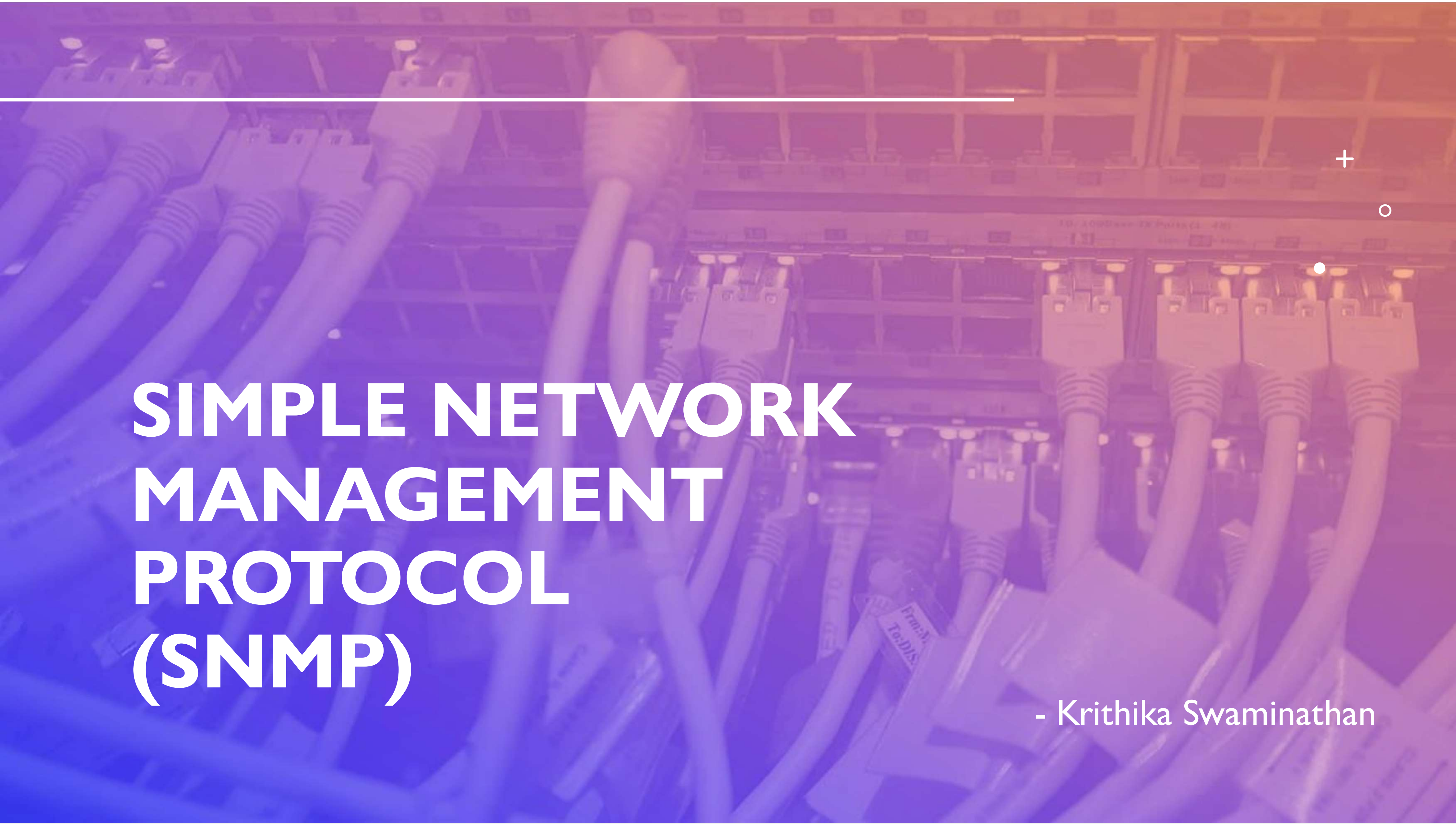


SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

- Krithika Swaminathan

Contents

- Introduction to SNMP
- Managers and Agents
- Management Components
- Management Overview
- SMI
- MIB
- SNMP

Introduction to SNMP

What is SNMP?

Framework for **managing devices** in an internet using the **TCP/IP protocol** suite

Which layer?

