Network Working Group
Request for Comments: 1449

J. Case
SNMP Research, Inc.
K. McCloghrie
Hughes LAN Systems
M. Rose
Dover Beach Consulting, Inc.
S. Waldbusser
Carnegie Mellon University
April 1993

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

Status of this Memo

This RFC specifes an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Introduction

A network 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 both authentication and authorization policies.

Network management stations execute management applications which monitor and control network elements. Network elements are devices such as hosts, routers, terminal servers, etc., which are monitored and controlled through 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, 1157, and 1212, 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
    snmpDomains, snmpProxys
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION
        FROM SNMPv2-TC;

-- SNMPv2 over UDP

snmpUDPDomain  OBJECT IDENTIFIER ::= { snmpDomains 1 }
-- for a SnmpUDPAddress of length 6:
--
-- octets   contents        encoding
--  1-4     IP-address      network-byte order
--  5-6     UDP-port        network-byte order
--
SnmpUDPAddress ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "1d.1d.1d.1d/2d"
    STATUS       current
    DESCRIPTION
            "Represents a UDP address."
    SYNTAX       OCTET STRING (SIZE (6))
```

```
-- SNMPv2 over OSI

snmpCLNSDomain OBJECT IDENTIFIER ::= { snmpDomains 2 }
snmpCONSDomain OBJECT IDENTIFIER ::= { snmpDomains 3 }
-- for a SnmpOSIAddress of length m:
--
-- 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
--
SnmpOSIAddress ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "*1x:/1x:"
    STATUS       current
    DESCRIPTION
            "Represents an OSI transport-address."
    SYNTAX       OCTET STRING (SIZE (1 | 4..85))
```

```
-- SNMPv2 over DDP

snmpDDPDomain  OBJECT IDENTIFIER ::= { snmpDomains 4 }
-- for a SnmpNBPAddress of length m:
--
--    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)..m     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)
--
SnmpNBPAddress ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
            "Represents an NBP name."
    SYNTAX       OCTET STRING (SIZE (3..99))


-- SNMPv2 over IPX

snmpIPXDomain  OBJECT IDENTIFIER ::= { snmpDomains 5 }
-- for a SnmpIPXAddress of length 12:
--
-- octets   contents            encoding
--  1-4     network-number      network-byte order
--  5-10    physical-address    network-byte order
-- 11-12    socket-number       network-byte order
--
SnmpIPXAddress ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "4x.1x:1x:1x:1x:1x:1x.2d"
    STATUS       current
    DESCRIPTION
            "Represents an IPX address."
    SYNTAX       OCTET STRING (SIZE (12))
```

```
-- for proxy to community-based SNMPv1 (RFC 1157)

rfc1157Proxy   OBJECT IDENTIFIER ::= { snmpProxys 1 }

-- uses SnmpUDPAddress
rfc1157Domain  OBJECT IDENTIFIER ::= { rfc1157Proxy 1 }

-- the community-based noAuth
rfc1157noAuth  OBJECT IDENTIFIER ::= { rfc1157Proxy 2 }


