Network Working Group
Internet-Draft
Expires: March 21, 2002

M. Daniele
Consultant
J. Schoenwaelder
TU Braunschweig
September 20, 2001

Textual Conventions for Transport Addresses
draft-ops-taddress-mib-01.txt

Status of this Memo

Copyright Notice

Abstract

This document introduces a MIB module which defines textual conventions to represent commonly used transport layer addressing information. The definitions are compatible with the concept of TAddress/TDomain pairs introduced by the SMIv2 and support the Internet transport protocols over IPv4 and IPv6.

Table of Contents

1. Introduction

Many MIB modules have the need to represent transport layer addresses in a generic way. Typical examples are MIBs for application protocols that can run over several different transports or application management MIBs that need to model generic communication endpoints.

The SMIv2 defines in RFC 2579 [7] the textual conventions TDomain and TAddress to represent generic transports layer endpoints. A generic TAddress value is interpreted in a given transport domain which is identified by a TDomain value. The TDomain is an object identifier which allows MIB authors to extend the set of supported transport domains by providing suitable definitions in enterprise specific MIB modules.

A set of TDomain values and concrete TAddress formats has been standardized in RFC 1906 [11]. These definitions are however mixed up with SNMP semantics and definitions for Internet transport protocols over IPv4 and IPv6 are missing.

The purpose of this memo is to introduce a set of well-known textual conventions to represent commonly used transport layer addressing information which is compatible with the original TDomain and TAddress approach and which includes definitions for Internet transport protocols over IPv4 and IPv6. This memo also defines a new textual convention which enumerates the well-known transport domains since such an enumeration provides in many cases enough flexibility and is more efficient compared to object identifiers.

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT" and "MAY" in this document are to be interpreted as described in RFC 2119 [1].

2. SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [2].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in STD 16, RFC 1155 [3], STD 16, RFC 1212 [4] and RFC 1215 [5]. The second version, called SMIv2, is described in STD 58, RFC 2578 [6], STD 58, RFC 2579 [7] and STD 58, RFC 2580 [8].
- Message protocols for transferring management information. The

first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [9]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [10] and RFC 1906 [11]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [11], RFC 2572 [12] and RFC 2574 [13].

- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [9]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [14].

- A set of fundamental applications described in RFC 2573 [15] and the view-based access control mechanism described in RFC 2575 [16].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [17].

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

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

3. Overview

This MIB module contains definitions for commonly used transport layer addressing information. In particular, it provides the following definitions:

1. Textual conventions for generic transport addresses and generic transport domains.

2. Object identifier registrations for well-known transport domains.

3. An enumeration of the well-known transport domains, called a transport address type.

4. A set of textual conventions for the address formats used by well-known transport domains.

It is expected that this MIB module will be updated by subsequent RFCs which register additional well-known transport domains and which introduce new textual conventions for the address formats used by those new transport domains.

This module does NOT define the transport mappings of any particular protocol. Rather, it defines a set of common identifiers and textual conventions that are intended to be used within various transport mappings documents.

3.1 Relationship to Other MIBs

This section discusses how the definitions provided by this memo relate to definitions in related MIB modules.

3.1.1 SNMPv2-TC (TAddress, TDomain)

The SNMPv2-TC module [7] defines the textual conventions TAddress and TDomain to represent generic transport addresses.

A TAddress is an octet string with a size between 1 and 255 octets. Experience has shown that there is sometimes a need to represent unknown transport addresses. This module therefore introduces a new textual convention TransportAddress which is an octet string with a size between 0 and 255 octets and otherwise identical semantics.

This module also introduces a new textual convention TransportDomain which is compatible with the TDomain definition so that a complete set of definitions is contained in a single module.

3.1.2 SNMPv2-TM

The transport domains defined in the SNMPv2-TM module [11] all contain "snmp" as the prefix in their name and are registered under `snmpDomains' (from RFC 2578 [6]). There has been some confusion as to whether these definitions are appropriate for designating transport endpoints for non-SNMP traffic. These definitions are also incomplete since new transport address domains are needed to support (at least) SNMP over IPv6.