Application-level protocol

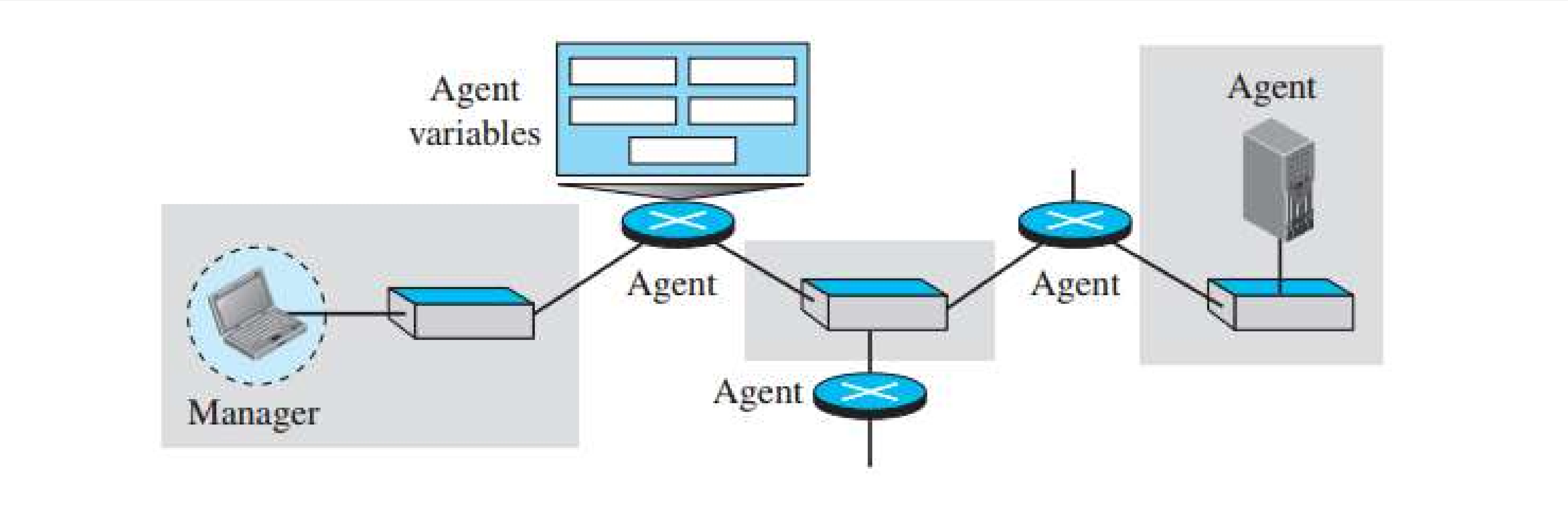
Why? **-** To monitor devices, independent of physical characteristics of managed devices and networking technology

Purpose?

Provides a set of **fundamental operations** for **monitoring and maintaining** the internet

Managers and Agents

A manager (host) controls and monitors a set of agents (routers or servers)



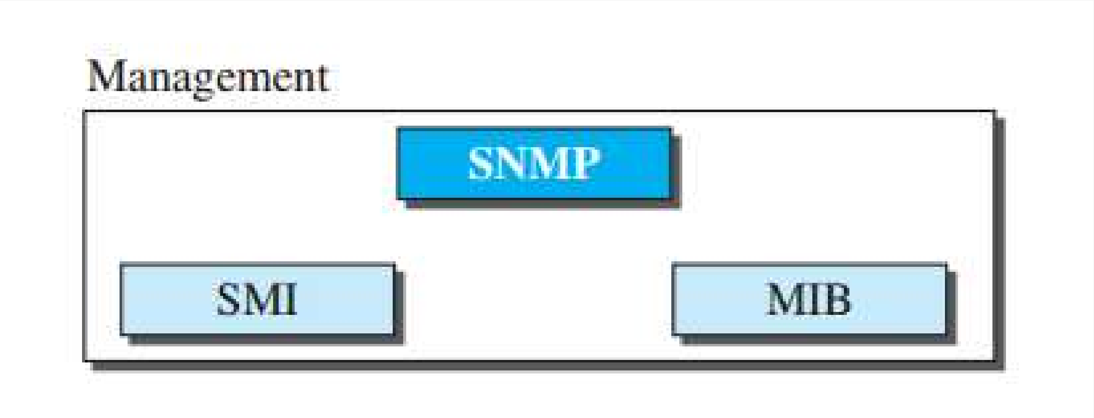
Managers and Agents

- Manager: runs the SNMP client program;
  Agent: runs the SNMP server program.
- Management – interaction between manager and agent.
- Agent keeps its performance info in a database;
  Manager has access to the values in the database.

