SNMP and SNMPv2: The Infrastructure for Network Management

William Stallings

ABSTRACT The Simple Network Management Protocol is the most widely used protocol for the management of IP-based networks and internets. The original version, now known as SNMPv1, is widely deployed. SNMPv2 adds functionality to the original version but does not address its security limitations; this relatively recent standard has not achieved much acceptance. An effort is currently underway to develop SNMPv3, which will retain the functional enhancements of SNMPv2 and add powerful privacy and authentication features. This article provides a survey of the three versions of SNMP, including a discussion of the way in which management information is represented and the protocol functionality.

The Simple Network Management Protocol (SNMP), issued in 1988, was designed to provide an easily implemented, low-overhead foundation for multivendor network management of routers, servers, workstations, and other network resources. The SNMP specification:

- Defines a protocol for exchanging information between one or more management systems and a number of agents
- Provides a framework for formatting and storing management information
- Defines a number of general-purpose management information variables, or objects

The original version of SNMP (now known as SNMPv1) rapidly became the most widely used vendor-independent network management scheme. However, as the protocol gained widespread use, its deficiencies became apparent. These include a lack of manager-to-manager communication, the inability to do bulk data transfer, and a lack of security. All of these deficiencies were addressed in SNMPv2, issued as a set of proposed Internet standards in 1993.

SNMPv2 has not received the acceptance its designers anticipated. While the functional enhancements have been welcome, developers found the security facility for SNMPv2 too complex. Accordingly, the SNMPv2 working group was reactivated to provide a "tune-up" of the SNMPv2 documents. The result of this effort has been one minor success and one major failure. The minor success is the tune-up of the functional aspects of SNMPv2. The major failure is in the area of security. The working group was unable to resolve the issue, and two competing approaches emerged. With this tune-up, the functional portion of SNMPv2 progressed from proposed to draft Internet standard status as of 1996. Then, in 1997, work began on SNMPv3, which makes additional minor functional changes and incorporates a new security approach.

This article will provide a survey of SNMPv1 and SNMPv2, and a brief overview of SNMPv3. The article begins with a discussion of basic concepts common to all versions; these concepts define the network management framework that SNMP is designed to support. Then, the operation of SNMPv1 is described. Next, the functional enhancements found in SNMPv2 are discussed. A final section introduces SNMPv3.

BASIC SNMP CONCEPTS

This section examines basic network management concepts that are used as a framework for all three versions of SNMP. We begin with a discussion of the network management architecture, in terms of managed and managing entities, that SNMP is designed to address. Then we look at the protocol architecture used in SNMP. Finally, two important operational concepts, trap-directed polling and proxies, are introduced.

NETWORK MANAGEMENT ARCHITECTURE

The model of network management that is used for SNMP includes the following key elements:

- Management station
- Management agent
- Management information base
- Network management protocol

A *management station* is typically a standalone device, but may be a capability implemented on a shared system. In either case, the management station serves as the interface for the human network manager into the network management system. The management station will have, at minimum:

