CHAPTER 32

Simple Network Management Protocol

Background

Simple Network Management Protocol (SNMP) is an application-layer protocol designed to facilitate the exchange of management information between network devices. By using SNMP to access management information data (such as packets per second and network error rates), network administrators can more easily manage network performance and find and solve network problems.

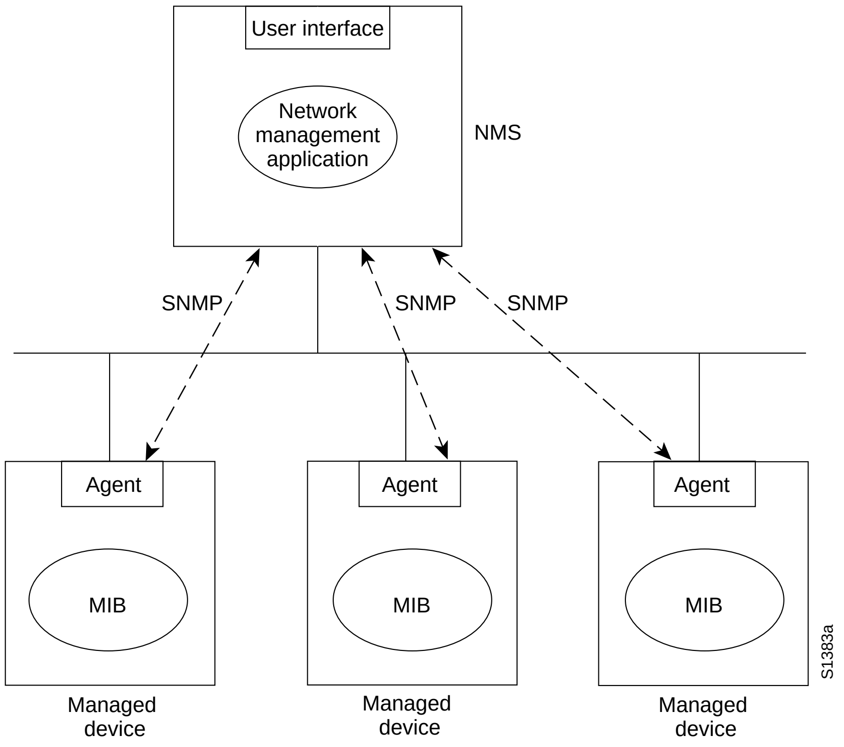
Today, SNMP is the most popular protocol for managing diverse commercial, university, and research internetworks. Standardization activity continues even as vendors develop and release state-of-the-art SNMP-based management applications. SNMP is a relatively simple protocol, yet its feature set is sufficiently powerful to handle the difficult problems presented by management of heterogeneous networks.

This chapter discusses two versions of SNMP: Version 1.0, which was the initial version of SNMP, and Version 2.0, which incorporates security features as well as improvements in protocol operations and management architecture.

SNMP Version 1.0

In SNMP v.1, agents are software modules that run in managed devices. Agents have access to information about the managed devices in which they run and make this information available to *network management systems* (NMSs) via SNMP v.1. This model is graphically represented in Figure 32-1.

Figure 32-1 SNMP v.1 Management Model



A managed device can be any type of node residing on a network, including computer hosts, communication servers, printers, routers, bridges, and hubs. Because some of these devices may have limited ability to run management software (they may have relatively slow CPUs or limited memory, for example), management software must assume the lowest common denominator. In other words, management software must be built in such a way as to minimize its own performance impact on the managed device.

Because managed devices contain a lowest common denominator of management software, the management burden falls on the NMS. Therefore, NMSs are typically engineering workstation-caliber computers that have fast CPUs, megapixel color displays, substantial memory, and lots of disk space. One or more NMSs can exist on any managed network. NMSs run the network management applications that present management information to users. The user interface is typically based on a standardized *graphical user interface* (GUI).

