

Network Working Group  
Request for Comments: 4113  
Obsoletes: 2454, 2013  
Category: Standards Track

B. Fenner  
AT&T Labs - Research  
J. Flick  
Hewlett-Packard Company  
June 2005

## Management Information Base for the User Datagram Protocol (UDP)

### 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 (2005).

### Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects used for implementations of the User Datagram Protocol (UDP) in an IP version independent manner. This memo obsoletes RFCs 2013 and 2454.

### Table of Contents

|        |                                                                |    |
|--------|----------------------------------------------------------------|----|
| 1.     | The Internet-Standard Management Framework . . . . .           | 2  |
| 2.     | Overview . . . . .                                             | 2  |
| 2.1.   | Relationship to Other MIBs . . . . .                           | 3  |
| 2.1.1. | Relationship to RFC1213-MIB . . . . .                          | 3  |
| 2.1.2. | Relationship to the IPV6-UDP-MIB . . . . .                     | 3  |
| 2.1.3. | Relationship to HOST-RESOURCES-MIB and<br>SYSAPPL-MIB. . . . . | 4  |
| 3.     | Definitions . . . . .                                          | 4  |
| 4.     | Acknowledgements . . . . .                                     | 15 |
| 5.     | Contributors . . . . .                                         | 15 |
| 6.     | Security Considerations . . . . .                              | 16 |
| 7.     | IANA Considerations . . . . .                                  | 17 |
| 8.     | References . . . . .                                           | 17 |
| 8.1.   | Normative References . . . . .                                 | 17 |
| 8.2.   | Informative References . . . . .                               | 18 |

## 1. 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].

## 2. Overview

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects used for implementations of the User Datagram Protocol (UDP), as defined in RFC 768 [RFC0768], in an IP version independent manner.

The current UDP-MIB defined in this memo consists of one table and a group of scalars:

- o The udp group of scalars reports parameters and statistics of a UDP protocol engine. Two scalars, udpHCInDatagrams and udpHCOutDatagrams, have been added to this group since the publication of RFC 2013 [RFC2013] in order to provide high-capacity counters for fast networks. Discontinuities in the values of the counters in this group are indicated by discontinuities in the value of the sysUpTime object, which is defined in RFC 3418 [RFC3418].
- o The udpEndpointTable provides access to status information for all UDP endpoints handled by a UDP protocol engine. The table provides for strictly listening endpoints, as with the historical udpTable, and also for "connected" UDP endpoints, which only accept packets from a given remote system. It also reports identification of the operating system level processes that handle UDP connections. Addresses and ports of UDP endpoints in this table are represented using the InetAddressType, InetAddress, and InetPortNumber textual conventions defined in RFC 4001 [RFC4001].

## 2.1. Relationship to Other MIBs

This section discusses the relationship of this UDP-MIB module to other MIB modules.

### 2.1.1. Relationship to RFC1213-MIB

UDP related MIB objects were originally defined as part of the RFC1213-MIB, defined in RFC 1213 [RFC1213]. The UDP related objects of the RFC1213-MIB were later copied into a separate MIB module and published in RFC 2013 [RFC2013] in SMIV2 format.

The previous versions of the UDP-MIB both defined the `udpTable`, which has been deprecated for basically two reasons:

- (1) The `udpTable` only supports IPv4.

The current approach in the IETF is to write IP version neutral MIBs rather than have different definitions for various version of IP. This reduces the amount of overhead when new objects are introduced, since there is only one place to add them. Hence, the approach taken in RFC 2454 [RFC2454] of having separate tables is not continued.

- (2) The `udpTable` does not permit describing "connected" UDP endpoints.

It turns out that "connected" endpoints tend to have a different behaviour and management access pattern from those of listening endpoints. Adding remote endpoint information to the `udpEndpointTable` thus allows for the addition of specific status and statistic objects for "connected" endpoints and connections.

### 2.1.2. Relationship to the IPV6-UDP-MIB

The IPV6-UDP-MIB, defined in RFC 2454 [RFC2454], has been moved to Historic because the approach of having separate IP version specific tables is not followed anymore. Implementation of RFC 2454 is thus not suggested anymore.