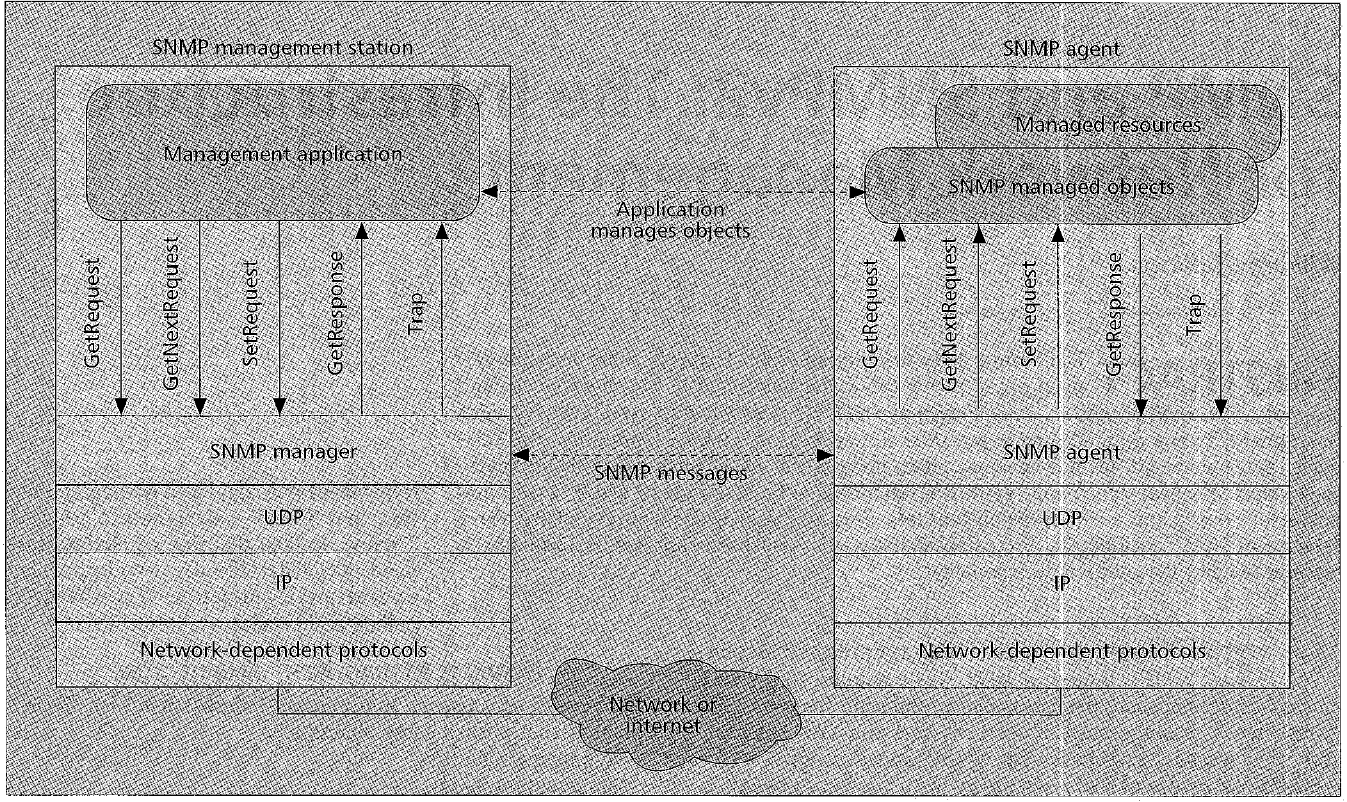
- A set of management applications for data analysis, fault recovery, and so on.
- An interface by which the network manager may monitor and control the network. That is, the interface between the user and the network management applications enables the user to request actions (monitoring and control) which are carried out by the management station by communicating with the managed elements of the network.
- A protocol by which the management station and managed entities exchange control and management information.
- A database of information extracted from the management databases of all the managed entities in the network. That is, the management station maintains at least a summary of the management information maintained at each of the managed elements in the network.

Only the last two elements are the subject of SNMP standardization.

The other active element in the network management system is the *management agent*. Key platforms, such as hosts, bridges, routers, and hubs, may be equipped with SNMP agent software so that they may be managed from a management station. The management agent responds to requests for information from a management station, responds to requests for actions from the management station, and may asynchronously provide the management station with important but unsolicited information.

In order to manage the resources in a network, these resources are represented as objects. Each object is, essentially, a data variable that represents one aspect of the managed system. The collection of objects is referred to as a *management information base* (MIB). The MIB functions as a collection of access points at the agent for the management station; the agent software maintains the MIB. These objects are stan-

0163-6804/98/$10.00 © 1998 IEEE



■ **Figure 1.** *The role of SNMP.*

dardized across systems of a particular class (e.g., bridges all support the same management objects). In addition, proprietary extensions can be made. A management station performs the monitoring function by retrieving the value of MIB objects. A management station can cause an action to take place at an agent or can change the configuration settings of an agent by modifying the value of specific variables.

The management station and agents are linked by a *network management protocol*, which includes the following key capabilities:

- *Get*: enables the management station to retrieve the values of objects at the agent
- *Set*: enables the management station to set the values of objects at the agent
- *Trap*: enables an agent to notify the management station of significant events

There are no specific guidelines in the standards as to the number of management stations or the ratio of management stations to agents. In general, it is prudent to have at least two systems capable of performing the management station function, to provide redundancy in case of failure. The other issue is the practical one of how many agents a single management station can handle. As long as SNMP remains relatively "simple," that number can be quite high, certainly in the hundreds.

Network Management Protocol Architecture

SNMP was designed to be an application-level protocol that is part of the TCP/IP protocol suite. As Fig. 1 illustrates, SNMP typically operates over the user datagram protocol (UDP), although it may also operate over TCP. For a standalone management station, a manager process controls access to the central MIB at the management station and provides an interface to the network manager. The manager process achieves network management by using SNMP, which is implemented on top of UDP, IP, and the relevant network-dependent protocols (e.g., Ethernet, FDDI, X.25).

Each agent must also implement SNMP, UDP, and IP. In addition, there is an agent process that interprets the SNMP messages and controls remote access to the agent's MIB. For an agent device that supports other applications, such as FTP, TCP as well as UDP is required.