Communication between managed devices and NMSs is governed by the network management protocol. The Internet-standard Network Management Framework assumes a remote-debugging paradigm, where managed devices maintain values for a number of variables and report those, on demand, to NMSs. For example, a managed device might keep track of the following:

- Number and state of its virtual circuits
- Number of certain kinds of error messages received
- Number of bytes and packets in and out of the device
- Maximum output queue length (for routers and other internetworking devices)
- Broadcast messages sent and received
- Network interfaces going down and coming up

Command Types

If an NMS wishes to control a managed device, it can do so by sending a message requiring the managed device to change the value of one or more of its variables. Managed devices respond to or initiate four different types of commands:

- *Reads*—Used by NMSs to monitor managed devices. NMSs read variables maintained by the devices.
- *Writes*—Used by NMSs to control managed devices. NMSs write variables stored within the managed devices.
- *Traversal operations*—Used by NMSs to determine which variables a managed device supports and to sequentially gather information in variable tables (such as an Internet Protocol [IP] routing table).
- *Traps*—Used by managed devices to asynchronously report certain events to NMSs.

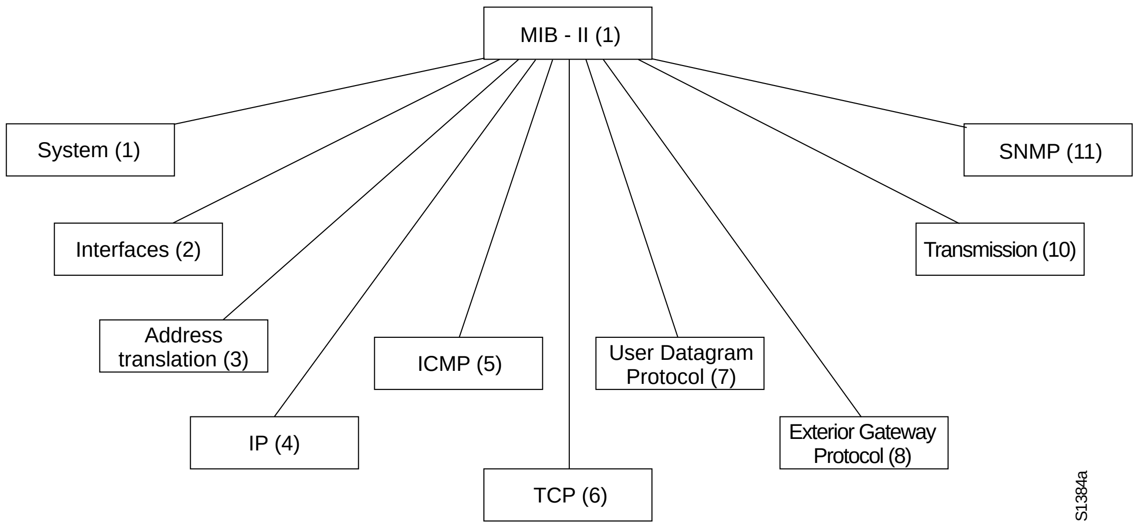
Data Representation Differences

Exchange of information in a managed network is potentially compromised by differences in the data-representation techniques used by the managed devices. In other words, various computers represent information differently; these incompatibilities must be rationalized to allow communication between diverse systems. The combination of an abstract and a transfer syntax performs this function. SNMP v.1 uses a subset of *Abstract Syntax Notation One* (ASN.1), an abstract syntax defined as part of OSI. ASN.1 is used to define packet formats and managed objects. A managed object is simply a characteristic of things that can be managed. A managed object differs from a variable, which is an instance of a managed object. Managed objects can be scalar (meaning that there is always exactly one instance) or tabular (meaning that there may be zero, one, or more instances).

