1. Status of this Memo

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress."

To learn the current status of any Internet-Draft, please check the "id-abstracts.txt" listing contained in the internet-drafts Shadow Directories on:

- ftp.is.co.za (Africa)
- nic.nordu.net (Europe)
- ds.internic.net (US East Coast)
- ftp.isi.edu (US West Coast)
- munnari.oz.au (Pacific Rim)
2. Introduction

This memo is informational. It specifies an approach to add floating-point types to versions 1 and 2 of the SNMP SMI[1][2][3][4][5][6], and versions 1 and 2 of the SNMP protocol[7][8][9] without changes. Thus, this addition requires no modifications to existing SNMP MIB compilers, and no changes to existing SNMP protocol engines used in SNMP agents and SNMP management applications.

This memo does not specify a standard for the Internet community.

3. Background

The SNMP protocol and SMI is based on elements from ASN.1[10] and BER[11]. The SMI allows use of only a few ASN.1 base types plus a few SNMP application specific types. Support for floating-point types is not currently found in SNMP. This is primarily due to two factors. The first was the focus during the original development of SNMP to keep SNMP simple and use it for managing computer network devices using the Internet protocol suite. The second factor was the problems with the support for floating-point types as it is defined in ASN.1.

SNMP has been found to be useful for purposes other than those for which it was originally developed. However, some of the limitations in SNMP have restricted its continuing growth. Lack of support for floating-point types has been a problem in some areas. The first example is in mid-level managers that gather and process management information from many sources. The processing includes computing values for mathematical formulas that require floating-point arithmetic. Mid-level managers typically execute on general purpose computers with built-in support for floating point. Thus, supporting floating-point types is not a burden for them. A second area is using SNMP in equipment that require floating point support as part of its normal operation. Examples include heating-cooling systems of large buildings, water and electrical supply systems for cities, and chemical processing plants.

There is a floating-point type in ASN.1. It is called REAL. It is quite complex and allows many options. Also, it does not trivially map to the internal floating-point supported in contemporary computers. On the other hand, the format of floating-point values specified in "IEEE Standard for Binary Floating-Point," ANSI/IEEE Std 754-1985[12] has become widely used. The two standard IEEE encodings of floating-point values are quite different from that specified for the REAL type in the ASN.1 and the BER specifications. Thus, translation between the IEEE encoding formats (used internally in computers) and the BER encoding format (to transmit values) has no value. However, the translation does have the following costs: execution cost; the cost to develop and test the code to perform the translation; and the cost to educate users and developers about a format that has no other application. Thus, a trivial serialization of the IEEE encodings for floating-point values is
needed. Note that IEEE encodings are used in XDR defined in RFC 1832[13].

4. Floating-Point Types

Four floating-point types are defined in "IEEE Standard for Binary Floating-Point." These types are "single," "extended single," "double," and "extended double." Only the single and double formats, which are called "float" and "double" in this memo, are to be used in SNMP. 

Section 5.1.2 of "The Domestication of the Opaque Type for SNMP"[14] requires that a new base type be identified and a textual convention be defined for each new "wrapped" type. Shown below are the definitions for these types and corresponding textual conventions.

-- A floating-point value encoded according to that specified
-- in "IEEE Standard for Binary Floating-Point,"
-- ANSI/IEEE Std 754-1985 for the "single" type.
-- The first octet in the string contains the "sign bit" and
-- the first 7 of the 8 bits of the biased exponent. (The sign
-- bit is the most significant bit of the octet.) The eighth
-- bit of the biased exponent and the 23 bits of the fraction
-- are contained in second through fourth octets. The
-- floating-point values have identical semantics to those
-- defined in the ANSI/IEEE document.

FloatType ::= [APPLICATION 8] IMPLICIT OCTET STRING (SIZE(4))

-- A floating-point value encoded according to that specified
-- in "IEEE Standard for Binary Floating-Point,"
-- ANSI/IEEE Std 754-1985 for the "double" type.
-- The first octet in the string contains the "sign bit" and
-- the first 7 of the 11 bits of the biased exponent. (The sign
-- bit is the most significant bit of the octet.) The eighth
-- through eleventh bits of the biased exponent and the 52
-- bits of the fraction are contained in second through eighth
-- octets. The floating-point values have identical semantics
-- to those defined in the ANSI/IEEE document.

DoubleType ::= [APPLICATION 9] IMPLICIT OCTET STRING (SIZE(8))

Float TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"A single precision floating-point number. The semantics
and encoding are identical for type 'single' defined in
IEEE Standard for Binary Floating-Point,
ANSI/IEEE Std 754-1985."
The value is restricted to the BER serialization of
the following ASN.1 type:

FLOATTYPE ::= [120] IMPLICIT FloatType
(note: the value 120 is the sum of '30'h and '48'h)
The BER serialization of the length for values of
this type must use the definite length, short
encoding form.

For example, the BER serialization of value 123
of type FLOATTYPE is '9f780442f60000'h. (The tag
is '9f78'h; the length is '04'h; and the value is
'42f60000'h.) The BER serialization of value
'9f780442f60000'h of data type Opaque is
'44079f780442f60000'h. (The tag is '44'h; the length
is '07'h; and the value is '9f780442f60000'h.)

SYNTAX Opaque (SIZE(7))

DOUBLE TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
 "A double precision floating-point number. The semantics
and encoding are identical for type 'double' defined in
IEEE Standard for Binary Floating-Point,

The value is restricted to the BER serialization of
the following ASN.1 type:

DOUBLETYPE ::= [121] IMPLICIT DoubleType
(note: the value 121 is the sum of '30'h and '49'h)
The BER serialization of the length for values of
this type must use the definite length, short
encoding form.

For example, the BER serialization of value 123
of type DOUBLETYPE is '9f7908405ec00000000000'h.
(The tag is '9f79'h; the length is '08'h; and the
value is '08405ec00000000000'h.) The BER serialization
of value '9f7908405ec00000000000'h of data type Opaque
is '440b9f7908405ec0000000000000'h. (The tag is '44'h;
the length is '07'h; and the value is
'9f7908405ec0000000000000'h.)"

SYNTAX Opaque (SIZE(11))

6. References

Information for TCP/IP-based Internets", RFC 1155, 05/10/1990.