END
```

3. SNMPv2 over UDP

This is the preferred transport mapping.

3.1. Serialization

Each instance of a message is serialized onto a single UDP[2] datagram, using the algorithm specified in Section 8.

3.2. Well-known Values

Although the partyTable gives transport addressing information for an SNMPv2 party, 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.

The partyTable also lists the maximum message size which a SNMPv2 party is willing to accept. This value must be at least 484 octets. 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

Although the partyTable gives transport addressing information for an SNMPv2 party, 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) 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.

The partyTable also lists the maximum message size which a SNMPv2 party is willing to accept. This value must be at least 484 octets. 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.

Although the partyTable gives transport addressing information for an SNMPv2 party, 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.

The partyTable also lists the maximum message size which a SNMPv2 party is willing to accept. This value must be at least 484 octets. 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 Packet).

Although the partyTable gives transport addressing information for an SNMPv2 party, 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)

The partyTable also lists the maximum message size which a SNMPv2 party is willing to accept. This value must be at least 546 octets. Implementation of larger values is encouraged whenever possible.

7. Proxy to SNMPv1

In order to provide proxy to community-based SNMP [8], some definitions are necessary for both transport domains and authentication protocols.

7.1. Transport Domain: rfc1157Domain

The transport domain, rfc1157Domain, indicates the transport mapping for community-based SNMP messages defined in RFC 1157. When a party's transport domain (partyTDomain) is rfc1157Domain:

(1) the party's transport address (partyTAddress) shall be 6 octets long, the initial 4 octets containing the IP-address in network-byte order, and the last two octets containing the UDP port in network-byte order; and,

(2) the party's authentication protocol (partyAuthProtocol) shall be rfc1157noAuth.

When a proxy relationship identifies a proxy destination party which has rfc1157Domain as its transport domain:

(1) the proxy source party (contextSrcPartyIndex) and proxy context (contextProxyContext) components of the proxy relationship are irrelevant; and,

(2) Section 3.1 of [9] specifies the behavior of the proxy agent.

7.2. Authentication Algorithm: rfc1157noAuth

A party's authentication protocol (partyAuthProtocol) specifies the protocol and mechanism by which the party authenticates the integrity and origin of the SNMPv1 or SNMPv2 PDUs it generates. When a party's authentication protocol is rfc1157noAuth:

(1) the party's public authentication key (partyAuthPublic), clock (partyAuthClock), and lifetime (partyAuthLifetime) are irrelevant; and,

(2) the party's private authentication key (partySecretsAuthPrivate) shall be used as the 1157 community for the proxy destination, and shall be at least one octet in length. (No maximum length is specified.)

Note that when setting the party's private authentication key, the exclusive-OR semantics specified in [10] still apply.

8. Serialization using the Basic Encoding Rules

When the Basic Encoding Rules [11] 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, OBJECT IDENTIFIER, and BIT STRING (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 a BIT STRING is serialized, all named-bits are transferred regardless of their truth-value. Further, if the number of named-bits is not an integral multiple of eight, then the fewest number of additional zero-valued bits are transferred so that an integral multiple of eight bits is transferred.

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. Acknowledgements

The UDP-based mapping is based, in part, on RFC 1157.

The OSI-based mapping is based, in part, on RFC 1283.

The DDP-based mapping is based, in part, on earlier work by Greg Minshall of Novell, Inc., and Mike Ritter of Apple Computer, Inc.

The IPX-based mapping is based, in part, on RFC 1298.

The section on proxy to community-based SNMP is based on earlier work that was based in part on a suggestion by Jonathan Biggar of Netlabs, Inc.

Finally, the comments of the SNMP version 2 working group are gratefully acknowledged:

Beth Adams, Network Management Forum
Steve Alexander, INTERACTIVE Systems Corporation
David Arneson, Cabletron Systems
Toshiya Asaba
Fred Baker, ACC
Jim Barnes, Xylogics, Inc.
Brian Bataille
Andy Bierman, SynOptics Communications, Inc.
Uri Blumenthal, IBM Corporation
Fred Bohle, Interlink
Jack Brown
Theodore Brunner, Bellcore
Stephen F. Bush, GE Information Services
Jeffrey D. Case, University of Tennessee, Knoxville
John Chang, IBM Corporation
Szusin Chen, Sun Microsystems
Robert Ching
Chris Chiotasso, Ungermann-Bass
Bobby A. Clay, NASA/Boeing
John Cooke, Chipcom
Tracy Cox, Bellcore