Management Database

All managed objects are contained in the *Management Information Base* (MIB), which is essentially a database of objects. A MIB is depicted as a tree, with individual data items as leaves. Object identifiers uniquely identify MIB objects in the tree. Object identifiers are like telephone numbers, in that they are organized hierarchically, and parts are assigned by different organizations. For example, international telephone numbers consist of country codes (assigned by an international organization) and the telephone number, as defined by that country. U.S. telephone numbers are further subdivided into an area code, a central office number, and a station number associated with that central office. Top-level MIB object identifiers are assigned by the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC). Lower-level object IDs are allocated by the associated organizations. Several prominent branches of the MIB tree appear in Figure 32-2.

Figure 32-2 MIB Tree



The MIB tree is extensible by virtue of experimental and private branches (not shown in Figure 32-2). Vendors, for example, can define their own private branches to include instances of their own products. MIBs that have not been standardized are typically positioned in the experimental branch.

The *Structure of Management Information* (SMI) document defines the structure of the MIB and the rules for defining MIBs. The SMI allows the use of standard ASN.1 data types: INTEGER, OCTET STRING, and OBJECT IDENTIFIER. It also defines the following data types:

- *Network addresses*—Represent an address from a particular protocol family. SNMP v.1 only supports 32-bit IP addresses.
- *Counters*—Nonnegative integers that increase until they reach a maximum value, when they wrap back to zero. The total number of bytes received on an interface is an example of a counter.
- *Gauges*—Nonnegative integers that can increase or decrease, but retain the maximum value reached. The length of an output packet queue (in packets) is an example of a gauge.
- *Time ticks*—Hundredths of a second since some event. The time since an interface entered its current state is an example of a time tick.
- *Opaque*—An arbitrary encoding used to pass arbitrary information strings outside of the strict data typing used by the SMI.

Operations

SNMP v.1 itself is a simple request-response protocol. Four SNMP v.1 operations are defined:

- *Get*—Retrieves an object instance from the agent.
- *Get-next*—Retrieves the next object instance from a table or list within an agent.
- *Set*—Sets object instances within an agent.
- *Trap*—Informs asynchronously the NMS of some event. Unlike the get, get-next, and set operations, the trap does not elicit a response from the receiver.

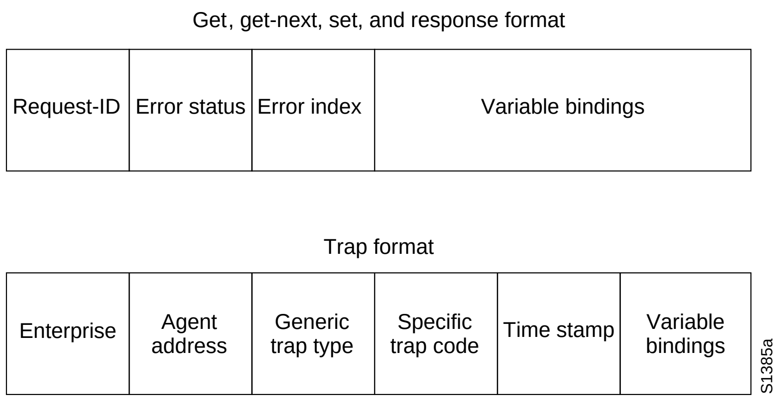
Message Format

SNMP v.1 messages contain two parts, a message header and a protocol data unit (PDU). The message header consists of a version number and a *community name*. The community name serves two functions. First, the community name defines an access environment for a set of NMSs using that community name. Second, because devices that do not know the proper community name are precluded from SNMP v.1 operations, network managers have also used the community name as a weak form of authentication.

The data portion of an SNMP v.1 message contains the specified SNMP v.1 operation (get, set, and so on) and associated operands. Operands indicate the object instances involved in the transaction.

Figure 32-3 shows the SNMP v.1 PDU format. All of the fields are variable in length as prescribed by the ASN.1 encoding rules.

