



# Chapter 28 Network Management: SNMP

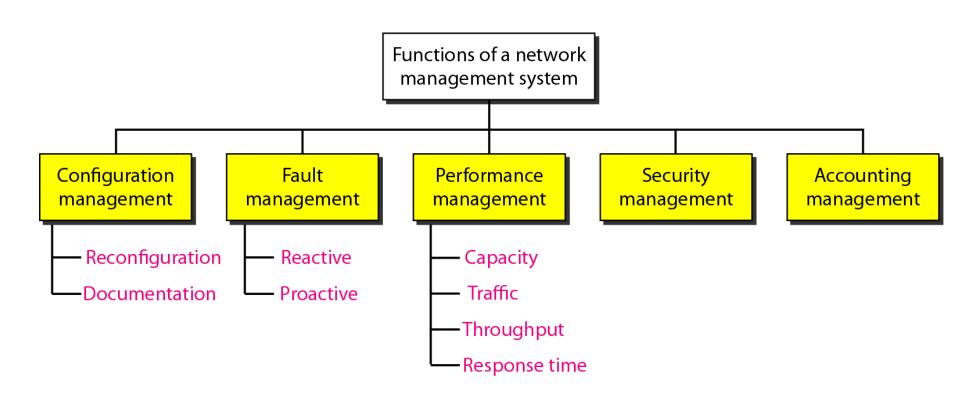
### 28-1 NETWORK MANAGEMENT SYSTEM

We can say that the functions performed by a network management system can be divided into five broad categories: configuration management, fault management, performance management, security management, and accounting management.

### Topics discussed in this section:

Configuration Management
Fault Management
Performance Management
Security Management
Accounting Management

### Figure 28.1 Functions of a network management system



### 28-2 SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

The Simple Network Management Protocol (SNMP) is a framework for managing devices in an internet using the TCP/IP protocol suite. It provides a set of fundamental operations for monitoring and maintaining an internet.

### Topics discussed in this section:

Concept

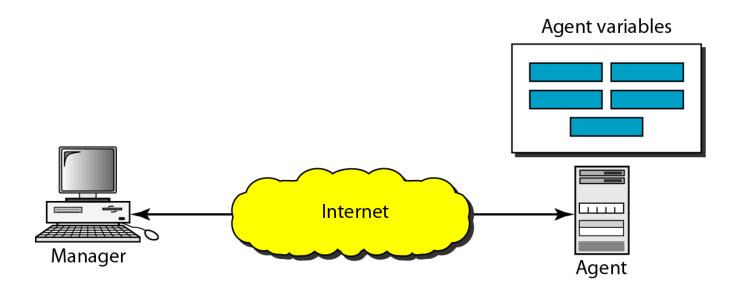
**Management Components** 

**Structure of Management Information (SMI)** 

**Management Information Base (MIB)** 

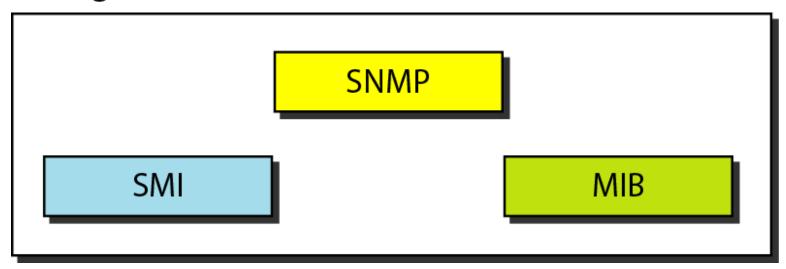
**SNMP** 

### Figure 28.2 SNMP concept



### Figure 28.3 Components of network management on the Internet

### Management



### Note

SNMP defines the format of packets exchanged between a manager and an agent. It reads and changes the status (values) of objects (variables) in SNMP packets.

## Note

SMI defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values. SMI does not define the number of objects an entity should manage or name the objects to be managed or define the association between the objects and their values.