3.1.3 INET-ADDRESS-MIB (InetAddressType, InetAddress)

The INET-ADDRESS-MIB module [18] defines the textual conventions InetAddressType and InetAddress to represent Internet network layer endpoints. Several MIBs use these textual conventions in conjunction

with the InetPortNumber textual convention to represent Internet transport layer endpoints. This is approach is fine as long as a MIB models protocols or applications that are specific to the Internet suite of transport protocols. For protocols or applications that can potentially use other transport protocols, the use of the definitions contained in this memo is more appropriate.

4. Definitions

```
TRANSPORT-ADDRESS-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-IDENTITY, mib-2     FROM SNMPv2-SMI
    TEXTUAL-CONVENTION                          FROM SNMPv2-TC;

transportAddressMIB MODULE-IDENTITY
    LAST-UPDATED "200109170000Z"
    ORGANIZATION
        "IETF Operations and Management Area"
    CONTACT-INFO
        "Juergen Schoenwaelder (Editor)
         TU Braunschweig
         Bueltenweg 74/75
         38106 Braunschweig, Germany

         Phone: +49 531 391-3289
         EMail: schoenw@ibr.cs.tu-bs.de

         Send comments to <mibs@ops.ietf.org>."
    DESCRIPTION
        "This MIB module provides commonly-used transport
         address definitions."
    REVISION    "200109170000Z"
    DESCRIPTION
        "Initial version, published as RFC XXXX."
    ::= { mib-2 XXXX } -- to be assigned by IANA


transportDomains OBJECT IDENTIFIER ::= { transportAddressMIB 1 }

transportDomainUdpIpv4 OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
        "The UDP over IPv4 transport domain.  The corresponding
         transport address is of type TransportAddressIPv4 for
         global IPv4 addresses."
    ::= { transportDomains 1 }
```

```
   transportDomainUdpIpv6 OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The UDP over IPv6 transport domain.  The corresponding
            transport address is of type TransportAddressIPv6 for
            global IPv6 addresses."
       ::= { transportDomains 2 }

   transportDomainUdpIpv4z OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The UDP over IPv4 transport domain.  The corresponding
            transport address is of type TransportAddressIPv4z for
            non-global IPv4 addresses."
       ::= { transportDomains 3 }

   transportDomainUdpIpv6z OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The UDP over IPv6 transport domain.  The corresponding
            transport address is of type TransportAddressIPv6z for
            non-global IPv6 addresses."
       ::= { transportDomains 4 }

   transportDomainTcpIpv4 OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The TCP over IPv4 transport domain.  The corresponding
            transport address is of type TransportAddressIPv4 for
            global IPv4 addresses."
       ::= { transportDomains 5 }

   transportDomainTcpIpv6 OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The TCP over IPv6 transport domain.  The corresponding
            transport address is of type TransportAddressIPv6 for
            global IPv6 addresses."
       ::= { transportDomains 6 }

   transportDomainTcpIpv4z OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The TCP over IPv4 transport domain.  The corresponding
            transport address is of type TransportAddressIPv4z for
            non-global IPv4 addresses."
       ::= { transportDomains 7 }
```

```
   transportDomainTcpIpv6z OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The TCP over IPv6 transport domain.  The corresponding
            transport address is of type TransportAddressIPv6z for
            non-global IPv6 addresses."
       ::= { transportDomains 8 }

   transportDomainLocal OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The Posix Local IPC transport domain. The corresponding
            transport address is of type TransportAddressLocal.

            The Posix Local IPC transport domain incorporates the
            well known UNIX domain sockets."
       ::= { transportDomains 9 }

   transportDomainClns OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The CLNS transport domain.  The corresponding
            transport address is of type TransportAddressOSI."
       ::= { transportDomains 10 }

   transportDomainCons OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The CONS transport domain.  The corresponding
            transport address is of type TransportAddressOSI."
       ::= { transportDomains 11 }

