Network Working Group
Request for Comments: 1906
Obsoletes: 1449
Category: Standards Track

SNMPv2 Working Group
J. Case
SNMP Research, Inc.
K. McCloghrie
Cisco Systems, Inc.
M. Rose
Dover Beach Consulting, Inc.
S. Waldbusser
International Network Services

January 1996

Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)

Status of this Memo

Table of Contents

1. Introduction

A management system contains: several (potentially many) nodes, each with a processing entity, termed an agent, which has access to management instrumentation; at least one management station; and, a management protocol, used to convey management information between the agents and management stations. Operations of the protocol are carried out under an administrative framework which defines authentication, authorization, access control, and privacy policies.

Management stations execute management applications which monitor and control managed elements. Managed elements are devices such as hosts, routers, terminal servers, etc., which are monitored and controlled via access to their management information.

The management protocol, version 2 of the Simple Network Management Protocol [1], may be used over a variety of protocol suites. It is the purpose of this document to define how the SNMPv2 maps onto an initial set of transport domains. Other mappings may be defined in the future.

Although several mappings are defined, the mapping onto UDP is the preferred mapping. As such, to provide for the greatest level of interoperability, systems which choose to deploy other mappings should also provide for proxy service to the UDP mapping.

1.1. A Note on Terminology

For the purpose of exposition, the original Internet-standard Network Management Framework, as described in RFCs 1155 (STD 16), 1157 (STD 15), and 1212 (STD 16), is termed the SNMP version 1 framework (SNMPv1). The current framework is termed the SNMP version 2 framework (SNMPv2).

2. Definitions

```
SNMPv2-TM DEFINITIONS ::= BEGIN

IMPORTS
    OBJECT-IDENTITY, snmpDomains, snmpProxys
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION
        FROM SNMPv2-TC;

-- SNMPv2 over UDP over IPv4

snmpUDPDomain  OBJECT-IDENTITY
    STATUS     current
    DESCRIPTION
            "The SNMPv2 over UDP transport domain.  The corresponding
            transport address is of type SnmpUDPAddress."
    ::= { snmpDomains 1 }

SnmpUDPAddress ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "1d.1d.1d.1d/2d"
    STATUS       current
    DESCRIPTION
            "Represents a UDP address:

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


-- SNMPv2 over OSI

snmpCLNSDomain OBJECT-IDENTITY
    STATUS     current
    DESCRIPTION
            "The SNMPv2 over CLNS transport domain.  The corresponding
            transport address is of type SnmpOSIAddress."
    ::= { snmpDomains 2 }

snmpCONSDomain OBJECT-IDENTITY
    STATUS     current
    DESCRIPTION
            "The SNMPv2 over CONS transport domain.  The corresponding
            transport address is of type SnmpOSIAddress."
    ::= { snmpDomains 3 }
```

```
SnmpOSIAddress ::= 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))


-- SNMPv2 over DDP

snmpDDPDomain  OBJECT-IDENTITY
    STATUS     current
    DESCRIPTION
            "The SNMPv2 over DDP transport domain.  The corresponding
            transport address is of type SnmpNBPAddress."
    ::= { snmpDomains 4 }

SnmpNBPAddress ::= 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))


-- SNMPv2 over IPX

snmpIPXDomain  OBJECT-IDENTITY
    STATUS     current
    DESCRIPTION
            "The SNMPv2 over IPX transport domain.  The corresponding
```

```
            transport address is of type SnmpIPXAddress."
    ::= { snmpDomains 5 }

SnmpIPXAddress ::= 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))


-- for proxy to SNMPv1 (RFC 1157)

rfc1157Proxy   OBJECT IDENTIFIER ::= { snmpProxys 1 }

rfc1157Domain  OBJECT-IDENTITY
    STATUS     current
    DESCRIPTION
            "The transport domain for SNMPv1 over UDP.  The
            corresponding transport address is of type SnmpUDPAddress."
    ::= { rfc1157Proxy 1 }

--  ::= { rfc1157Proxy 2 }            this OID is obsolete


END
```

3. SNMPv2 over UDP

This is the preferred transport mapping.

3.1. Serialization

Each instance of a message is serialized (i.e., encoded according to the convention of [1]) onto a single UDP[2] datagram, using the algorithm specified in Section 8.

3.2. Well-known Values

It is suggested that administrators configure their SNMPv2 entities acting in an agent role to listen on UDP port 161. Further, it is suggested that notification sinks be configured to listen on UDP port

162.