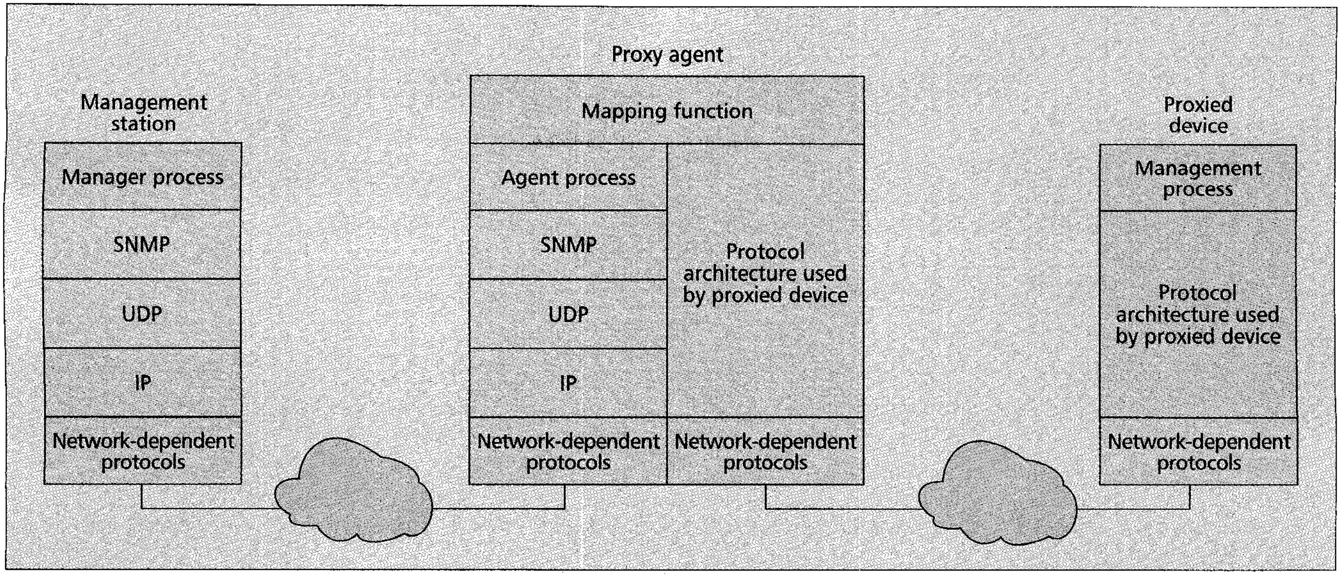
From a management station, three types of SNMP messages are issued on behalf of a management application: GetRequest, GetNextRequest, and SetRequest. The first two are variations of the get function. All three messages are acknowledged by the agent in the form of a GetResponse message, which is passed up to the management application. In addition, an agent may issue a trap message in response to an event that affects the MIB and the underlying managed resources.

SNMP relies on UDP, which is a connectionless protocol, and SNMP is itself connectionless. No ongoing connections are maintained between a management station and its agents. Instead, each exchange is a separate transaction between a management station and an agent.

Trap-Directed Polling

If a management station is responsible for a large number of agents, and if each agent maintains a large number of objects, it becomes impractical for the management station to regularly poll all agents for all of their readable object data. Instead, SNMP and the associated MIB are designed to encourage the manager to use a technique referred to as *trap-directed polling*.

The recommended strategy is this. At initialization time, and perhaps at infrequent intervals, such as once a day, a management station can poll all the agents it knows of for some key information, such as interface characteristics, and perhaps some baseline performance statistics, such as average number of packets sent and received over each interface over a given period of time. Once this baseline is established, the management station refrains from polling. Instead, each agent is responsible for notifying the management station of any unusual event. Examples are if the agent crashes and is


Proxy agent
Management station
Manager process
SNMP
UDP
IP
Network-dependent protocols
Mapping function
Agent process
SNMP
UDP
IP
Network-dependent protocols
Protocol architecture used by proxied device
Network-dependent protocols
Proxied device
Management process
Protocol architecture used by proxied device
Network-dependent protocols

■ **Figure 2.** *Proxy configuration.*

rebooted, the failure of a link, or an overload condition as defined by the packet load crossing some threshold. These events are communicated in SNMP messages known as *traps*.

Once a management station is alerted to an exception condition, it may choose to take some action. At this point, the management station may direct polls to the agent reporting the event and perhaps to some nearby agents in order to diagnose any problem and to gain more specific information about the exception condition. However, because traps are communicated via UDP and are therefore delivered unreliably, a management station may wish to infrequently poll agents.

Trap-directed polling can result in substantial savings of network capacity and agent processing time. In essence, the network is not made to carry management information that the management station does not need, and agents are not made to respond to frequent requests for uninteresting information.

Proxies

The use of SNMP requires that all agents, as well as management stations, must support UDP and IP. This limits direct management to such devices and excludes other devices, such as some bridges and modems, that do not support any part of the TCP/IP protocol suite. Furthermore, there may be numerous small systems (personal computers, workstations, programmable controllers), that do implement TCP/IP to support their applications, but for which it is not desirable to add the additional burden of SNMP, agent logic, and MIB maintenance.

To accommodate devices that do not implement SNMP, the concept of proxy was developed. In this scheme an SNMP agent acts as a proxy for one or more other devices; that is, the SNMP agent acts on behalf of the proxied devices.