Figure 32-3 SNMP v.1 Message Format



SNMP v.1 get, get-next, response, and set PDUs consist of the following fields:

- *Request-ID*—Associates requests with responses.
- *Error status*—Indicates an error and an error type.
- *Error index*—Associates the error with a particular variable in the variable bindings.
- *Variable bindings*—Comprises the data of an SNMP v.1 PDU. Each variable binding associates a particular variable with its current value (with the exception of get and get-next requests, for which the value is ignored).

Trap PDUs are slightly different from the get, get-next, response, and set PDUs. They consist of the following fields:

- *Enterprise*—Identifies the type of object generating the trap.
- *Agent address*—Provides the address of the object generating the trap.
- *Generic trap type*—Provides the generic trap type.
- *Specific trap code*—Provides the specific trap code.
- *Time stamp*—Provides the amount of time that has elapsed between the last network reinitialization and generation of this trap.
- *Variable bindings*—Provides a list of variables containing interesting information about the trap.

SNMP Version 2.0

SNMP Version 2.0 is an evolution of the initial version SNMP (now called SNMP v.1) derived from two specifications published in July 1992: Secure SNMP and the Simple Management Protocol (SMP).

Secure SNMP defined security features that are not available in SNMP v.1, but Secure SNMP used message formats that are incompatible with SNMP v.1. Compared with SNMP v.1, SMP offered greater flexibility in terms of the resources it can manage, the size of data transfers, and the environments in which it can operate (OSI networks and other communications architectures in addition to TCP/IP networks.) A subset of SMP capabilities allowed it to interoperate with SNMP v.1.

As the Internet community analyzed these two new specifications, a consensus emerged that SMP should serve as the basis for the development of a new SNMP standard and that many of the security features from Secure SNMP would be incorporated into that new standard. The result is SNMP v.2, which was published as a set of proposed Internet standards in the spring of 1993.

SNMP v.2 includes improvements in the SMI, in protocol operations, in management architecture, and in security.

SMI

The SNMP v.2 SMI supports several new data types and incorporates a new convention for creating and deleting conceptual rows in a table. The network address data type now supports OSI NSAP addresses, as well as IP addresses. There is now a 64-bit counter data type, as well as the existing 32-bit counter, and there is now an unsigned integer type to represent integers in the range 0 to $2^{32}–1$.

SNMP v.2 introduces the concept of an information module, which specifies a group of related definitions. There are three kinds of information modules:

- MIB modules—Contain definitions of interrelated managed objects.
- Compliance statements for MIB modules—Provide a systematic way to describe the group of managed objects that must be implemented for conformance.
- Capability statements for agent implementations—Define the precise level of support that an agent claims with respect to a MIB group. The statements may indicate that some objects have restricted or augmented syntax or access levels. A formal definition of agent capabilities can be useful in promoting and optimizing interoperability among vendors. A management station that has the capabilities statement for each of the agents with which it interacts can adjust its behavior to optimize the use of its own resources and the resources of the agent and the network.

Protocol Operations

SNMP v.2 defines two new operations:

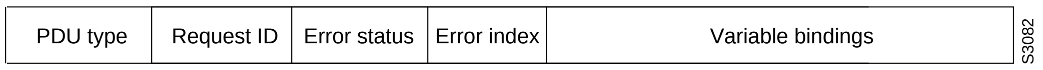
- *Inform*—Allows one manager to send trap type information to another manager and request a response.
- *Get-bulk*—Allows a manager to retrieve efficiently large blocks of data, such as multiple rows in a table, which would otherwise require the transmission of many small blocks of data.

With the exception of the way SNMP v.2 responds to get, get-next, set, and trap operations, the SNMP v.2 versions of these operations are identical to their SNMP v.1 counterparts. In SNMP v. 1, if the responding agent cannot provide values for all the variables in the list, it does not provide any values. In SNMP v.2, a variable binding list is prepared even if values cannot be supplied for all variables. If an exception condition is associated with the variable, the variable is paired with the name of that exception condition.

Message Format

To simplify PDU processing, all operations except the get-bulk operation use the same PDU format. Figure 32-4 shows the PDU format used by the get, get-next, set, response, and trap operations, as well as the new inform operation.