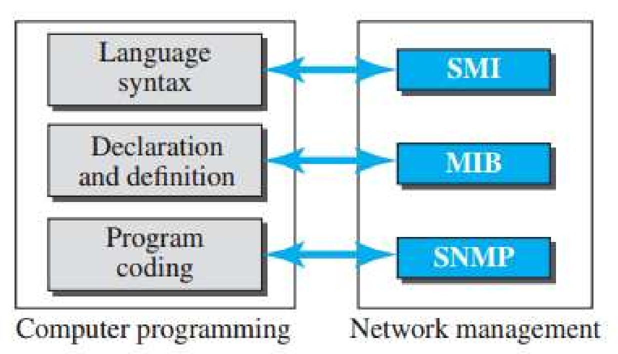
- Management with SNMP based on 3 basic ideas:
  1. Manager checks Agent: request info that reflects behaviour of agent.
  2. Manager forces Agent: perform a task by resetting values in database.
  3. Agent warns Manager: send warning message (trap), if unusual situation.

Management Components

- Management is done through the cooperation of three protocols:
  - SNMP
  - **SMI – Structure of Management Information**
  - **MIB – Management Information Base**



Roles of SNMP, SMI and MIB



SNMP

- Defines the format of packets exchanged between a manager and an agent.
- Reads and changes the status of objects in SNMP packets.

(Program coding – manipulating values of objects declared by MIB acc. to SMI's rules)

SMI

- Defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values.

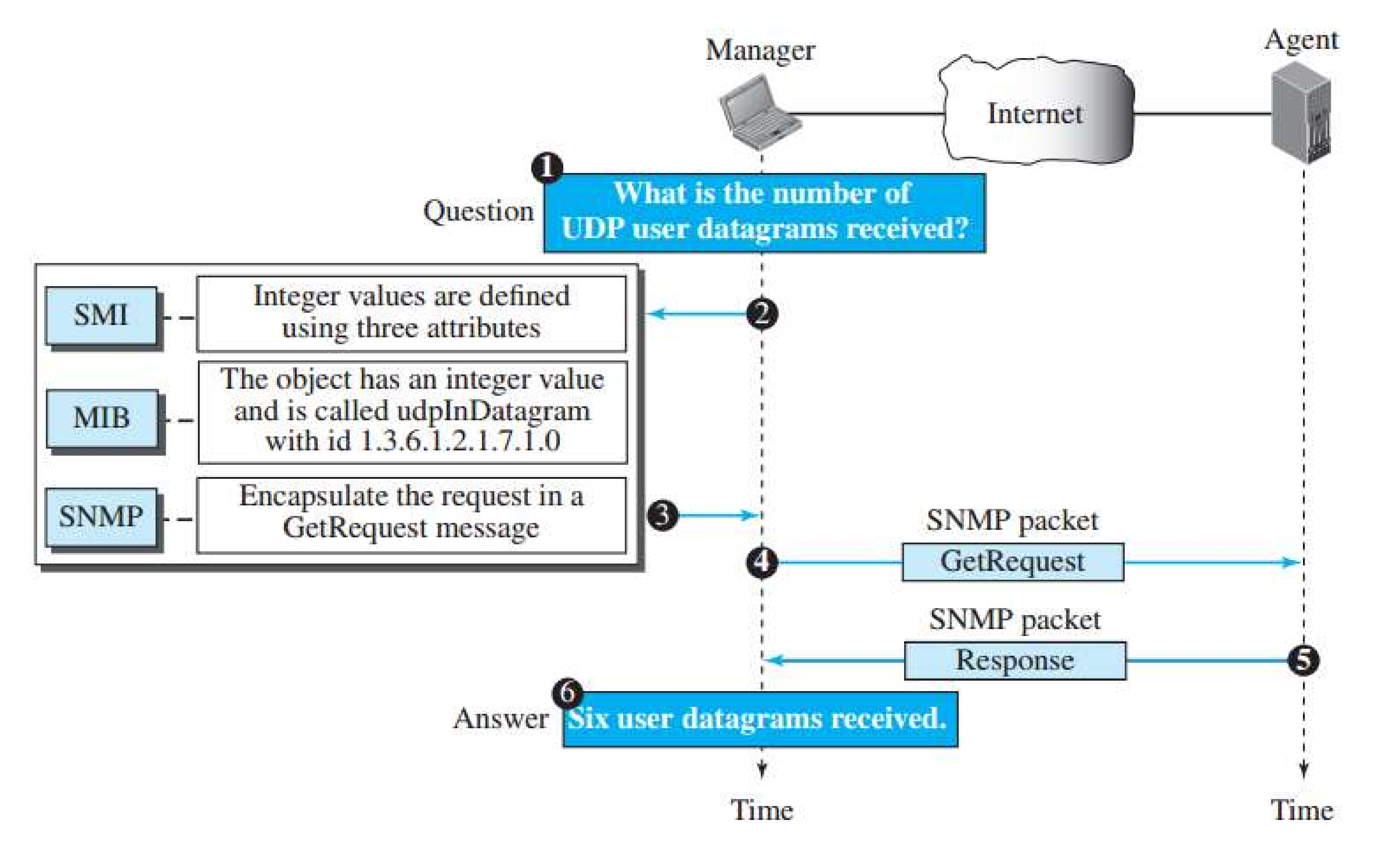
(Datatypes and Syntax)