Figure 2 indicates the type of protocol architecture that is often involved. The management station sends queries concerning a device to its proxy agent. The proxy agent converts each query into the management protocol that is used by the device. When a reply to a query is received by the agent, it passes that reply back to the management station. Similarly, if an event notification of some sort from the device is transmitted to the proxy, the proxy sends that on to the management station in the form of a trap message.

RFC	Title	Date
1155	Structure and identification of management information for TCP/IP-based internets	May 1990
1157	A Simple Network Management Protocol (SNMP)	May 1990
1212	Concise MIB definitions	March 1991
1213	Managment information base for network management of TCP/IP-based Internets: MIB-II	March 1991

■ **Table 1.** *Key SNMPv1 RFCs.*

SNMPv1

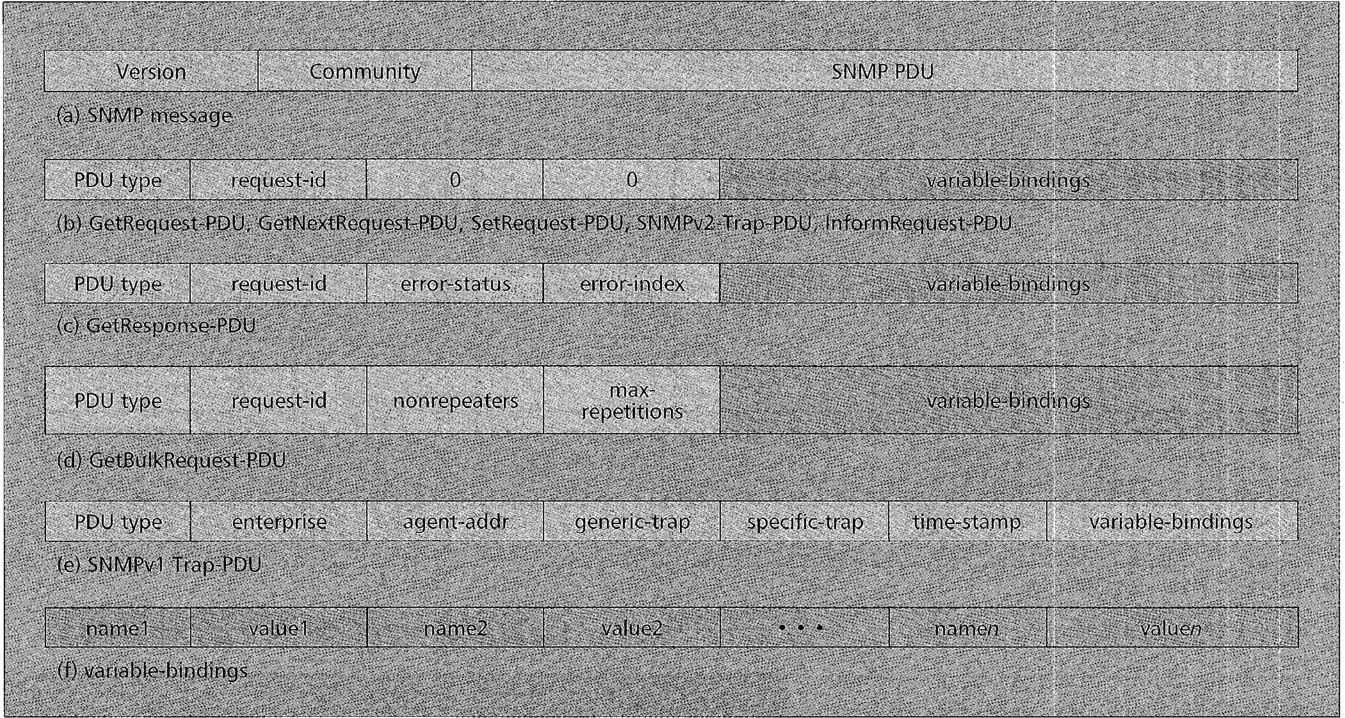
Table 1 lists the key RFCs that define SNMPv1. In this section, we describe the basic formats and operations of the protocol. These formats and operations, with extensions and some modifications, are retained in SNMPv2 and SNMPv3.

With SNMPv1, information is exchanged between a management station and an agent in the form of a message. Each SNMPv1 message includes a version number, indicating the version of SNMP, a community name to be used for this exchange, and one of five types of protocol data units (PDUs). This structure is depicted in Fig. 3, and the constituent fields are defined in Table 2. Note that the GetRequest, GetNextRequest, and SetRequest PDUs have the same format as the GetResponse PDU, with the error-status and error-index fields always set to 0. This convention reduces by one the number of different PDU formats with which the SNMP entity must deal.