   transportDomainDdp OBJECT-IDENTITY
       STATUS      current
       DESCRIPTION
           "The DDP transport domain.  The corresponding
            transport address is of type TransportAddressNBP."
       ::= { transportDomains 12 }

   transportDomainIpx OBJECT-IDENTITY
       STATUS     current
       DESCRIPTION
           "The IPX transport domain.  The corresponding
            transport address is of type TransportAddressIPX."
       ::= { transportDomains 13 }


   TransportDomain ::= TEXTUAL-CONVENTION
```

```
        STATUS      current
        DESCRIPTION
            "A value that represents a transport domain.

             Some possible values, such as transportDomainUdpIpv4, are
             defined in this module.  Other possible values are defined
             in other MIB modules."
        SYNTAX      OBJECT IDENTIFIER

    --
    -- The enumerated values of the textual convention below SHOULD
    -- be identical to the last sub-identifier of the OID registered
    -- for the same domain.
    --

    TransportAddressType ::= TEXTUAL-CONVENTION
        STATUS      current
        DESCRIPTION
            "A value that represents a transport domain. This is the
          enumerated version of the transport domain registrations
          in this MIB module. The enumerated values have the
          following meaning:

             unknown(0)     unknown transport address type
             udpIpv4(1)     transportDomainUdpIpv4
             udpIpv6(2)     transportDomainUdpIpv6
             udpIpv4z(3)    transportDomainUdpIpv4z
             udpIpv6z(4)    transportDomainUdpIpv6z
             tcpIpv4(5)     transportDomainTcpIpv4
             tcpIpv6(6)     transportDomainTcpIpv6
             tcpIpv4z(7)    transportDomainTcpIpv4z
             tcpIpv6z(8)    transportDomainTcpIpv6z
             local(9)       transportDomainLocal
             clns(10)       transportDomainClns
             cons(11)       transportDomainCons
             ddp(12)        transportDomainDdp
             ipx(13)        transportDomainIpx

             This textual convention can be used to represent transport
             domains in situations where a syntax of TransportDomain is
             unwieldy (for example, when used as an index)."
        SYNTAX      INTEGER {
                        unknown(0),
                        udpIpv4(1),
                        udpIpv6(2),
                        udpIpv4z(3),
                        udpIpv6z(4),
                        tcpIpv4(5),
```

```
                    tcpIpv6(6),
                    tcpIpv4z(7),
                    tcpIpv6z(8),
                    local(9),
                    clns(10),
                    cons(11),
                    ddp(12),
                    ipx(13)
                }

  TransportAddress ::= TEXTUAL-CONVENTION
      STATUS      current
      DESCRIPTION
          "Denotes a generic transport address.

           A TransportAddress value is always interpreted within the
           context of a TransportAddressType or TransportDomain value.
           The TransportAddressType or TransportDomain object which
           defines the format of the TransportAddress value MUST be
           registered before the object(s) which use the
           TransportAddress textual convention.

           The value of a TransportAddress object must always be
           consistent with the value of the associated
           TransportAddressType or TransportDomain object. Attempts
           to set a TransportAddress object to a value which is
           inconsistent with the associated TransportAddressType or
           TransportDomain must fail with an inconsistentValue error.

           When this textual convention is used as a syntax of an
           index object, there may be issues with the limit of 128
           sub-identifiers specified in SMIv2, STD 58. In this case,
           the OBJECT-TYPE declaration MUST include a 'SIZE' clause
           to limit the number of potential instance sub-identifiers."
      SYNTAX      OCTET STRING (SIZE (0..255))

  TransportAddressIPv4 ::= TEXTUAL-CONVENTION
      DISPLAY-HINT "1d.1d.1d.1d:2d"
      STATUS      current
      DESCRIPTION
          "Represents a TCP-over-IPv4 or a UDP-over-IPv4
           transport address:

            octets       contents                  encoding
             1-4         IPv4 address              network-byte order
             5-6         TCP or UDP port           network-byte order"
      SYNTAX      OCTET STRING (SIZE (6))