When an SNMPv2 entity uses this transport mapping, it must be capable of accepting messages that are at least 484 octets in size. Implementation of larger values is encouraged whenever possible.

4. SNMPv2 over OSI

This is an optional transport mapping.

4.1. Serialization

Each instance of a message is serialized onto a single TSDU [3,4] for the OSI Connectionless-mode Transport Service (CLTS), using the algorithm specified in Section 8.

4.2. Well-known Values

It is suggested that administrators configure their SNMPv2 entities acting in an agent role to listen on transport selector "snmp-l" (which consists of six ASCII characters), when using a CL-mode network service to realize the CLTS. Further, it is suggested that notification sinks be configured to listen on transport selector "snmpt-l" (which consists of seven ASCII characters, six letters and a hyphen) when using a CL-mode network service to realize the CLTS. Similarly, when using a CO-mode network service to realize the CLTS, the suggested transport selectors are "snmp-o" and "snmpt-o", for agent and notification sink, respectively.

When an SNMPv2 entity uses this transport mapping, it must be capable of accepting messages that are at least 484 octets in size. Implementation of larger values is encouraged whenever possible.

5. SNMPv2 over DDP

This is an optional transport mapping.

5.1. Serialization

Each instance of a message is serialized onto a single DDP datagram [5], using the algorithm specified in Section 8.

5.2. Well-known Values

SNMPv2 messages are sent using DDP protocol type 8. SNMPv2 entities acting in an agent role listens on DDP socket number 8, whilst notification sinks listen on DDP socket number 9.

Administrators must configure their SNMPv2 entities acting in an agent role to use NBP type "SNMP Agent" (which consists of ten ASCII characters), whilst notification sinks must be configured to use NBP type "SNMP Trap Handler" (which consists of seventeen ASCII characters).

The NBP name for agents and notification sinks should be stable - NBP names should not change any more often than the IP address of a typical TCP/IP node. It is suggested that the NBP name be stored in some form of stable storage.

When an SNMPv2 entity uses this transport mapping, it must be capable of accepting messages that are at least 484 octets in size. Implementation of larger values is encouraged whenever possible.

5.3. Discussion of AppleTalk Addressing

The AppleTalk protocol suite has certain features not manifest in the TCP/IP suite. AppleTalk's naming strategy and the dynamic nature of address assignment can cause problems for SNMPv2 entities that wish to manage AppleTalk networks. TCP/IP nodes have an associated IP address which distinguishes each from the other. In contrast, AppleTalk nodes generally have no such characteristic. The network-level address, while often relatively stable, can change at every reboot (or more frequently).

Thus, when SNMPv2 is mapped over DDP, nodes are identified by a "name", rather than by an "address". Hence, all AppleTalk nodes that implement this mapping are required to respond to NBP lookups and confirms (e.g., implement the NBP protocol stub), which guarantees that a mapping from NBP name to DDP address will be possible.

In determining the SNMP identity to register for an SNMPv2 entity, it is suggested that the SNMP identity be a name which is associated with other network services offered by the machine.

NBP lookups, which are used to map NBP names into DDP addresses, can cause large amounts of network traffic as well as consume CPU resources. It is also the case that the ability to perform an NBP lookup is sensitive to certain network disruptions (such as zone table inconsistencies) which would not prevent direct AppleTalk communications between two SNMPv2 entities.

Thus, it is recommended that NBP lookups be used infrequently, primarily to create a cache of name-to-address mappings. These cached mappings should then be used for any further SNMP traffic. It is recommended that SNMPv2 entities acting in a manager role should maintain this cache between reboots. This caching can help minimize

network traffic, reduce CPU load on the network, and allow for (some amount of) network trouble shooting when the basic name-to-address translation mechanism is broken.

5.3.1. How to Acquire NBP names

An SNMPv2 entity acting in a manager role may have a pre-configured list of names of "known" SNMPv2 entities acting in an agent role. Similarly, an SNMPv2 entity acting in a manager role might interact with an operator. Finally, an SNMPv2 entity acting in a manager role might communicate with all SNMPv2 entities acting in an agent role in a set of zones or networks.

5.3.2. When to Turn NBP names into DDP addresses

When an SNMPv2 entity uses a cache entry to address an SNMP packet, it should attempt to confirm the validity mapping, if the mapping hasn't been confirmed within the last T1 seconds. This cache entry lifetime, T1, has a minimum, default value of 60 seconds, and should be configurable.

