { ipsAuthCredKerbAttributesEntry 3 }

--*****
-- Notifications

-- There are no notifications necessary in this MIB module.

```

--\*\*\*\*\*

-- Conformance Statements

ipsAuthCompliances OBJECT IDENTIFIER ::= { ipsAuthConformance 1 }  
ipsAuthGroups OBJECT IDENTIFIER ::= { ipsAuthConformance 2 }

ipsAuthInstanceAttributesGroup OBJECT-GROUP  
OBJECTS {  
    ipsAuthInstDescr,  
    ipsAuthInstStorageType  
}  
STATUS current  
DESCRIPTION  
    "A collection of objects providing information about  
    authorization instances."  
::= { ipsAuthGroups 1 }

ipsAuthIdentAttributesGroup OBJECT-GROUP  
OBJECTS {  
    ipsAuthIdentDescription,  
    ipsAuthIdentRowStatus,  
    ipsAuthIdentStorageType  
}  
STATUS current  
DESCRIPTION  
    "A collection of objects providing information about  
    user identities within an authorization instance."  
::= { ipsAuthGroups 2 }

ipsAuthIdentNameAttributesGroup OBJECT-GROUP  
OBJECTS {  
    ipsAuthIdentName,  
    ipsAuthIdentNameRowStatus,  
    ipsAuthIdentNameStorageType  
}  
STATUS current  
DESCRIPTION  
    "A collection of objects providing information about  
    user names within user identities within an authorization  
    instance."  
::= { ipsAuthGroups 3 }

ipsAuthIdentAddrAttributesGroup OBJECT-GROUP  
OBJECTS {  
    ipsAuthIdentAddrType,  
    ipsAuthIdentAddrStart,  
    ipsAuthIdentAddrEnd,  
}

```
        ipsAuthIdentAddrRowStatus,
        ipsAuthIdentAddrStorageType
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
        address ranges within user identities within an
        authorization instance."
 ::= { ipsAuthGroups 4 }

ipsAuthIdentCredAttributesGroup OBJECT-GROUP
    OBJECTS {
        ipsAuthCredAuthMethod,
        ipsAuthCredRowStatus,
        ipsAuthCredStorageType
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
        credentials within user identities within an authorization
        instance."
 ::= { ipsAuthGroups 5 }

ipsAuthIdentChapAttrGroup OBJECT-GROUP
    OBJECTS {
        ipsAuthCredChapUserName,
        ipsAuthCredChapRowStatus,
        ipsAuthCredChapStorageType
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
        CHAP credentials within user identities within an
        authorization instance."
 ::= { ipsAuthGroups 6 }

ipsAuthIdentSrpAttrGroup OBJECT-GROUP
    OBJECTS {
        ipsAuthCredSrpUserName,
        ipsAuthCredSrpRowStatus,
        ipsAuthCredSrpStorageType
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
        SRP credentials within user identities within an
        authorization instance."
 ::= { ipsAuthGroups 7 }
```

```

ipsAuthIdentKerberosAttrGroup OBJECT-GROUP
    OBJECTS {
        ipsAuthCredKerbPrincipal,
        ipsAuthCredKerbRowStatus,
        ipsAuthCredKerbStorageType
    }
    STATUS current
    DESCRIPTION
        "A collection of objects providing information about
        Kerberos credentials within user identities within an
        authorization instance."
 ::= { ipsAuthGroups 8 }

--*****

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

        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 verify identities."