Note that because scoped addresses are now represented using the `IPv4z` and `IPv6z` address types, there is no longer a need to explicitly include the `ifIndex` in the index clause of the `udpEndpointTable`. This is a change from the use of `ipv6UdpIfIndex` in RFC 2454.

### 2.1.3. Relationship to HOST-RESOURCES-MIB and SYSAPPL-MIB

The udpEndpointTable reports the identification of the operating system level process that handles a connection or a listening endpoint. The value is reported as an Unsigned32, which is expected to be the same as the hrSWRunIndex of the HOST-RESOURCES-MIB [RFC2790] (if the value is smaller than 2147483647) or the sysAppElmtRunIndex of the SYSAPPL-MIB [RFC2287]. This allows management applications to identify the UDP connections that belong to an operating system level process, which has proven valuable in operational environments.

## 3. Definitions

UDP-MIB DEFINITIONS ::= BEGIN

### IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, Integer32, Counter32, Counter64,  
 Unsigned32, IpAddress, mib-2 FROM SNMPv2-SMI  
 MODULE-COMPLIANCE, OBJECT-GROUP FROM SNMPv2-CONF  
 InetAddress, InetAddressType,  
 InetPortNumber FROM INET-ADDRESS-MIB;

### udpMIB MODULE-IDENTITY

LAST-UPDATED "200505200000Z" -- May 20, 2005

#### ORGANIZATION

"IETF IPv6 Working Group  
<http://www.ietf.org/html.charters/ipv6-charter.html>"

#### CONTACT-INFO

"Bill Fenner (editor)

AT&T Labs -- Research  
 75 Willow Rd.  
 Menlo Park, CA 94025

Phone: +1 650 330-7893  
 Email: <fenner@research.att.com>

John Flick (editor)

Hewlett-Packard Company  
 8000 Foothills Blvd. M/S 5557  
 Roseville, CA 95747

Phone: +1 916 785 4018  
 Email: <john.flick@hp.com>

Send comments to <ipv6@ietf.org>"

## DESCRIPTION

"The MIB module for managing UDP implementations.  
Copyright (C) The Internet Society (2005). This  
version of this MIB module is part of RFC 4113;  
see the RFC itself for full legal notices."

REVISION "200505200000Z" -- May 20, 2005

## DESCRIPTION

"IP version neutral revision, incorporating the  
following revisions:

- Added udpHCInDatagrams and udpHCOutDatagrams in order  
to provide high-capacity counters for fast networks.
- Added text to the descriptions of all counter objects  
to indicate how discontinuities are detected.
- Deprecated the IPv4-specific udpTable and replaced it  
with the version neutral udpEndpointTable. This  
table includes support for connected UDP endpoints  
and support for identification of the operating  
system process associated with a UDP endpoint.
- Deprecated the udpGroup and replaced it with object  
groups representing the current set of objects.
- Deprecated udpMIBCompliance and replaced it with  
udpMIBCompliance2, which includes the compliance  
information for the new object groups.

This version published as RFC 4113."

REVISION "199411010000Z" -- November 1, 1994

## DESCRIPTION

"Initial SMIV2 version, published as RFC 2013."

REVISION "199103310000Z" -- March 31, 1991

## DESCRIPTION

"The initial revision of this MIB module was part of  
MIB-II, published as RFC 1213."

::= { mib-2 50 }

-- the UDP group

udp OBJECT IDENTIFIER ::= { mib-2 7 }

udpInDatagrams OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The total number of UDP datagrams delivered to UDP  
users."

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by discontinuities in the value of sysUpTime."

::= { udp 1 }

udpNoPorts OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of received UDP datagrams for which there was no application at the destination port.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by discontinuities in the value of sysUpTime."

::= { udp 2 }

udpInErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by discontinuities in the value of sysUpTime."

::= { udp 3 }

udpOutDatagrams OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of UDP datagrams sent from this entity.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by discontinuities in the value of sysUpTime."

::= { udp 4 }

## udpHCInDatagrams OBJECT-TYPE

SYNTAX Counter64

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The total number of UDP datagrams delivered to UDP users, for devices that can receive more than 1 million UDP datagrams per second.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by discontinuities in the value of sysUpTime."