MIB

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

(Metadata in a database – declaration and definition)

Management Overview

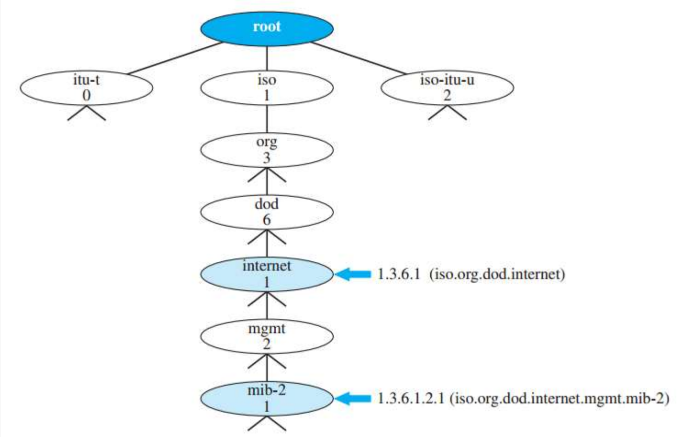

Manager
Agent
Internet
1
Question
What is the number of UDP user datagrams received?
SMI
Integer values are defined using three attributes
2
MIB
The object has an integer value and is called udpInDatagram with id 1.3.6.1.2.1.7.1.0
SNMP
Encapsulate the request in a GetRequest message
3
SNMP packet
4
GetRequest
SNMP packet
Response
5
6
Answer
Six user datagrams received.
Time
Time

- MIB finds an object that holds the number of UDP user datagrams received.

- SMI encodes the name of the object.

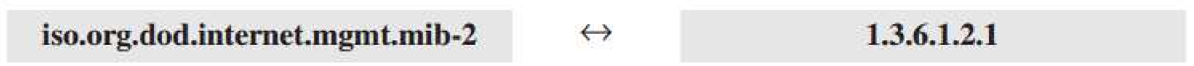
- SNMP creates a GetRequest message and encapsulates the encoded message.

SMIv2 (SMI, version 2)



root
itu-t 0
iso 1
iso-itu-u 2
org 3
dod 6
internet 1
1.3.6.1 (iso.org.dod.internet)
mgmt 2
mib-2 1
1.3.6.1.2.1 (iso.org.dod.internet.mgmt.mib-2)

- Guideline for SNMP
- Emphasises three attributes to handle an object:
  - Name
    - Unique name – uses an object identifier (hierarchical)
    - Integer-dot representation