### -

### Note

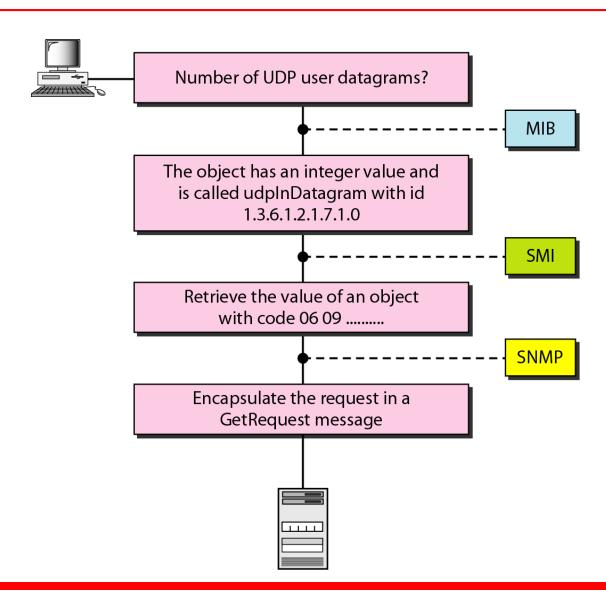
MIB creates a collection of named objects, their types, and their relationships to each other in an entity to be managed.



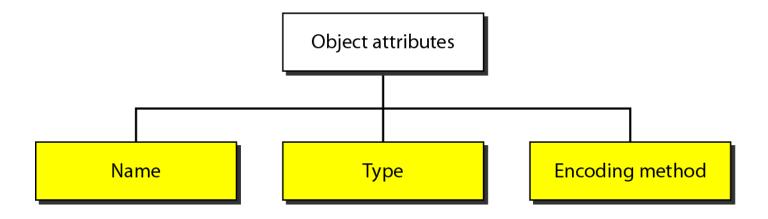
We can compare the task of network management to the task of writing a program.

- Both tasks need rules. In network management this is handled by SMI.
- Both tasks need variable declarations. In network management this is handled by MIB.
- Both tasks have actions performed by statements. In network management this is handled by SNMP.