::= { udp 8 }

## udpHCOutDatagrams OBJECT-TYPE

SYNTAX Counter64

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The total number of UDP datagrams sent from this entity, for devices that can transmit more than 1 million UDP datagrams per second.

Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by discontinuities in the value of sysUpTime."

::= { udp 9 }

--

-- { udp 6 } was defined as the ipv6UdpTable in RFC2454's  
-- IPV6-UDP-MIB. This RFC obsoletes RFC 2454, so { udp 6 } is  
-- obsoleted.

--

-- The UDP "Endpoint" table.

## udpEndpointTable OBJECT-TYPE

SYNTAX SEQUENCE OF UdpEndpointEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"A table containing information about this entity's UDP endpoints on which a local application is currently accepting or sending datagrams.

The address type in this table represents the address type used for the communication, irrespective of the higher-layer abstraction. For example, an application using IPv6 'sockets' to communicate via IPv4 between `::ffff:10.0.0.1` and `::ffff:10.0.0.2` would use `InetAddressType ipv4(1)`.

Unlike the `udpTable` in RFC 2013, this table also allows the representation of an application that completely specifies both local and remote addresses and ports. A listening application is represented in three possible ways:

- 1) An application that is willing to accept both IPv4 and IPv6 datagrams is represented by a `udpEndpointLocalAddressType` of `unknown(0)` and a `udpEndpointLocalAddress` of `'h` (a zero-length octet-string).
- 2) An application that is willing to accept only IPv4 or only IPv6 datagrams is represented by a `udpEndpointLocalAddressType` of the appropriate address type and a `udpEndpointLocalAddress` of `'0.0.0.0'` or `:::` respectively.
- 3) An application that is listening for datagrams only for a specific IP address but from any remote system is represented by a `udpEndpointLocalAddressType` of the appropriate address type, with `udpEndpointLocalAddress` specifying the local address.

In all cases where the remote is a wildcard, the `udpEndpointRemoteAddressType` is `unknown(0)`, the `udpEndpointRemoteAddress` is `'h` (a zero-length octet-string), and the `udpEndpointRemotePort` is 0.

If the operating system is demultiplexing UDP packets by remote address and port, or if the application has 'connected' the socket specifying a default remote address and port, the `udpEndpointRemote*` values should be used to reflect this."

`::= { udp 7 }`

```
udpEndpointEntry OBJECT-TYPE
    SYNTAX      UdpEndpointEntry
    MAX-ACCESS  not-accessible
    STATUS      current
```

## DESCRIPTION

"Information about a particular current UDP endpoint.

Implementers need to be aware that if the total number of elements (octets or sub-identifiers) in udpEndpointLocalAddress and udpEndpointRemoteAddress exceeds 111, then OIDs of column instances in this table will have more than 128 sub-identifiers and cannot be accessed using SNMPv1, SNMPv2c, or SNMPv3."

INDEX { udpEndpointLocalAddressType,  
udpEndpointLocalAddress,  
udpEndpointLocalPort,  
udpEndpointRemoteAddressType,  
udpEndpointRemoteAddress,  
udpEndpointRemotePort,  
udpEndpointInstance }  
::= { udpEndpointTable 1 }

UdpEndpointEntry ::= SEQUENCE {  
udpEndpointLocalAddressType InetAddressType,  
udpEndpointLocalAddress InetAddress,  
udpEndpointLocalPort InetPortNumber,  
udpEndpointRemoteAddressType InetAddressType,  
udpEndpointRemoteAddress InetAddress,  
udpEndpointRemotePort InetPortNumber,  
udpEndpointInstance Unsigned32,  
udpEndpointProcess Unsigned32  
}

## udpEndpointLocalAddressType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"The address type of udpEndpointLocalAddress. Only IPv4, IPv4z, IPv6, and IPv6z addresses are expected, or unknown(0) if datagrams for all local IP addresses are accepted."

::= { udpEndpointEntry 1 }

## udpEndpointLocalAddress OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"The local IP address for this UDP endpoint.