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

```

GROUP ipsAuthIdentChapAttrGroup

DESCRIPTION

"This group is mandatory for all implementations that use CHAP to help verify identities.

The ipsAuthIdentCredAttributesGroup must be implemented if this group is implemented."

GROUP ipsAuthIdentSrpAttrGroup

DESCRIPTION

"This group is mandatory for all implementations that use SRP to help verify identities.

The ipsAuthIdentCredAttributesGroup must be implemented if this group is implemented."

GROUP ipsAuthIdentKerberosAttrGroup

DESCRIPTION

"This group is mandatory for all implementations that use Kerberos to help verify identities.

The ipsAuthIdentCredAttributesGroup must be implemented if this group is implemented."

OBJECT ipsAuthInstDescr

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthInstStorageType

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthIdentDescription

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthIdentRowStatus

SYNTAX INTEGER { active(1) } -- subset of RowStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required, and only one of the six enumerated values for the RowStatus textual convention need be supported, specifically: active(1)."



OBJECT ipsAuthIdentName

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthIdentNameRowStatus

SYNTAX INTEGER { active(1) } -- subset of RowStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required, and only one of the six enumerated values for the RowStatus textual convention need be supported, specifically: active(1)."

OBJECT ipsAuthIdentAddrType

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthIdentAddrStart

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthIdentAddrEnd

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthIdentAddrRowStatus

SYNTAX INTEGER { active(1) } -- subset of RowStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required, and only one of the six enumerated values for the RowStatus textual convention need be supported, specifically: active(1)."

OBJECT ipsAuthCredAuthMethod

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT ipsAuthCredRowStatus

SYNTAX INTEGER { active(1) } -- subset of RowStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required, and only one of the

six enumerated values for the RowStatus textual convention need be supported, specifically:  
active(1)."

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

OBJECT ipsAuthCredChapRowStatus  
SYNTAX INTEGER { active(1) } -- subset of RowStatus  
MIN-ACCESS read-only  
DESCRIPTION  
    "Write access is not required, and only one of the  
    six enumerated values for the RowStatus textual  
    convention need be supported, specifically:  
    active(1)."

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

OBJECT ipsAuthCredSrpRowStatus  
SYNTAX INTEGER { active(1) } -- subset of RowStatus  
MIN-ACCESS read-only  
DESCRIPTION  
    "Write access is not required, and only one of the  
    six enumerated values for the RowStatus textual  
    convention need be supported, specifically:  
    active(1)."

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

OBJECT ipsAuthCredKerbRowStatus  
SYNTAX INTEGER { active(1) } -- subset of RowStatus  
MIN-ACCESS read-only  
DESCRIPTION  
    "Write access is not required, and only one of the six  
    enumerated values for the RowStatus textual convention need  
    be supported, specifically: active(1)."

::= { ipsAuthCompliances 1 }

END

## 9. Security Considerations

### 9.1. MIB Security Considerations

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

- o in the `ipsAuthInstanceAttributesTable`:
  - `ipsAuthInstDescr` could be modified to camouflage the existence of a rogue authorization instance;
- o in the `ipsAuthIdentAttributesTable`:
  - `ipsAuthIdentDescription` could be modified to camouflage the existence of a rogue identity;
  - `ipsAuthIdentRowStatus` could be modified to add or delete a rogue identity;
  - `ipsAuthIdentStorageType` could be modified to make temporary rows permanent, or permanent rows temporary;
- o in the `ipsAuthIdentNameAttributesTable`:
  - `ipsAuthIdentName` could be modified to change the name of an existing identity;
  - `ipsAuthIdentNameRowStatus` could be modified to add or delete a name of an existing identity;
  - `ipsAuthIdentNameStorageType` could be modified to make temporary rows permanent, or permanent rows temporary;
- o in the `ipsAuthIdentAddrAttributesTable`:
  - `ipsAuthIdentAddrType` could be modified to change the type of address checking performed;
  - `ipsAuthIdentAddrStart` could be modified to change the start of the allowed range;