```

```
   TransportAddressIPv6 ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "0a[2x:2x:2x:2x:2x:2x:2x:2x]0a:2d"
       STATUS      current
       DESCRIPTION
           "Represents a TCP-over-IPv6 or a UDP-over-IPv6
            transport address for global IPv6 addresses:

             octets       contents                  encoding
              1-16        IPv6 address              network-byte order
             17-18        TCP or UDP port           network-byte order

             This textual convention must not be used for non-global
             scoped IPv6 addresses."
       REFERENCE
           "IP Version 6 Addressing Architecture (RFC 2373)"
       SYNTAX      OCTET STRING (SIZE (18))

   TransportAddressIPv4z ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "1d.1d.1d.1d%4d:2d"
       STATUS      current
       DESCRIPTION
           "Represents a TCP-over-IPv4 or a UDP-over-IPv4
            transport address for scoped IPv6 addresses:

             octets       contents                  encoding
              1-4         IPv4 address              network-byte order
              5-8         zone index                network-byte order
              9-10        TCP or UDP port           network-byte order

            This textual convention must not be used for global IPv4
            addresses."
       SYNTAX      OCTET STRING (SIZE (10))

   TransportAddressIPv6z ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "0a[2x:2x:2x:2x:2x:2x:2x:2x%4d]0a:2d"
       STATUS      current
       DESCRIPTION
           "Represents a TCP-over-IPv6 or a UDP-over-IPv6
            transport address for scoped IPv6 addresses:

             octets       contents                  encoding
              1-16        IPv6 address              network-byte order
             17-20        scope identifier          network-byte order
             21-22        TCP or UDP port           network-byte order

            This textual convention must not be used for global IPv6
            addresses."
       REFERENCE
```

```
          "IP Version 6 Addressing Architecture (RFC 2373)"
      SYNTAX      OCTET STRING (SIZE (22))

   TransportAddressLocal ::= TEXTUAL-CONVENTION
      DISPLAY-HINT "1a"
      STATUS      current
      DESCRIPTION
          "Represents a POSIX Local IPC transport address:

           octets       contents                        encoding
            all         POSIX Local IPC address    string

           The Posix Local IPC transport domain subsumes UNIX domain
           sockets."
      REFERENCE
          "Protocol Independent Interfaces (IEEE POSIX 1003.1g)"
      SYNTAX      OCTET STRING (SIZE (1..255))

   TransportAddressOSI ::= TEXTUAL-CONVENTION
      DISPLAY-HINT "*1x:/1x:"
      STATUS      current
      DESCRIPTION
          "Represents an OSI transport-address:

           octets       contents           encoding
             1          length of NSAP     'n' as an unsigned-integer
                                           (either 0 or from 3 to 20)
           2..(n+1)     NSAP               concrete binary representation
           (n+2)..m     TSEL               string of (up to 64) octets"
      SYNTAX      OCTET STRING (SIZE (1 | 4..85))

   TransportAddressNBP ::= TEXTUAL-CONVENTION
      STATUS      current
      DESCRIPTION
          "Represents an NBP name:

               octets        contents          encoding
                  1          length of object  'n' as an unsigned integer
               2..(n+1)      object            string of (up to 32) octets
                 n+2         length of type    'p' as an unsigned integer
            (n+3)..(n+2+p)   type              string of (up to 32) octets
                n+3+p        length of zone    'q' as an unsigned integer
           (n+4+p)..(n+3+p+q) zone             string of (up to 32) octets

           For comparison purposes, strings are case-insensitive. All
           strings may contain any octet other than 255 (hex ff)."
      SYNTAX      OCTET STRING (SIZE (3..99))
```

```
TransportAddressIPX ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "4x.1x:1x:1x:1x:1x:1x.2d"
    STATUS      current
    DESCRIPTION
        "Represents an IPX address:

          octets       contents           encoding
           1-4         network-number     network-byte order
           5-10        physical-address   network-byte order
          11-12        socket-number      network-byte order"
    SYNTAX      OCTET STRING (SIZE (12))

END
```