The value of this object can be represented in three

possible ways, depending on the characteristics of the listening application:

1. For an application that is willing to accept both IPv4 and IPv6 datagrams, the value of this object must be ''h (a zero-length octet-string), with the value of the corresponding instance of the udpEndpointLocalAddressType object being unknown(0).
2. For an application that is willing to accept only IPv4 or only IPv6 datagrams, the value of this object must be '0.0.0.0' or ':::', respectively, while the corresponding instance of the udpEndpointLocalAddressType object represents the appropriate address type.
3. For an application that is listening for data destined only to a specific IP address, the value of this object is the specific IP address for which this node is receiving packets, with the corresponding instance of the udpEndpointLocalAddressType object representing the appropriate address type.

As this object is used in the index for the udpEndpointTable, implementors of this table should be careful not to create entries that would result in OIDs with more than 128 subidentifiers; else the information cannot be accessed using SNMPv1, SNMPv2c, or SNMPv3."

::= { udpEndpointEntry 2 }

#### udpEndpointLocalPort OBJECT-TYPE

SYNTAX InetPortNumber

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The local port number for this UDP endpoint."

::= { udpEndpointEntry 3 }

#### udpEndpointRemoteAddressType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The address type of udpEndpointRemoteAddress. Only IPv4, IPv4z, IPv6, and IPv6z addresses are expected, or unknown(0) if datagrams for all remote IP addresses are accepted. Also, note that some combinations of

udpEndpointLocalAdresType and  
 udpEndpointRemoteAddressType are not supported. In  
 particular, if the value of this object is not  
 unknown(0), it is expected to always refer to the  
 same IP version as udpEndpointLocalAddressType."  
 ::= { udpEndpointEntry 4 }

#### udpEndpointRemoteAddress OBJECT-TYPE

SYNTAX InetAddress  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION

"The remote IP address for this UDP endpoint. If  
 datagrams from any remote system are to be accepted,  
 this value is ''h (a zero-length octet-string).  
 Otherwise, it has the type described by  
 udpEndpointRemoteAddressType and is the address of the  
 remote system from which datagrams are to be accepted  
 (or to which all datagrams will be sent).

As this object is used in the index for the  
 udpEndpointTable, implementors of this table should be  
 careful not to create entries that would result in OIDs  
 with more than 128 subidentifiers; else the information  
 cannot be accessed using SNMPv1, SNMPv2c, or SNMPv3."

::= { udpEndpointEntry 5 }

#### udpEndpointRemotePort OBJECT-TYPE

SYNTAX InetPortNumber  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION

"The remote port number for this UDP endpoint. If  
 datagrams from any remote system are to be accepted,  
 this value is zero."

::= { udpEndpointEntry 6 }

#### udpEndpointInstance OBJECT-TYPE

SYNTAX Unsigned32 (1..'ffffffff'h)  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION

"The instance of this tuple. This object is used to  
 distinguish among multiple processes 'connected' to  
 the same UDP endpoint. For example, on a system  
 implementing the BSD sockets interface, this would be  
 used to support the SO\_REUSEADDR and SO\_REUSEPORT  
 socket options."

```

    ::= { udpEndpointEntry 7 }

udpEndpointProcess OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The system's process ID for the process associated with
        this endpoint, or zero if there is no such process.
        This value is expected to be the same as
        HOST-RESOURCES-MIB::hrSWRunIndex or SYSAPPL-MIB::
        sysAppElmtRunIndex for some row in the appropriate
        tables."
    ::= { udpEndpointEntry 8 }

-- The deprecated UDP Listener table

-- The deprecated UDP listener table only contains information
-- about this entity's IPv4 UDP end-points on which a local
-- application is currently accepting datagrams. It does not
-- provide more detailed connection information, or information
-- about IPv6 endpoints.

udpTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF UdpEntry
    MAX-ACCESS  not-accessible
    STATUS      deprecated
    DESCRIPTION
        "A table containing IPv4-specific UDP listener
        information. It contains information about all local
        IPv4 UDP end-points on which an application is
        currently accepting datagrams. This table has been
        deprecated in favor of the version neutral
        udpEndpointTable."
    ::= { udp 5 }

udpEntry OBJECT-TYPE
    SYNTAX      UdpEntry
    MAX-ACCESS  not-accessible
    STATUS      deprecated
    DESCRIPTION
        "Information about a particular current UDP listener."
    INDEX      { udpLocalAddress, udpLocalPort }
    ::= { udpTable 1 }

UdpEntry ::= SEQUENCE {
    udpLocalAddress  IpAddress,
    udpLocalPort     Integer32