An SNMPv2 entity acting in a manager role may decide to prime its cache of names prior to actually communicating with another SNMPv2 entity. In general, it is expected that such an entity may want to keep certain mappings "more current" than other mappings, e.g., those nodes which represent the network infrastructure (e.g., routers) may be deemed "more important".

Note that an SNMPv2 entity acting in a manager role should not prime its entire cache upon initialization - rather, it should attempt resolutions over an extended period of time (perhaps in some pre-determined or configured priority order). Each of these resolutions might, in fact, be a wildcard lookup in a given zone.

An SNMPv2 entity acting in an agent role must never prime its cache. Such an entity should do NBP lookups (or confirms) only when it needs to send an SNMP trap. When generating a response, such an entity does not need to confirm a cache entry.

5.3.3. How to Turn NBP names into DDP addresses

If the only piece of information available is the NBP name, then an NBP lookup should be performed to turn that name into a DDP address. However, if there is a piece of stale information, it can be used as a hint to perform an NBP confirm (which sends a unicast to the network address which is presumed to be the target of the name lookup) to see if the stale information is, in fact, still valid.

An NBP name to DDP address mapping can also be confirmed implicitly using only SNMP transactions. For example, an SNMPv2 entity acting in a manager role issuing a retrieval operation could also retrieve the relevant objects from the NBP group [6] for the SNMPv2 entity acting in an agent role. This information can then be correlated with the source DDP address of the response.

5.3.4. What if NBP is broken

Under some circumstances, there may be connectivity between two SNMPv2 entities, but the NBP mapping machinery may be broken, e.g.,

- the NBP FwdReq (forward NBP lookup onto local attached network) mechanism might be broken at a router on the other entity's network; or,

- the NBP BrRq (NBP broadcast request) mechanism might be broken at a router on the entity's own network; or,

- NBP might be broken on the other entity's node.

An SNMPv2 entity acting in a manager role which is dedicated to AppleTalk management might choose to alleviate some of these failures by directly implementing the router portion of NBP. For example, such an entity might already know all the zones on the AppleTalk internet and the networks on which each zone appears. Given an NBP lookup which fails, the entity could send an NBP FwdReq to the network in which the agent was last located. If that failed, the station could then send an NBP LkUp (NBP lookup packet) as a directed (DDP) multicast to each network number on that network. Of the above (single) failures, this combined approach will solve the case where either the local router's BrRq-to-FwdReq mechanism is broken or the remote router's FwdReq-to-LkUp mechanism is broken.

6. SNMPv2 over IPX

This is an optional transport mapping.

6.1. Serialization

Each instance of a message is serialized onto a single IPX datagram [7], using the algorithm specified in Section 8.

6.2. Well-known Values

SNMPv2 messages are sent using IPX packet type 4 (i.e., Packet Exchange Protocol).

It is suggested that administrators configure their SNMPv2 entities acting in an agent role to listen on IPX socket 36879 (900f hexadecimal). Further, it is suggested that notification sinks be configured to listen on IPX socket 36880 (9010 hexadecimal)

When an SNMPv2 entity uses this transport mapping, it must be capable of accepting messages that are at least 546 octets in size. Implementation of larger values is encouraged whenever possible.

7. Proxy to SNMPv1

In order to provide proxy to SNMPv1 [8], it may be useful to define a transport domain, rfc1157Domain, which indicates the transport mapping for SNMP messages as defined in RFC 1157. Section 3.1 of [9] specifies the behavior of the proxy agent.

8. Serialization using the Basic Encoding Rules

When the Basic Encoding Rules [10] are used for serialization:

(1) When encoding the length field, only the definite form is used; use of the indefinite form encoding is prohibited. Note that when using the definite-long form, it is permissible to use more than the minimum number of length octets necessary to encode the length field.

(2) When encoding the value field, the primitive form shall be used for all simple types, i.e., INTEGER, OCTET STRING, and OBJECT IDENTIFIER (either IMPLICIT or explicit). The constructed form of encoding shall be used only for structured types, i.e., a SEQUENCE or an IMPLICIT SEQUENCE.

(3) When encoding an object whose syntax is described using the BITS construct, the value is encoded as an OCTET STRING, in which all the named bits in (the definition of) the bitstring, commencing with the first bit and proceeding to the last bit, are placed in bits 8 to 1 of the first octet, followed by bits 8 to 1 of each subsequent octet in turn, followed by as many bits as are needed of the final subsequent octet, commencing with bit 8. Remaining bits, if any, of the final octet are set to zero on generation and ignored on receipt.