Juan Cruz, Datability, Inc.
David Cullerot, Cabletron Systems
Cathy Cunningham, Microcom
James R. (Chuck) Davin, Bellcore
Michael Davis, Clearpoint

Mike Davison, FiberCom
Cynthia DellaTorre, MITRE
Taso N. Devetzis, Bellcore
Manual Diaz, DAVID Systems, Inc.
Jon Dreyer, Sun Microsystems
David Engel, Optical Data Systems
Mike Erlinger, Lexcel
Roger Fajman, NIH
Daniel Fauvarque, Sun Microsystems
Karen Frisa, CMU
Shari Galitzer, MITRE
Shawn Gallagher, Digital Equipment Corporation
Richard Graveman, Bellcore
Maria Greene, Xyplex, Inc.
Michel Guittet, Apple
Robert Gutierrez, NASA
Bill Hagerty, Cabletron Systems
Gary W. Haney, Martin Marietta Energy Systems
Patrick Hanil, Nokia Telecommunications
Matt Hecht, SNMP Research, Inc.
Edward A. Heiner, Jr., Synernetics Inc.
Susan E. Hicks, Martin Marietta Energy Systems
Geral Holzhauer, Apple
John Hopprich, DAVID Systems, Inc.
Jeff Hughes, Hewlett-Packard
Robin Iddon, Axon Networks, Inc.
David Itusak
Kevin M. Jackson, Concord Communications, Inc.
Ole J. Jacobsen, Interop Company
Ronald Jacoby, Silicon Graphics, Inc.
Satish Joshi, SynOptics Communications, Inc.
Frank Kastenholz, FTP Software
Mark Kepke, Hewlett-Packard
Ken Key, SNMP Research, Inc.
Zbiginew Kielczewski, Eicon
Jongyeoi Kim
Andrew Knutsen, The Santa Cruz Operation
Michael L. Kornegay, VisiSoft
Deirdre C. Kostik, Bellcore
Cheryl Krupczak, Georgia Tech
Mark S. Lewis, Telebit
David Lin
David Lindemulder, AT&T/NCR
Ben Lisowski, Sprint
David Liu, Bell-Northern Research

John Lunny, The Wollongong Group
Robert C. Lushbaugh Martin, Marietta Energy Systems
Michael Luufer, BBN
Carl Madison, Star-Tek, Inc.
Keith McCloghrie, Hughes LAN Systems
Evan McGinnis, 3Com Corporation
Bill McKenzie, IBM Corporation
Donna McMaster, SynOptics Communications, Inc.
John Medicke, IBM Corporation
Doug Miller, Telebit
Dave Minnich, FiberCom
Mohammad Mirhakkak, MITRE
Rohit Mital, Protools
George Mouradian, AT&T Bell Labs
Patrick Mullaney, Cabletron Systems
Dan Myers, 3Com Corporation
Rina Nathaniel, Rad Network Devices Ltd.
Hien V. Nguyen, Sprint
Mo Nikain
Tom Nisbet
William B. Norton, MERIT
Steve Onishi, Wellfleet Communications, Inc.
David T. Perkins, SynOptics Communications, Inc.
Carl Powell, BBN
Ilan Raab, SynOptics Communications, Inc.
Richard Ramons, AT&T
Venkat D. Rangan, Metric Network Systems, Inc.
Louise Reingold, Sprint
Sam Roberts, Farallon Computing, Inc.
Kary Robertson, Concord Communications, Inc.
Dan Romascanu, Lannet Data Communications Ltd.
Marshall T. Rose, Dover Beach Consulting, Inc.
Shawn A. Routhier, Epilogue Technology Corporation
Chris Rozman
Asaf Rubissa, Fibronics
Jon Saperia, Digital Equipment Corporation
Michael Sapich
Mike Scanlon, Interlan
Sam Schaen, MITRE
John Seligson, Ultra Network Technologies
Paul A. Serice, Corporation for Open Systems
Chris Shaw, Banyan Systems
Timon Sloane
Robert Snyder, Cisco Systems
Joo Young Song

Roy Spitier, Sprint
Einar Stefferud, Network Management Associates
John Stephens, Cayman Systems, Inc.
Robert L. Stewart, Xyplex, Inc. (chair)
Kaj Tesink, Bellcore
Dean Throop, Data General
Ahmet Tuncay, France Telecom-CNET
Maurice Turcotte, Racal Datacom
Warren Vik, INTERACTIVE Systems Corporation
Yannis Viniotis
Steven L. Waldbusser, Carnegie Mellon Universitty
Timothy M. Walden, ACC
Alice Wang, Sun Microsystems
James Watt, Newbridge
Luanne Waul, Timeplex
Donald E. Westlake III, Digital Equipment Corporation
Gerry White
Bert Wijnen, IBM Corporation
Peter Wilson, 3Com Corporation
Steven Wong, Digital Equipment Corporation
Randy Worzella, IBM Corporation
Daniel Woycke, MITRE
Honda Wu
Jeff Yarnell, Protools
Chris Young, Cabletron
Kiho Yum, 3Com Corporation

10. References

[1] Case, J., McCloghrie, K., Rose, M., and Waldbusser, S., "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1448, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.

[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., Davin, J., "Simple Network Management Protocol", STD 15, RFC 1157, SNMP Research, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[9] Case, J., McCloghrie, K., Rose, M., and Waldbusser, S., "Coexistence between version 1 and version 2 of the Internet-standard Network Management Framework", RFC 1452, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.

[10] McCloghrie, K., and Galvin, J., "Party MIB for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC

1447, Hughes LAN Systems, Trusted Information Systems, April 1993.

[11] 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).

11. Security Considerations

Security issues are not discussed in this memo.

12. Authors' Addresses

Jeffrey D. Case
SNMP Research, Inc.
3001 Kimberlin Heights Rd.
Knoxville, TN 37920-9716
US

Phone: +1 615 573 1434
Email: case@snmp.com

Keith McCloghrie
Hughes LAN Systems
1225 Charleston Road
Mountain View, CA 94043
US

Phone: +1 415 966 7934
Email: kzm@hls.com

Marshall T. Rose
Dover Beach Consulting, Inc.
420 Whisman Court
Mountain View, CA 94043-2186
US

Phone: +1 415 968 1052
Email: mrose@dbc.mtview.ca.us

Steven Waldbusser
Carnegie Mellon University
4910 Forbes Ave
Pittsburgh, PA 15213
US

Phone: +1 412 268 6628
Email: waldbusser@cmu.edu