5. Examples

This section shows some examples how transport addresses are encoded and rendered using some of the initial transport domain and address definitions.

```
Description:        Unspecified IPv4 address on port 80.
Encoding (hex):     000000000050
Display:            0.0.0.0:80

Description:        Global IPv4 address on port 80.
Encoding (hex):     86A922010050
Display:            134.169.34.1:80

Description:        Unspecified IPv6 address on port 80.
Encoding (hex):     00000000000000000000000000000050
Display:            [0:0:0:0:0:0:0:0]:80

Description:        Global IPv6 address on port 80.
Encoding (hex):     10800000000000000080800200C417A0050
Display:            [1080:0:0:0:8:800:200C:417A]:80

Description:        Link-local IPv6 address with zone-index 42 on port 80.
Encoding (hex):     FE800000000000000010000000002000000002A0050
Display:            [FE80:0:0:0:1:0:0:200%42]:80

Description:        Posix Local IPC address (UNIX domain).
Encoding (hex):     2F7661722F6167656E74782F6D6173746572
Display:            /var/agentx/master
```

6. Security Considerations

The MIB module contained in this memo does not define any management objects. Instead, it defines a set of textual conventions which may be used by other MIB modules to define management objects.

Meaningful security considerations can only be written for MIB modules that define concrete management objects. This document has therefore no impact on the security of the Internet.

7. Acknowledgments

This document was produced by the Operations and Management Area "IPv6MIB" design team. Some of the definitions in this module are taken from RFC 1906 [11]. The authors would like to thank Mark Ellison, Brian Haberman, Erik Nordmark, Bill Strahm and Dave Thaler for their comments and suggestions.

8. Intellectual Property Notice

References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

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

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

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

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

[6] 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.

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

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

[9] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "A Simple Network Management Protocol (SNMP)", STD 15, RFC 1157, May 1990.

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

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

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

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

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

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

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

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

[18] Daniele, M., Haberman, B., Routhier, S. and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", ID draft-ietf-ops-rfc2851-update-03.txt, September 2001.

[19] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

[20] IEEE, "Protocol Independent Interfaces", IEEE Std 1003.1g, DRAFT 6.6, March 1997.

[21] Deering, S., Haberman, B., Jinmei, T., Nordmark, E., Onoe, A. and B. Zill, "IPv6 Scoped Address Architecture", draft-ietf-ipngwg-scoping-arch-02.txt, September 2001.

Authors' Addresses

Mike Daniele
Consultant
19 Pinewood Rd
Hudson, NH 03051
USA

Phone: +1 603 883-6365
EMail: mwdaniele@adelphia.net

Juergen Schoenwaelder
TU Braunschweig
Bueltenweg 74/75
38106 Braunschweig
Germany

Phone: +49 531 391-3289
EMail: schoenw@ibr.cs.tu-bs.de

Appendix A. Open Issues

1. Provide suitable transport domain and address format definitions for DNS names, e.g. www.tu-bs.de:80?

2. This version adopts a URL format wherever possible, e.g. 10.1.2.3:80 instead of 10.1.2.3/80 for IPv4 and [00:00:00:00:0A:01:02:03]:80 instead of 00:00:00:00:0A:01:02:03/80 for IPv6 (RFC 2732). Is this useful? Are the DISPLAY-HINTs to achieve the desired output format acceptable?

3. Need to find experts to review the TC definitions for protocols we are not familiar with (TransportAddressOSI, TransportAddressNBP, TransportAddressIPX). Remove the TCs if no expert can be found.

4. Add references and REFERENCE clauses for the various address formats? Probably copying stuff from RFC 1906? Are the references in RFC 1906 still valid?

5. Shall we add more explicit guidelines and examples for the usage of the TransportAddressType TC, similar to what is in the INET-ADDRESS-MIB document?

6. Support for SCTP? How does it work with SCTP failover?

7. Should we give guidance when to use the RFC 1906 definitions and when to use the definitions provided by this memo?

8. Any ideas for a better descriptor prefix to be used throughout this MIB module?

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.