These restrictions apply to all aspects of ASN.1 encoding, including the message wrappers, protocol data units, and the data objects they contain.

8.1. Usage Example

As an example of applying the Basic Encoding Rules, suppose one wanted to encode an instance of the GetBulkRequest-PDU [1]:

```
  [5] IMPLICIT SEQUENCE {
          request-id      1414684022,
          non-repeaters   1,
          max-repetitions 2,
          variable-bindings {
              { name sysUpTime,
                value { unspecified NULL } },
              { name ipNetToMediaPhysAddress,
                value { unspecified NULL } },
              { name ipNetToMediaType,
                value { unspecified NULL } }
          }
      }
```

Applying the BER, this would be encoded (in hexadecimal) as:

```
[5] IMPLICIT SEQUENCE          a5 82 00 39
    INTEGER                    02 04 52 54 5d 76
    INTEGER                    02 01 01
    INTEGER                    02 01 02
    SEQUENCE                   30 2b
        SEQUENCE               30 0b
            OBJECT IDENTIFIER  06 07 2b 06 01 02 01 01 03
            NULL               05 00
        SEQUENCE               30 0d
            OBJECT IDENTIFIER  06 09 2b 06 01 02 01 04 16 01 02
            NULL               05 00
        SEQUENCE               30 0d
            OBJECT IDENTIFIER  06 09 2b 06 01 02 01 04 16 01 04
            NULL               05 00
```

Note that the initial SEQUENCE is not encoded using the minimum number of length octets. (The first octet of the length, 82, indicates that the length of the content is encoded in the next two octets.)

9. Security Considerations

Security issues are not discussed in this memo.

10. Editor's Address

Keith McCloghrie
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
US

Phone: +1 408 526 5260
EMail: kzm@cisco.com

11. Acknowledgements

This document is the result of significant work by the four major contributors:

Jeffrey D. Case (SNMP Research, case@snmp.com)
Keith McCloghrie (Cisco Systems, kzm@cisco.com)
Marshall T. Rose (Dover Beach Consulting, mrose@dbc.mtview.ca.us)
Steven Waldbusser (International Network Services, stevew@uni.ins.com)

In addition, the contributions of the SNMPv2 Working Group are acknowledged. In particular, a special thanks is extended for the contributions of:

Alexander I. Alten (Novell)
Dave Arneson (Cabletron)
Uri Blumenthal (IBM)
Doug Book (Chipcom)
Kim Curran (Bell-Northern Research)
Jim Galvin (Trusted Information Systems)
Maria Greene (Ascom Timeplex)
Iain Hanson (Digital)
Dave Harrington (Cabletron)
Nguyen Hien (IBM)
Jeff Johnson (Cisco Systems)
Michael Kornegay (Object Quest)
Deirdre Kostick (AT&T Bell Labs)
David Levi (SNMP Research)
Daniel Mahoney (Cabletron)
Bob Natale (ACE*COMM)
Brian O'Keefe (Hewlett Packard)
Andrew Pearson (SNMP Research)
Dave Perkins (Peer Networks)
Randy Presuhn (Peer Networks)
Aleksey Romanov (Quality Quorum)
Shawn Routhier (Epilogue)
Jon Saperia (BGS Systems)

Bob Stewart (Cisco Systems, bstewart@cisco.com), chair
Kaj Tesink (Bellcore)
Glenn Waters (Bell-Northern Research)
Bert Wijnen (IBM)

12. References

[1] SNMPv2 Working Group, 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.

[2] Postel, J., "User Datagram Protocol", STD 6, RFC 768, USC/Information Sciences Institute, August 1980.

[3] Information processing systems - Open Systems Interconnection - Transport Service Definition, International Organization for Standardization. International Standard 8072, (June, 1986).

[4] Information processing systems - Open Systems Interconnection - Transport Service Definition - Addendum 1: Connectionless-mode Transmission, International Organization for Standardization. International Standard 8072/AD 1, (December, 1986).

[5] G. Sidhu, R. Andrews, A. Oppenheimer, Inside AppleTalk (second edition). Addison-Wesley, 1990.

[6] Waldbusser, S., "AppleTalk Management Information Base", RFC 1243, Carnegie Mellon University, July 1991.

[7] Network System Technical Interface Overview. Novell, Inc, (June, 1989).

[8] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, SNMP Research, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[9] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework", RFC 1908, January 1996.

[10] Information processing systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), International Organization for Standardization. International Standard 8825, December 1987.