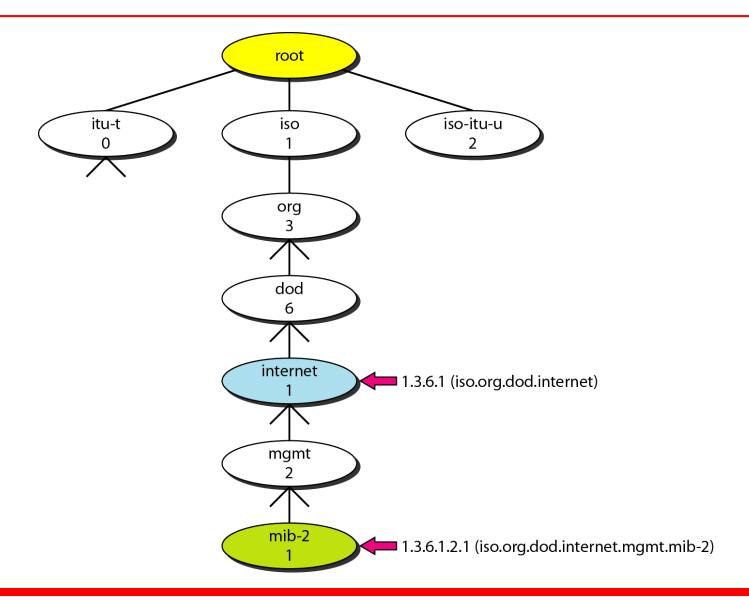
### Figure 28.4 Management overview



### Figure 28.5 Object attributes



### Figure 28.6 Object identifier



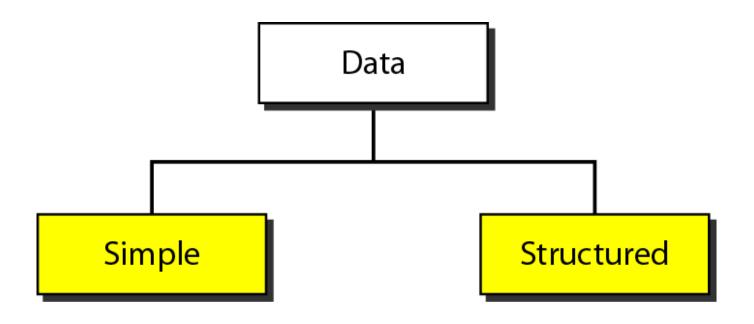


### Note

All objects managed by SNMP are given an object identifier.

The object identifier always starts with 1.3.6.1.2.1.

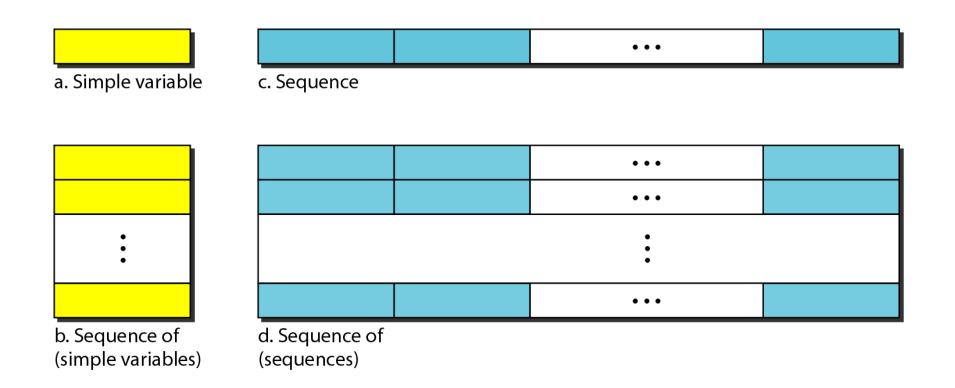
### Figure 28.7 Data type



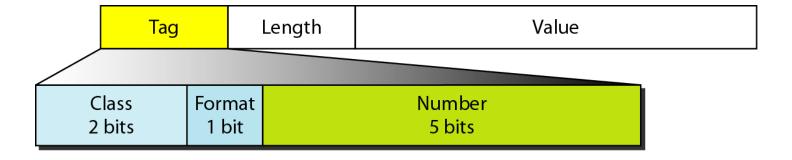
### Table 28.1 Data types

Туре	Size	Description
INTEGER	4 bytes	An integer with a value between $-2^{31}$ and $2^{31} - 1$
Integer32	4 bytes	Same as INTEGER
Unsigned32	4 bytes	Unsigned with a value between 0 and $2^{32} - 1$
OCTET STRING	Variable	Byte string up to 65,535 bytes long
OBJECT IDENTIFIER	Variable	An object identifier
IPAddress	4 bytes	An IP address made of four integers
Counter32	4 bytes	An integer whose value can be incremented from 0 to $2^{32}$ ; when it reaches its maximum value, it wraps back to 0.
Counter64	8 bytes	64-bit counter
Gauge32	4 bytes	Same as Counter32, but when it reaches its maximum value, it does not wrap; it remains there until it is reset
TimeTicks	4 bytes	A counting value that records time in $\frac{1}{100}$ s
BITS		A string of bits
Opaque	Variable	Uninterpreted string

### Figure 28.8 Conceptual data types



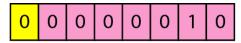
### Figure 28.9 Encoding format



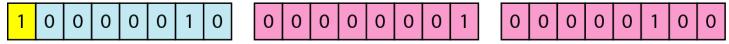
### Table 28.2 Codes for data types

Data Type	Class	Format	Number	Tag (Binary)	Tag (Hex)
INTEGER	00	0	00010	00000010	02
OCTET STRING	00	0	00100	00000100	04
OBJECT IDENTIFIER	00	0	00110	00000110	06
NULL	00	0	00101	00000101	05
Sequence, sequence of	00	1	10000	00110000	30
IPAddress	01	0	00000	01000000	40
Counter	01	0	00001	01000001	41
Gauge	01	0	00010	01000010	42
TimeTicks	01	0	00011	01000011	43
Opaque	01	0	00100	01000100	44

### Figure 28.10 Length format



a. The colored part defines the length (2).



b. The shaded part defines the length of the length (2 bytes); the colored bytes define the length (260 bytes).