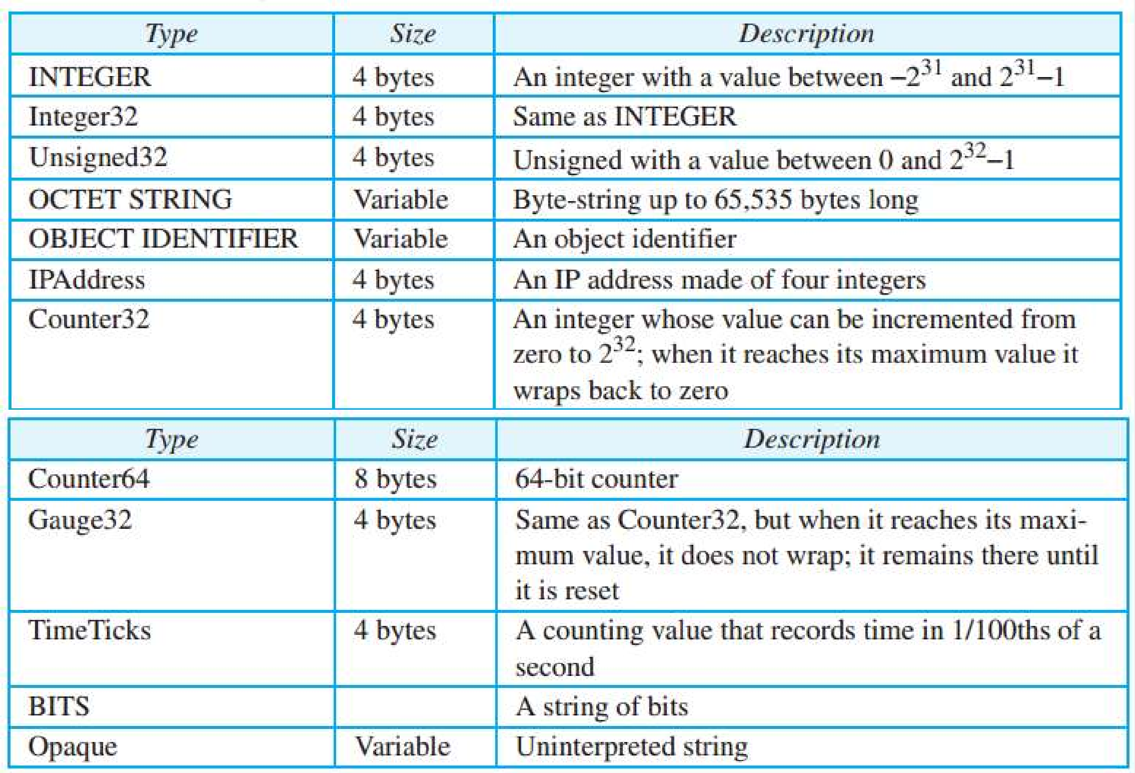


iso.org.dod.internet.mgmt.mib-2 ↔ 1.3.6.1.2.1

  - Data type
    - To define, uses Abstract Syntax Notation One (ASN.1)
    - Simple type – atomic data type
    - Structured type:
      - Sequence – combination of simple types, not necessarily of the same type – list in Python
      - Sequence of – combination of simple types or sequences, all of the same type – array in C
  - Encoding method