```

```
}
```

```
udpLocalAddress OBJECT-TYPE
```

```
    SYNTAX      IpAddress
```

```
    MAX-ACCESS  read-only
```

```
    STATUS      deprecated
```

```
    DESCRIPTION
```

```
        "The local IP address for this UDP listener.  In the
        case of a UDP listener that is willing to accept
        datagrams for any IP interface associated with the
        node, the value 0.0.0.0 is used."
```

```
    ::= { udpEntry 1 }
```

```
udpLocalPort OBJECT-TYPE
```

```
    SYNTAX      Integer32 (0..65535)
```

```
    MAX-ACCESS  read-only
```

```
    STATUS      deprecated
```

```
    DESCRIPTION
```

```
        "The local port number for this UDP listener."
```

```
    ::= { udpEntry 2 }
```

```
-- conformance information
```

```
udpMIBConformance OBJECT IDENTIFIER ::= { udpMIB 2 }
```

```
udpMIBCompliances OBJECT IDENTIFIER ::= { udpMIBConformance 1 }
```

```
udpMIBGroups      OBJECT IDENTIFIER ::= { udpMIBConformance 2 }
```

```
-- compliance statements
```

```
udpMIBCompliance2 MODULE-COMPLIANCE
```

```
    STATUS      current
```

```
    DESCRIPTION
```

```
        "The compliance statement for systems that implement
        UDP."
```

```
        There are a number of INDEX objects that cannot be
        represented in the form of OBJECT clauses in SMIV2, but
        for which we have the following compliance
        requirements, expressed in OBJECT clause form in this
        description clause:
```

```
        -- OBJECT      udpEndpointLocalAddressType
        -- SYNTAX      InetAddressType { unknown(0), ipv4(1),
        --                                     ipv6(2), ipv4z(3),
        --                                     ipv6z(4) }
        -- DESCRIPTION
        --      Support for dns(5) is not required.
        -- OBJECT      udpEndpointLocalAddress
```

```

-- SYNTAX      InetAddress (SIZE(0|4|8|16|20))
-- DESCRIPTION
--      Support is only required for zero-length
--      octet-strings, and for scoped and unscoped
--      IPv4 and IPv6 addresses.
-- OBJECT      udpEndpointRemoteAddressType
-- SYNTAX      InetAddressType { unknown(0), ipv4(1),
--                               ipv6(2), ipv4z(3),
--                               ipv6z(4) }
-- DESCRIPTION
--      Support for dns(5) is not required.
-- OBJECT      udpEndpointRemoteAddress
-- SYNTAX      InetAddress (SIZE(0|4|8|16|20))
-- DESCRIPTION
--      Support is only required for zero-length
--      octet-strings, and for scoped and unscoped
--      IPv4 and IPv6 addresses.
"
MODULE -- this module
    MANDATORY-GROUPS { udpBaseGroup, udpEndpointGroup }
    GROUP      udpHCGroup
    DESCRIPTION
        "This group is mandatory for systems that
        are capable of receiving or transmitting more than
        1 million UDP datagrams per second. 1 million
        datagrams per second will cause a Counter32 to
        wrap in just over an hour."
    ::= { udpMIBCompliances 2 }

udpMIBCompliance MODULE-COMPLIANCE
    STATUS      deprecated
    DESCRIPTION
        "The compliance statement for IPv4-only systems that
        implement UDP. For IP version independence, this
        compliance statement is deprecated in favor of
        udpMIBCompliance2. However, agents are still
        encouraged to implement these objects in order to
        interoperate with the deployed base of managers."
    MODULE -- this module
        MANDATORY-GROUPS { udpGroup }
        ::= { udpMIBCompliances 1 }

-- units of conformance

udpGroup OBJECT-GROUP
    OBJECTS      { udpInDatagrams, udpNoPorts,
                  udpInErrors, udpOutDatagrams,
                  udpLocalAddress, udpLocalPort }