Figure 32-4 PDU Format for the Get, Get-Next, Inform, Response, Set, and SNMP v.2 Trap Operations



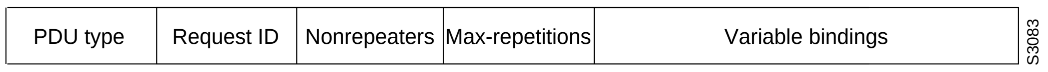
The fields of the PDU for get, get-next, set, response, and trap operations are as follows:

- *PDU type*—Identifies the PDU type (get, get-next, set, response, or trap).
- *Request ID*—Associates requests with responses.
- *Error status*—Indicates an error and an error type.
- *Error index*—Associates the error with a particular variable in the variable bindings.
- *Variable bindings*—Associates particular variables with their current values (with the exception of get and get-next requests, for which the value is ignored).

When used by the get, get-next, set, trap, and inform operations, the error status and error index fields are set to 0. Only the response operation sets the error status and error index fields.

The get-bulk request operation uses the PDU format shown in Figure 32-5.

Figure 32-5 PDU Format for the Get-Bulk Operation



The fields of the PDU for the get-bulk operation are as follows:

- *PDU type*, *request ID*, and *variables bindings*—Serve the same function as the PDU for get, get-next, set, response, and trap operations.
- *Nonrepeaters*—Specifies the number of variables in the variable bindings list for which a single lexicographic successor is to be returned.
- *Max-repetitions*—Specifies the number of lexicographic successors to be returned for the remaining variables in the variable bindings list.

Management Architecture

SNMP v.2 supports the centralized network management strategies of SNMP v.1 as well as distributed strategies based on the new manager-to-manager MIB. In a distributed architecture, some systems operate both in the role of manager and of agent. When acting as an agent, a system accepts commands from a superior management system. These commands can deal with access to information stored locally or can require the system (now acting as an intermediate manager) to provide summary information about subordinate agents. In addition, an intermediate manager can issue trap information to a superior manager.

Security

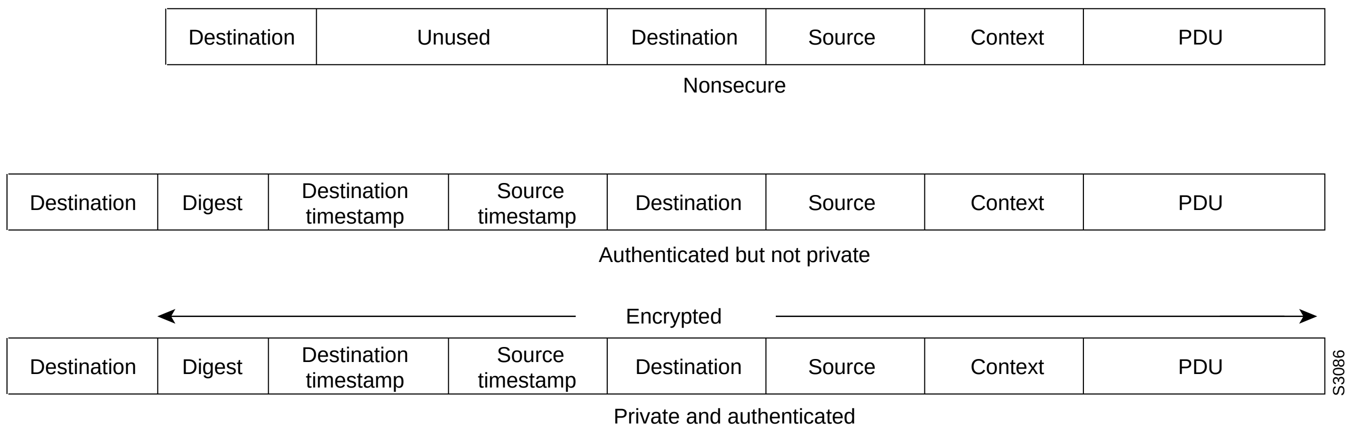
The lack of authentication capability is the most serious security deficiency of SNMP v.1, making it vulnerable to unauthorized changes to the configuration of a network. For that reason, many vendors have not implemented set operations, thereby reducing SNMP v.1 to a monitoring facility.