- ipsAuthIdentAddrEnd could be modified to change the end of the allowed range;
- ipsAuthIdentAddrRowStatus could be modified to add or delete the checking of an address range;
- ipsAuthIdentAddrStorageType could be modified to make temporary rows permanent, or permanent rows temporary;
- o in the ipsAuthCredentialAttributesTable:
  - ipsAuthCredAuthMethod could be modified to change the type of authentication to be used;
  - ipsAuthCredRowStatus could be modified to add or delete checking of credentials;
  - ipsAuthCredStorageType could be modified to make temporary rows permanent, or permanent rows temporary;
- o in the ipsAuthCredChapAttributesTable:
  - ipsAuthCredChapUserName could be modified to change the CHAP user name for a credential;
  - ipsAuthCredChapRowStatus could be modified to add or delete CHAP attributes for credentials;
  - ipsAuthCredChapStorageType could be modified to make temporary rows permanent, or permanent rows temporary;
- o in the ipsAuthCredSrpAttributesTable:
  - ipsAuthCredSrpUserName could be modified to change the SRP user name for a credential;
  - ipsAuthCredSrpRowStatus could be modified to add or delete SRP attributes for credentials;
  - ipsAuthCredSrpStorageType could be modified to make temporary rows permanent, or permanent rows temporary;
- o in the ipsAuthCredKerbAttributesTable:
  - ipsAuthCredKerbPrincipal could be modified to change the Kerberos principal for a credential;

- ipsAuthCredKerbRowStatus could be modified to add or delete Kerberos attributes for credentials;
- ipsAuthCredKerbStorageType could be modified to make temporary rows permanent, or permanent rows temporary;

Note that removal of legitimate credentials can result in either denial of service or weakening the requirements for access of a particular service. Note also that some types of credentials, such as CHAP or SRP, also require passwords or verifiers to be associated with the credential. These are managed outside this MIB module.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- o All tables (specifically: ipsAuthInstanceAttributesTable, ipsAuthIdentAttributesTable, ipsAuthIdentNameAttributesTable, ipsAuthIdentAddrAttributesTable, ipsAuthCredentialAttributesTable, ipsAuthCredChapAttributesTable, ipsAuthCredSrpAttributesTable, and ipsAuthCredKerbAttributesTable) provide the ability to find out which names, addresses, and credentials would be required to access services on the managed system. If these credentials are easily spoofed (particularly the name or address), read access to this MIB module must be tightly controlled. When used with pointers from another MIB module to rows in the ipsAuthIdentAttributesTable, this MIB module provides information about which entities are authorized to connect to which entities.

SNMP versions prior to SNMPv3 did not include adequate security. 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 module.

It is RECOMMENDED that implementors consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an

instance of this MIB module 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.

In many implementations, the objects in this MIB module can be read and modified via other mechanisms or protocols in addition to this MIB module. For the system to be secure, other mechanisms that can read and modify the contents of this MIB module must also address the above issues, and handle the threats outlined in [RFC3411], section 1.4.

Given the sensitivity of information contained in this MIB module, it is strongly recommended that encryption (SNMPv3 with a securityLevel of authPriv [RFC3411]) be used for all access to objects in this MIB module.

## 9.2. Other Security Considerations

An identity consists of a set of names (e.g., an iSCSI Initiator Name), addresses (e.g., an IP address or Fibre Channel World Wide Name (WWN)), and credentials (e.g., a CHAP user name).

To match an identity, one must match:

- o One of the IdentNames belonging to the IdentIndex, unless there are no IdentNames for the IdentIndex, and
- o One of the IdentAddrs belonging to the IdentIndex, unless there are no IdentAddrs for the IdentIndex, and
- o One of the IdentCreds belonging to the IdentIndex, unless there are no Creds for the IdentIndex.

Note that if any of the above lists are empty for a given IdentIndex, any identifier of that type is considered to match the identity. The non-empty lists will still be checked. For example, if the IdentAddrs list is empty for the IdentIndex, but there are entries in IdentNames and IdentCreds, any address will be considered a match, as long as the offered name and credential match one of the IdentNames and IdentCreds, respectively.