```

```
STATUS      deprecated
DESCRIPTION
    "The deprecated group of objects providing for
    management of UDP over IPv4."
 ::= { udpMIBGroups 1 }

udpBaseGroup OBJECT-GROUP
OBJECTS      { udpInDatagrams, udpNoPorts, udpInErrors,
               udpOutDatagrams }
STATUS      current
DESCRIPTION
    "The group of objects providing for counters of UDP
    statistics."
 ::= { udpMIBGroups 2 }

udpHCGroup OBJECT-GROUP
OBJECTS      { udpHCInDatagrams, udpHCOutDatagrams }
STATUS      current
DESCRIPTION
    "The group of objects providing for counters of high
    speed UDP implementations."
 ::= { udpMIBGroups 3 }

udpEndpointGroup OBJECT-GROUP
OBJECTS      { udpEndpointProcess }
STATUS      current
DESCRIPTION
    "The group of objects providing for the IP version
    independent management of UDP 'endpoints'."
 ::= { udpMIBGroups 4 }

END
```

#### 4. Acknowledgements

This document contains a modified subset of RFC 1213 and replaces RFCs 2013 and 2454. Acknowledgments are therefore due to the authors and editors of these documents for their excellent work.

#### 5. Contributors

This document is an output of the IPv6 MIB revision team, and contributors to earlier versions of this document include:

Bill Fenner, AT&T Labs -- Research  
Email: fenner@research.at.com

Brian Haberman  
Email: brian@innovationslab.net

Shawn A. Routhier, Wind River  
Email: sar@epilogue.com

Juergen Schoenwalder, TU Braunschweig  
Email: schoenw@ibr.cs.tu-bs.de

Dave Thaler, Microsoft  
Email: dthaler@windows.microsoft.com

Much of Keith McCloghrie's text from RFC1213/RFC2013 remains in this document, and the structure of the MIB is due to him.

Mike Daniele wrote the original IPv6 UDP MIB in RFC2454.

Juergen Schoenwalder provided much of the text for section 2.

## 6. Security Considerations

There are no management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. So, if this MIB is implemented correctly, then there is no risk that an intruder can alter or create any management objects of this MIB module via direct SNMP SET operations.

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:

The indices of the udpEndpointTable and udpTable contain information on the listeners on an entity. In particular, the udpEndpointLocalPort and udpLocalPort objects in the indices can be used to identify what ports are open on the machine and what attacks are likely to succeed, without the attacker having to run a port scanner.

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 the 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).

Furthermore, 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.

## 7. IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values, recorded in the SMI Numbers registry:

| +-----+<br>  Descriptor | +-----+<br>  OBJECT IDENTIFIER value |
|-------------------------|--------------------------------------|
| +-----+                 | +-----+                              |
| udp                     | { mib-2 7 }                          |
| udpMIB                  | { mib-2 50 }                         |
| +-----+                 | +-----+                              |

## 8. References

### 8.1. Normative References

- [RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.
- [RFC2578] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

- [RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

## 8.2. Informative References

- [RFC1213] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets:MIB-II", STD 17, RFC 1213, March 1991.
- [RFC2013] McCloghrie, K., "SNMPv2 Management Information Base for the User Datagram Protocol using SMIV2", RFC 2013, November 1996.
- [RFC2287] Krupczak, C. and J. Saperia, "Definitions of System-Level Managed Objects for Applications", RFC 2287, February 1998.
- [RFC2454] Daniele, M., "IP Version 6 Management Information Base for the User Datagram Protocol", RFC 2454, December 1998.
- [RFC2790] Waldbusser, S. and P. Grillo, "Host Resources MIB", RFC 2790, March 2000.
- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

## Authors' Addresses

Bill Fenner  
AT&T Labs -- Research  
75 Willow Rd  
Menlo Park, CA 94025  
USA

Email: fenner@research.att.com

John Flick  
Hewlett-Packard Company  
8000 Foothills Blvd. M/S 5557  
Roseville, CA 95747-5557  
USA

Email: john.flick@hp.com

## Full Copyright Statement

Copyright (C) The Internet Society (2005).

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 currently provided by the Internet Society.