SNMP v.2 includes provisions for preventing the following types of security threats:

- *Masquerades*—An unauthorized entity may attempt to perform management operations by assuming the identity of an authorized entity.
- *Modification of information*—An entity may alter a message generated by an authorized entity so that the message results in unauthorized management operations, including operations related to configuration and accounting.
- *Message sequence and timing modifications*—Because SNMP v.1 is designed to operate over a connectionless transport, an entity could reorder, delay, or copy and later replay an SNMP v.1 message. For example, an entity could copy and then replay a message to reboot a device.
- *Disclosures*—An entity can learn the values of managed objects and learn of notifiable events by monitoring exchanges between a manager and an agent.

  A change in the message format allows SNMP v.2 to improve the security of message exchanges. Figure 32-6 shows the new message format.

Figure 32-6 SNMP v.2 Message Format



The fields of the SNMP v.2 message formats are as follows:

- *Destination*—Identifies the recipient. This field appears twice in the SNMP v.2 message format—at the beginning of the message, where it is remains unencrypted so that the destination entity can determine the privacy characteristics of the message, and again in a part of the message that is subject to encryption.
- *Source*—Identifies the sender.
- *Context*—Identifies the collection of managed object resources that are accessible by an SNMP v.2 entity. The context field replaces the community name portion of an SNMP v.1 message header.
- *PDU*—Identifies the desired management operation.
- *Digest*—Contains the value resulting from the calculation of the message-digest algorithm over a portion of the message format.
- *Destination timestamp*—Contains the value that the sender has for the receiver's clock obtained from a previous exchange of messages.
- *Source timestamp*—Contains the value of the sender's clock.

SNMP v.2 supports three uses of the message format:

- *Nonsecure*—Messages for which none of the SNMP v.2 security measures are taken.
- *Authenticated but not private*—SNMP v.2 uses a secret value, known only the sender and recipient, to authenticate the sender of a message. The message sender prepends a secret value, already known to the message recipient, to the message and calculates the digest over the message, including the secret value. Then, the sender overwrites the secret value with the digest value and sends the message. The recipient, which already has the secret value, recomputes the digest using a local copy of the secret value. If the incoming message digest matches the calculated digest, the receiver accepts the message as authentic.
- *Private and authenticated*—Messages that are encrypted and authenticated.

Through the authentication procedure, SNMP v.2 can assure that messages are received as sent, without modification or replays. A message-digest algorithm calculates a 128-bit digest over the appropriate portion of an SNMP v.2 message. The digest added to the message is recalculated by the recipient to verify that there has been no modification. The timestamps are based on the maintenance of loosely synchronized clocks among managers and agents. The message recipient uses the timestamp to verify that the message is recent, to determine the proper resequencing of multiple messages, and to detect message replays.