The GetRequest and GetNextRequest PDUs are both commands from a manager to retrieve data from an agent. The difference is that the GetRequest lists a specific variable or variables to be retrieved, while the GetNextRequest is used for traversing a tree-structured MIB. In both cases, the values, if available, are returned in a GetResponse PDU. The Set command is a command from a manager to update variables in an agent; in this case the GetResponse PDU provides an acknowledgment. Finally, the Trap PDU is a notification from an agent to a manager.

Transmission of an SNMP Message

In principle, an SNMP entity performs the following actions to transmit one of the five PDU types to another SNMP entity:



■ **Figure 3.** *SNMP formats.*

1 The PDU is constructed.
2 This PDU is then passed to an authentication service, together with the source and destination transport addresses and a community name. The authentication service then performs any required transformations for this exchange, such as encryption or the inclusion of an authentication code, and returns the result. The community name is a value that indicates the context for this authentication procedure.
3 The protocol entity then constructs a message, consisting of a version field, the community name, and the result from step 2.
4 This message is passed to the transport service.

In practice, authentication is not typically invoked.

Receipt of an SNMP Message

In principle, an SNMP entity performs the following actions upon reception of an SNMP message:

- It does a basic syntax check of the message, and discards the message if it fails to parse.
- It verifies the version number, and discards the message if there is a mismatch.
- The protocol entity then passes the user name, the PDU portion of the message, and the source and destination transport addresses (supplied by the transport service that delivered the message) to an authentication service.
  – If authentication fails, the authentication service signals the SNMP protocol entity, which generates a trap and discards the message.
  – If authentication succeeds, the authentication service returns the PDU.
- The protocol entity does a basic syntax check of the PDU and discards the PDU if it fails to parse. Otherwise, using the named community, the appropriate SNMP access policy is selected and the PDU is processed accordingly.

In practice, the authentication service merely serves to verify that the community name authorizes receipt of messages from the source SNMP entity.

Variable Bindings

All SNMP operations involve access to scalar objects. However, it is possible in SNMP to group a number of operations of the same type (get, set, trap) into a single message. Thus, if a management station wants to get the values of all scalar objects in a particular group at a particular agent, it can send a single message requesting all values, and get a single response, listing all values. This technique can greatly reduce the communications burden of network management.

To implement multiple-object exchanges, all of the SNMP PDUs include a variable-bindings field. This field consists of a sequence of references to object instances, together with the value of those objects. Some PDUs are concerned only with the name of the object instance (e.g., get operations). In this case, the value entries in the variable-bindings field are ignored by the receiving protocol entity.

SNMPv2

SNMPv1 has proliferated rapidly because it is what it claims to be: a simple tool for network management. SNMPv1 provides a bare-bones set of functions that is easy to implement, relatively easy to use, and, if used sensibly, imposes minimal overhead on network operations. The popularity of SNMPv1 eventually caught up with it. Now that (human) managers are used to the level of control available with SNMPv1, they see its flaws and want more functionality. Among the most noteworthy areas needing improvement were support for efficient transfer of large blocks of data, decentralized network management strategies, and security. The first two of these are addressed in the SNMPv2 specifications (Table 3).

Data Transfer Enhancements

SNMPv1 can generate considerable traffic as managers communicate with agents. That is because, with SNMPv1, only a limited amount of data can be exchanged in a single transaction, frequently forcing management workstations and agents

Field	Description
version	SNMP version; RFC 1157 is version 1.
community	A pairing of an SNMP agent with some arbitrary set of SNMP application entities. The name of the community functions as a password to authenticate the SNMP message.
request-id	Used to distinguish among outstanding requests by providing each request with a unique ID.
error-status	Used to indicate that an exception occurred while processing a request. Values are: noError (0), tooBig (1), noSuchName (2), badValue (3), readOnly (4), genErr (5)
error-index	When error-status is nonzero, error-index may provide additional information by indicating which variable in a list caused the exception. A variable is an instance of a managed object.
variable-bindings	A list of variable names and corresponding values. In some cases (e.g. GetRequest-PDU), the values are null.
enterprise	Type of object generating trap; based on sysObjectID.
agent-addr	Address of object generating trap.
generic-trap	Generic trap type. Values are: coldStart (0), warmStart (1), linkDown (2), linkUp (3), authenticationFailure (4), egpNeighborLoss (5), enterpriseSpecific (6).
specific-trap	Specific trap code.
time-stamp	Time elapsed between the last (re)initialization of the network entity and the generation of the trap; contains the value of sysUpTime.
non-repeaters	Indicates how many listed variables are to return just one value each.
max-repetitions	Indicates number of values to be returned for each of the remaining variables.

■ **Table 2.** *SNMP message and PDU fields.*

to generate multiple transactions. The result can be a heavy load on the network that can affect response time for end-user applications.

To streamline these exchanges, SNMPv2 adds a new command, the GetBulk command, and introduces an improved version of SNMP's Get command.