### Figure 28.11 shows how to define INTEGER 14.

#### Figure 28.11 Example 28.1, INTEGER 14

02	04	00	00	00	OE
00000010	00000100	00000000	00000000	00000000	00001110
Tag (integer)	Length (4 bytes)		Value	2 (14)	

### Figure 28.12 shows how to define the OCTET STRING "HI".

#### Figure 28.12 Example 28.2, OCTET STRING "HI"

04	02	48	49
00000100	0000010	01001000	01001001
Tag (String)	Length (2 bytes)	Value (H)	Value (I)

Figure 28.13 shows how to define ObjectIdentifier 1.3.6.1 (iso.org.dod.internet).

#### Figure 28.13 Example 28.3, ObjectIdentifier 1.3.6.1

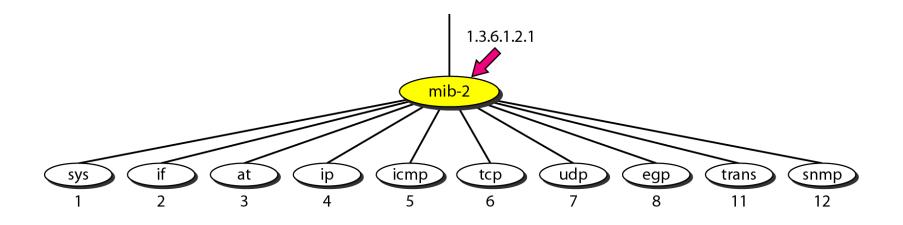
06	04	01	03	06	01		
00000110	00000100	00000001	00000011	00000110	0000001		
Tag (ObjectId)	Length (4 bytes)	Value (1)	Value (3)	Value (6)	Value (1)		
		← 1.3.6.1 (iso.org.dod.internet) ← →					

Figure 28.14 shows how to define IPAddress 131.21.14.8..

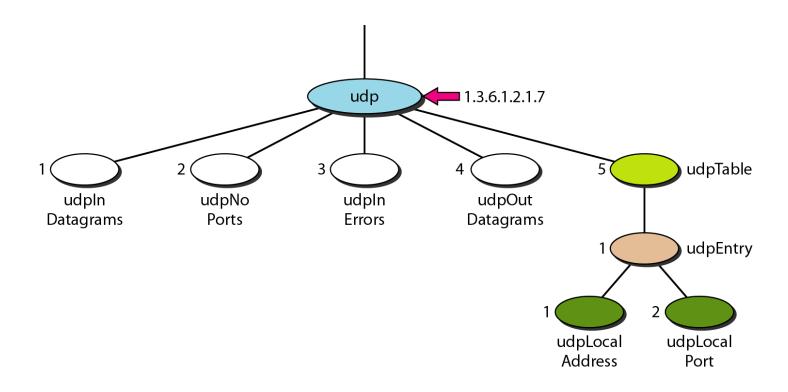
### Figure 28.14 Example 28.4, IPAddress 131.21.14.8.