This leaves a possible security window while adding and removing entries from one of these lists. For example, an identity could consist of no IdentNames, no IdentAddrs, and exactly one IdentCred. If that IdentCred was to be updated, several methods could be used:

- o The UserName or Principal could be simply written in the appropriate table, if the credential's type remained the same (recommended).
- o The new credential could be added, then the old deleted (recommended).
- o The new credential could be added, and the old deleted in the same SNMP request (recommended, but do the add first).
- o The old credential could be deleted, then the new added (Don't use!).

Of the above methods, the last leaves a window in which the list is empty, possibly allowing unconstrained access to the resource making use of this MIB. This method should never be used for Names, Addrs, or Creds.

The use of the third method, adding and deleting within the same request, should be used with care. It is recommended that within the request, the add be done first. Otherwise, an implementation may attempt to perform these operations in order, potentially leaving a window.

The first two methods are recommended.

Care must also be taken when updating the IdentAddr for an identity. Each IdentAddr specifies a range of addresses that match the identity, and has an address type, starting address, and ending address. Modifying these one at a time can open a temporary window where a larger range of addresses are allowed. For example, a single address is specified using IdentAddrType = ipv4, IdentAddrStart = IdentAddrEnd = 192.0.2.5. We want to update this to specify the single address 192.0.2.34. If the end address is updated first, we temporarily allow the range 192.0.2.5 .. 192.0.2.34, which is not what we want. Similarly, if we change from 192.0.2.34 back to 192.0.2.5, and we update IdentAddrStart first, we end up with the range again. To handle this, an application must either:

- o update both IdentAddrStart and IdentAddrEnd in the same SNMP set request, or
- o add the new IdentAddrStart and IdentAddrEnd with a new IdentAddrIndex, then delete the old one, using the methods shown before.

Since the value of `IdentAddrType` specifies the formats of `IdentAddrStart` and `IdentAddrEnd`, modification of `IdentAddrType` is not allowed for an existing row.

## 10. IANA Considerations

The IANA has assigned a MIB OID number under the `mib-2` branch for the `IPS-AUTH-MIB`.

## 11. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [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.
- [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", RFC 3411, December 2002.
- [RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.
- [IANA-AF] IANA, "IANA Address Family Numbers MIB", <http://www.iana.org/assignments/ianaaddressfamilynumbers-mib>.
- [RFC4293] Routhier, S., "Management Information Base for the Internet Protocol (IP)", RFC 4293, April 2006.
- [RFC1994] Simpson, W., "PPP Challenge Handshake Authentication Protocol (CHAP)", RFC 1994, August 1996.



- [RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.
- [RFC2945] Wu, T., "The SRP Authentication and Key Exchange System", RFC 2945, September 2000.

## 12. Informative References

- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 3414, December 2002.
- [RFC3720] Satran, J., Meth, K., Sapuntzakis, C., Chadalapaka, M., and E. Zeidner, "Internet Small Computer Systems Interface (iSCSI)", RFC 3720, March 2004.
- [RFC1737] Sollins, K. and L. Masinter, "Functional Requirements for Uniform Resource Names", RFC 1737, December 1994.
- [RFC4044] McCloghrie, K., "Fibre Channel Management MIB", RFC 4044, May 2005.

## 13. Acknowledgements

In addition to the authors, several people contributed to the development of this MIB module through discussions of authentication, authorization, and access within the iSCSI MIB module and security teams, including John Hufferd, Marjorie Krueger, Keith McCloghrie, Tom McSweeney, Steve Senum, and Josh Tseng. Thanks also to Bill Studenmund (Wasabi Systems) for adding the Kerberos method, and to Ayman Ghanem for finding and suggesting changes to several problems found in the MIB module.

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

## Authors' Addresses

Mark Bakke  
Postal: Cisco Systems, Inc  
7900 International Drive, Suite 400  
Bloomington, MN  
USA 55425

EMail: mbakke@cisco.com

James Muchow  
Postal: Qlogic Corp.  
6321 Bury Drive  
Eden Prairie, MN  
USA 55346

EMail: james.muchow@qlogic.com

## Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at [ietf-ipr@ietf.org](mailto:ietf-ipr@ietf.org).

## Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