The GetBulk command (Fig. 3) targets the one area of information exchange capable of generating the most traffic: retrieval of tables. A table represents a related set of information about a resource (e.g., a router) or activity (e.g., the traffic over a TCP connection). It is organized as a collection of rows of variables, with each row having the same sequence of variables. For example, each router in a configuration maintains a routing table with one row for each destination. The row is indexed by the destination address and includes a field for the next hop to take to get to the destination, and the amount of time since this routing information was last changed. All of the rows have the same format, with one row per destination.

With SNMPv1, it is only possible to retrieve information from such a table one row at a time. If a manager needs to see an entire routing table, for example, then a tedious series of get/response transactions is needed, one for each row.

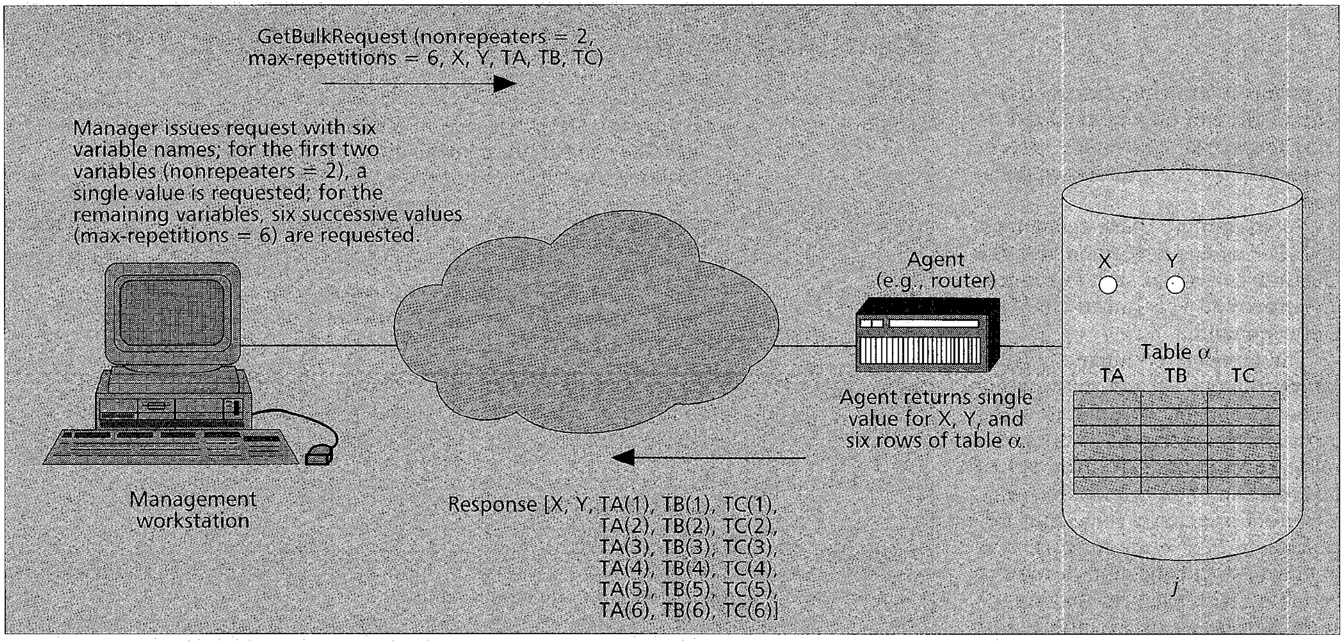
With the GetBulk command, the manager can retrieve the entire table with one transaction and even retrieve additional non-table information in that same transaction. For example, suppose a manager wished to retrieve the entire routing table plus the variable sysUpTime so that it could associate a system time with the retrieved table. The manager would issues a getBulk command that would list the variable sysUpTime, plus the variables that correspond to each of the fields in the table, including destination, next hop, and age. The command also includes two parameters: the nonrepeaters parameter indicates how many of the listed variables are to return just one value; in this case there is only one such variable, sysUpTime, so nonrepeaters is set to 1. The max-repetitions parameter indicates how many rows of the table are to be retrieved. If the manager knows the number, of rows, then max-repetitions is set to that value. Otherwise, the manager makes an educated guess and, if necessary, issues additional GetBulk commands to get additional rows. Figure 4 depicts an example.

Another feature SNMPv2 offers to improve the efficiency of data transfer is the so-called nonatomic Get command. Management stations in both SNMPv1 and SNMPv2 use the Get command to obtain the value of one or more variables. In SNMPv1, if a Get command lists multiple variables, and if the agent is unable to return a value for even one of those variables, the entire command is rejected. If this happens, the manager must reissue the Get command with fewer variables. SNMPv2's nonatomic Get command allows partial results to be returned (hence the term "nonatomic"); that is, the agent will return those values it can and ignore the rest of the variables in the command. Again, this improves efficiency by reducing the number of exchanges across the network.