40	04	83	15	OE	08	
01000000	00000100	10000011	00010101	00001110	00001000	
Tag (IPAddress)	Length (4 bytes)	Value (131)	Value (21)	Value (14)	Value (8)	
		131.21.14.8				

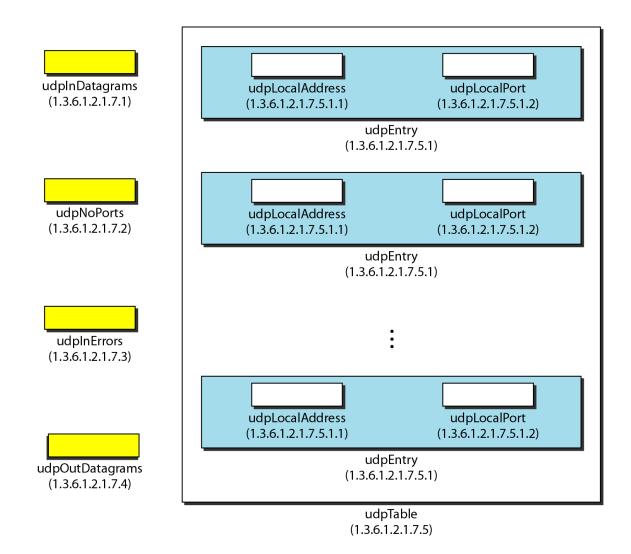
### **Figure 28.15** *mib-2*



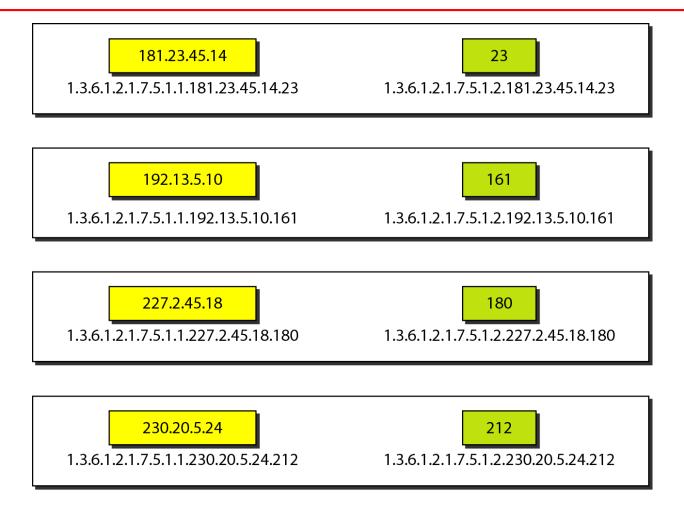
### Figure 28.16 udp group



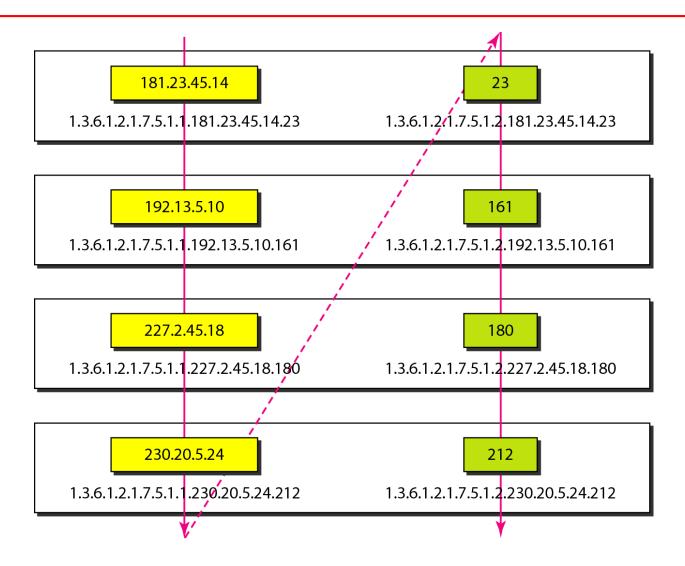
### Figure 28.17 udp variables and tables