Interoperability Issues

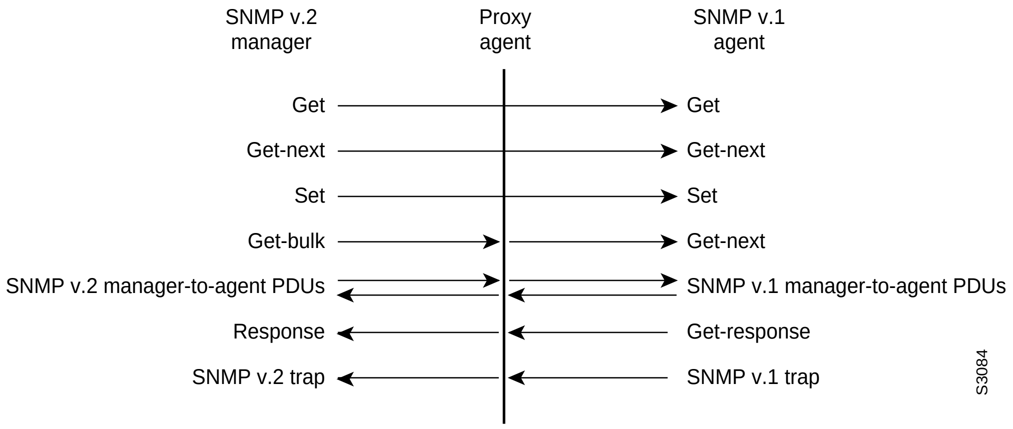
The new message formats, protocol operations, and security features make SNMP v.2 incompatible with SNMP v.1. The easiest way to accomplish to transition an existing network from SNMP v.1 to SNMP v.2 is to upgrade the manager systems to support SNMP v.2 in a way that allows coexistence of SNMP v.2 managers, SNMP v.2 agents, and SNMP v.1 agents. The issues fall into two categories:

- *Management information*—The SNMP v.2 SMI is nearly a proper superset of the SMI for SNMP v.1. SNMP v.2 largely codifies the existing practice for defining MIB modules. Modules defined to conform to the SNMP v.1 SMI can still be used with SNMP v.2. That is, it is possible for an agent to maintain an unchanged SNMP v.1 MIB and still coexist in an SNMP v.2 environment. To bring an SNMP v.1 MIB into conformance with SNMP v.2 requires syntactic changes in object definitions, and trap definitions, as well as the addition of compliance definitions.

- *Protocol operations*—The protocol defined in the SNMP v.2 framework uses the same PDU formats with extensions to the set of PDUs for the get-bulk and inform operations with a change in semantics to allow get operations to provide partial results rather than operating in an all-or-none manner.

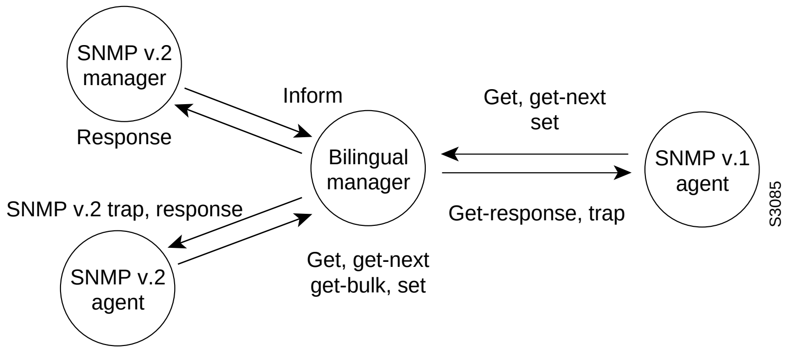
One way to achieve coexistence at the protocol level is to reach existing SNMP v.1 agents through an SNMP v.2 agent that acts as a proxy agent on behalf of an SNMP v.2 manager. The proxy agent maps get-bulk PDUs to get-next PDUs and SNMP v.1 trap PDUs to SNMP v.2 trap PDUs, but passes get, get-next, and set PDUs unchanged, as shown in Figure 32-7.

Figure 32-7 Mapping Approach for Interoperability



Another way to achieve coexistence at the protocol level is to use a "bilingual" SNMP v.2 manager that supports both SNMP v.1 and SNMP v.2. When a management application needs to contact an agent, it can use SNMP v.1 or SNMP v.2, based on information in a local database that identifies the agent as supporting SNMP v.1 or SNMP v.2, as shown in Figure 32-8.

Figure 32-8 Bilingual Manager Approach for Interoperability


SNMP v.2 manager
Inform
Response
Get, get-next set
Bilingual manager
SNMP v.1 agent
Get-response, trap
SNMP v.2 trap, response
SNMP v.2 agent
Get, get-next get-bulk, set
S3085