Decentralized Network Management

In a traditional centralized network management scheme, one host in the configuration has the role of a network management station; there may possibly be one or two other management stations in a backup role. The remainder of the devices on the network contain agent software and an MIB, to allow monitoring and control from the management station. As networks grow in size and traffic load, such a centralized system is unworkable. Too much burden is placed on the management station, and there is too much traffic, with reports from every single agent having to wend their way across the entire network to headquarters. In such circumstances, a decentralized, distributed approach works best (e.g., Fig. 5). In a decentralized network management scheme, there may be multiple top-level management stations, which might be referred to as *management servers*. Each such server might directly manage a portion of the total pool of agents. However, for many of the agents, the management server delegates responsibility to an intermediate manager. The intermediate manager plays the role of manager to monitor and control the agents under its responsibility. It also plays an agent role to provide information and accept control from a higher-level management server. This type of architecture spreads the processing burden and reduces total network traffic.

To support manager-to-manager cooperation, SNMPv2


GetBulkRequest (nonrepeaters = 2,
max-repetitions = 6, X, Y, TA, TB, TC)
Manager issues request with six variable names; for the first two variables (nonrepeaters = 2), a single value is requested; for the remaining variables, six successive values (max-repetitions = 6) are requested.
Agent
(e.g., router)
X
Y
Table α
TA
TB
TC
Agent returns single value for X, Y, and six rows of table α.
Management workstation
Response [X, Y, TA(1), TB(1), TC(1),
TA(2), TB(2), TC(2),
TA(3), TB(3), TC(3),
TA(4), TB(4), TC(4),
TA(5), TB(5), TC(5),
TA(6), TB(6), TC(6)]

■ **Figure 4.** *GetBulkRequest command.*

RFC	Title	Date
1901	Introduction to Community-Based SNMPv2	January 1996
1902	Structure of Management Information for SNMPv2	January 1996
1903	Textual Conventions for SNMPv2	January 1996
1904	Conformance Statements for SNMPv2	January 1996
1905	Protocol Operations for SNMPv2	January 1996
1906	Transport Mappings for SNMPv2	January 1996
1907	Management Information Base for SNMPv2	January 1996
1908	Coexistence between Version 1 and Version 2 of the Internet-Standard Network Management Framework	January 1996

■ **Table 3.** *Key SNMPv2 RFCs.*

introduces two new features: an Inform command and a manager-to-manager MIB. A manager uses the Inform command to send unsolicited information to another manager. For example, using the Inform command, a manager can notify another manager when some unusual event occurs, such as the loss of a physical link or an excessive rate of traffic at some point in the network. This information is defined in the manager-to-manager MIB. Such unsolicited notifications provide an ideal tool for configuring a decentralized network management scheme. Higher-level managers need not concern themselves with the details of remote parts of the network; for example, when a local event that requires central attention occurs, the local manager can use the Inform command to alert the central manager. This ability for one manager to alert another is lacking in SNMPv1.

SNMPv3

In September 1996, the IETF formed an advisory committee to analyze the competing proposed approaches to SNMP security. In early 1997, this committee produced a white paper describing SNMPng, or next generation (available at http://www.tis.com/docs/research/network/snmp-ng.html). SNMPng includes the functionality of SNMPv2 and incorporates security features found in the proposed security approaches. With further refinement and implementation experience, SNMPng is intended to become SNMPv3. To that end, the Internet Engineering Task Force (IETF) chartered an SNMPv3 working group to prepare RFCs for SNMPv3. As of this writing, the working group has produced a set of Internet Drafts (available at http://www.ietf.org/html.charters/snmpv3-charter.html). The group expects to produce RFCs by the end of 1997, with a goal of submitting a complete set of SNMPv3 specifications for consideration as Proposed Standards by April 1998. Products based on SNMPv3 are likely to become available in 1998.

SNMPv3 consists of three modules. The Message Processing and Control module handles SNMP message creation and parsing functions, and also determines if proxy handling is required for any SNMP message. The Local Processing module performs access control for variable binding data, processing that data, and trap processing. The Security module provides authentication and encryption functions, and checks the timeliness of certain SNMP messages.