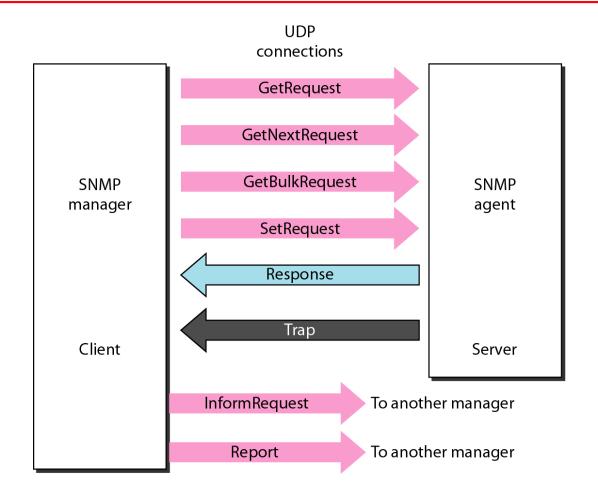
### Figure 28.18 Indexes for udpTable



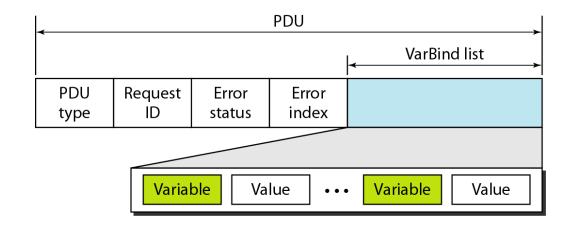
### Figure 28.19 Lexicographic ordering



### Figure 28.20 SNMP PDUs



### Figure 28.21 SNMP PDU format



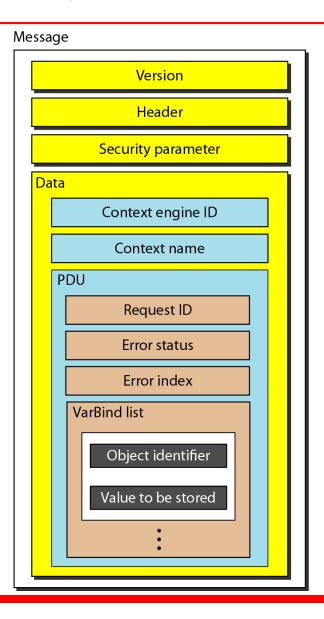
#### Differences:

- 1. Error status and error index values are zeros for all request messages except GetBulkRequest.
- 2. Error status field is replaced by nonrepeater field and error index field is replaced by max-repetitions field in GetBulkRequest.

### Table 28.3 Types of errors

Status	Name	Meaning
0	noError	No error
1	tooBig	Response too big to fit in one message
2	noSuchName	Variable does not exist
3	badValue	The value to be stored is invalid
4	readOnly	The value cannot be modified
5	genErr	Other errors

### Figure 28.22 SNMP message



### Table 28.4 Codes for SNMP messages

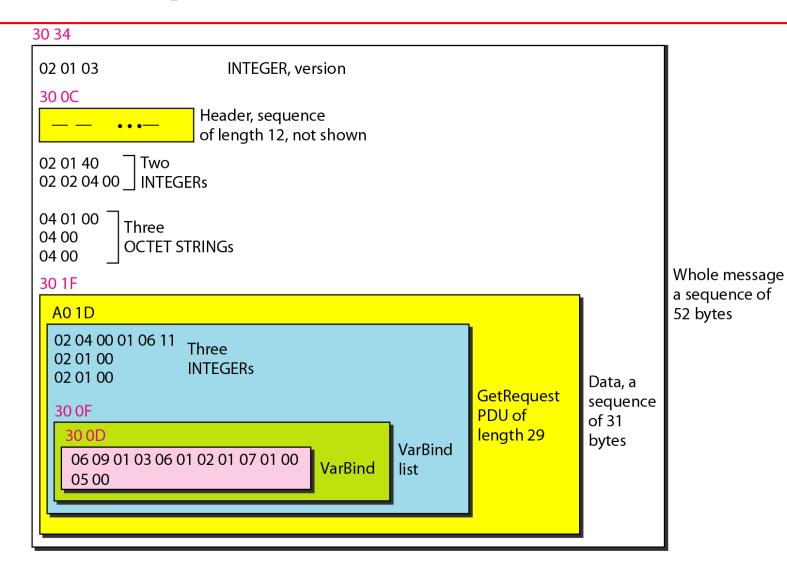
Data	Class	Format	Number	Whole Tag (Binary)	Whole Tag (Hex)
GetRequest	10	1	00000	10100000	A0
GetNextRequest	10	1	00001	10100001	A1
Response	10	1	00010	10100010	A2
SetRequest	10	1	00011	10100011	A3
GetBulkRequest	10	1	00101	10100101	A5
InformRequest	10	1	00110	10100110	A6
Trap (SNMPv2)	10	1	00111	10100111	A7
Report	10	1	01000	10101000	A8

In this example, a manager station (SNMP client) uses the GetRequest message to retrieve the number of UDP datagrams that a router has received. There is only one VarBind entity. The corresponding MIB variable related to this information is udpInDatagrams with the object identifier 1.3.6.1.2.1.7.1.0. The manager wants to retrieve a value (not to store a value), so the value defines a null entity. Figure 28.23 shows the conceptual view of the packet and the hierarchical nature of sequences. We have used white and colored boxes for the sequences and a gray one for the PDU. The VarBind list has only one VarBind.

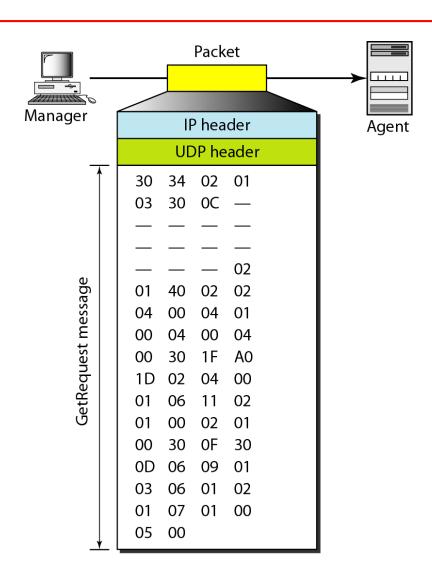
### Example 28.5 (continued)

The variable is of type 06 and length 09. The value is of type 05 and length 00. The whole VarBind is a sequence of length 0D (13). The VarBind list is also a sequence of length 0F (15). The GetRequest PDU is of length ID (29). Now we have three OCTET STRINGs related to the security parameter, security model, and flags. Then we have two integers defining maximum size (1024) and message ID (64). The header is a sequence of length 12, which we left blank for simplicity. There is one integer, version (version 3). The whole message is a sequence of 52 bytes. Figure 28.24 shows the actual message sent by the manager station (client) to the agent (server).

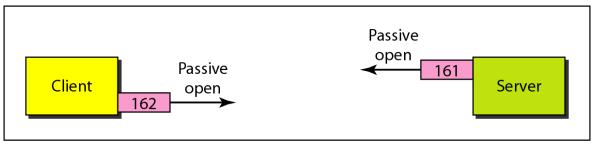
### **Figure 28.23** *Example 28.5*



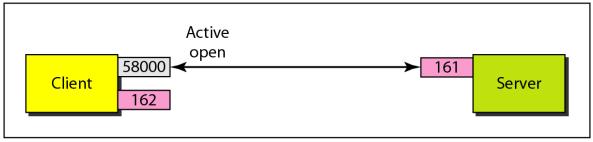
### Figure 28.24 GetRequest message



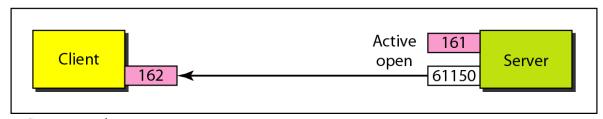
### Figure 28.25 Port numbers for SNMP



a. Passive open by both client and server



b. Exchange of request and response messages



c. Server sends trap message