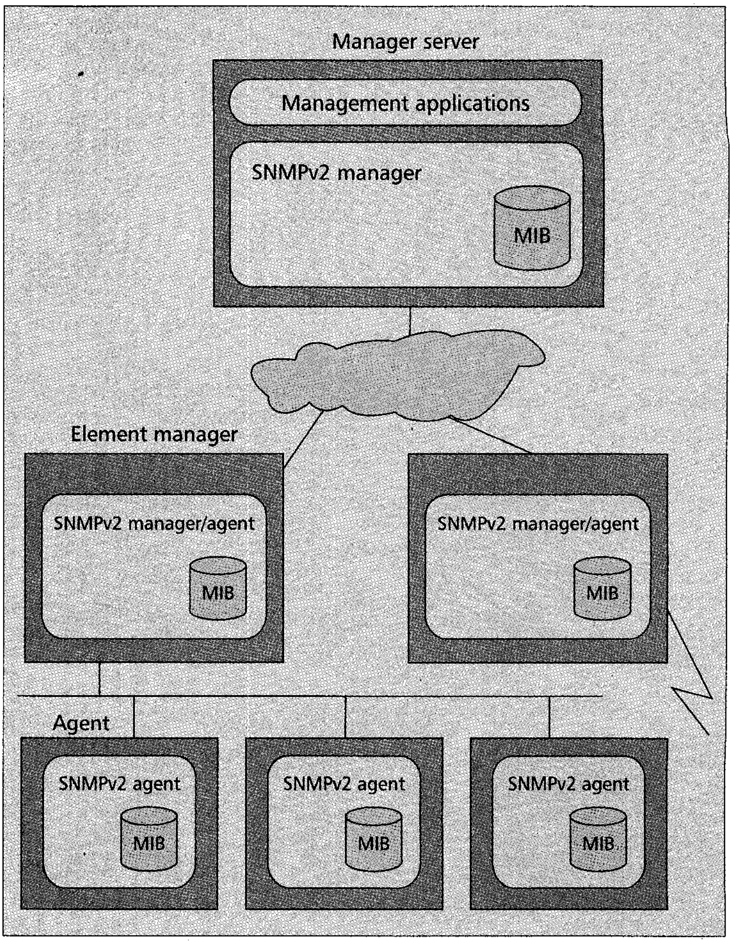
The most substantial improvement SNMPv3 offers over SNMPv1 and SNMPv2 is the addition of security features. This deals with one of the major concerns that users of SNMP have expressed: its lack of effective security. Specifically, users want to know that only authorized personnel are able to perform network management functions (e.g., disable/enable a line) and that only authorized personnel are able to read network management information (e.g., contents of a configuration file).

The three new security features provided by SNMPv3 are authentication, secrecy, and access control. Authentication enables an agent to verify that an incoming command is from an authorized manager and that the contents of the command have not been altered. To achieve this, each manager and agent that wish to communicate must share a secret key. The manager uses this key to calculate a message authentication code which is a function of the message to be transmitted and appends that code to the message. When the agent receives the message, it uses the same key and calculates the message authentication code once again. If the agent's version of the code matches the value appended to the incoming message, then the agent knows that the message can only have originated from the authorized manager, and that the message was not altered in transit.

The secrecy facility enables managers and agents to encrypt messages to prevent eavesdropping by third parties. Again, manager and agent share a secret key. In this case, if the two are configured to use the secrecy facility, all traffic between them is encrypted.

Finally, the access control facility makes it possible to configure agents to provide different levels of access to different managers. Access can be limited in terms of the commands the agent will accept from a given manager and also in terms of the portion of the agent's MIB a given manager may access. The access control policy to be used by an agent for each manager must be preconfigured and essentially consists of a table that details the access privileges of the various authorized managers.

With these new security features, network managers should have a much greater comfort level in using SNMPv2, particularly in large installations and/or those with a large user population.



■ **Figure 5.** *SNMPv2-managed configuration.*

Conclusion

SNMPv2 is a substantial improvement over SNMPv1, while retaining its essential character of ease of understanding and implementation. Version 2 provides better support for a decentralized network management architecture, enhances performance, and provides a few other bells and whistles of interest to application developers.

SNMPv3 fixes the most obvious failing of versions 1 and 2: lack of security. There is now, at last, a worthy successor to SNMPv1, and the new standard should succeed in the marketplace. Vendors are likely to adopt the new version to provide more features and more efficient operation to their users. Also, we can expect additional MIBs to be defined within the SNMPv3 framework to extend its scope of support various network management applications.

Additional Reading

More detail on SNMPv1 and SNMPv2 can be found in [1]. Good coverage of SNMPv2 can also be found in [2]. The Web site at http://netman.cit.buffalo.edu/index.html is a good source of information on SNMP and other network management topics. The site has links to many of the vendors who offer SNMP, RMON, and other network management products.

[1] W. Stallings, *SNMP, SNMPv2, and RMON: Practical Network Management*, 2nd ed., Reading, MA: Addison-Wesley, 1996.
[2] M. Rose, *The Simple Book: An Introduction to Network Management*, 3rd ed., Upper Saddle River, NJ: Prentice Hall, 1996.

Biography

William Stallings (ws@shore.net) is a consultant, lecturer, and author of over a dozen professional reference books and textbooks on data communications and computer networking. He has twice received the award for the best Computer Science textbook of the year from the Text and Academic Authors Association (1996: *Computer Organization and Architecture*, 4th ed.; 1997: *Data and Computer Communications*, 5th ed.). He has a Ph.D. from M.I.T. in computer science. His home in cyberspace is http://www.shore.net/~ws.