SMI data types - examples

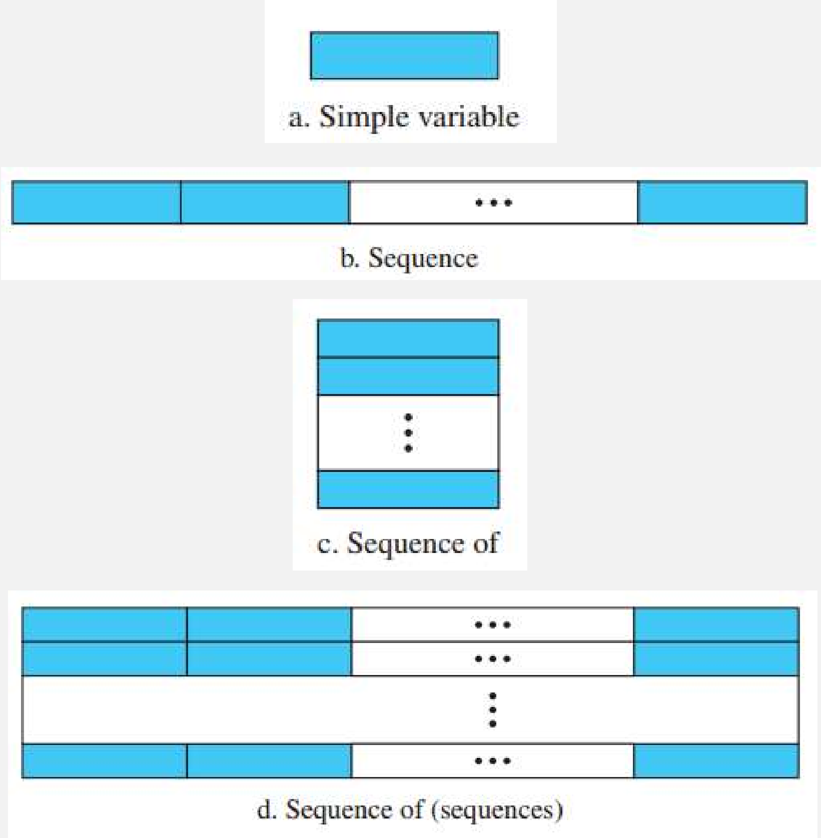
- Simple types



Type	*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 zero to 2^{32}; when it reaches its maximum value it wraps back to zero

Type	*Size*	*Description*
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 1/100ths of a second
BITS		A string of bits
Opaque	Variable	Uninterpreted string

- Structured types



a. Simple variable

b. Sequence

c. Sequence of

d. Sequence of (sequences)

SMI encoding method

- SMI uses a standard called Basic Encoding Rules (BER) to encode data.
- BER specifies that each piece of data be encoded in triplet format: *tag*, *length*, and *value* (TLV).
  - **Tag:** defines the type of data
  - **Length:** specifies length
    - If 1 byte, MSB = 0 and other 7 bits specify length of the data.
    - If more than 1 byte,
      - MSB of first byte = 1
      - Other 7 bits of first byte specify the number of bytes needed to specify the length of the data, say, n
      - The next n bytes specify the length of the data.
  - **Value:** codes the value of the data according to the BER rules

Encoding format

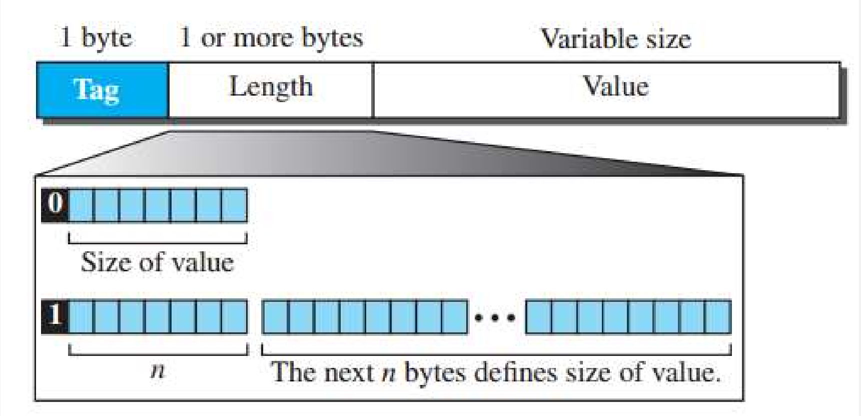
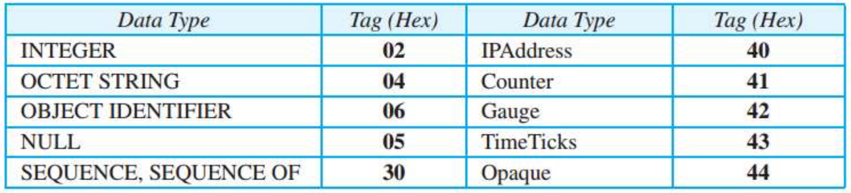


Table 27.2 *Codes for data types*



Data Type	*Tag (Hex)*	*Data Type*	*Tag (Hex)*
INTEGER	**02**	IPAddress	**40**
OCTET STRING	**04**	Counter	**41**
OBJECT IDENTIFIER	**06**	Gauge	**42**
NULL	**05**	TimeTicks	**43**
SEQUENCE, SEQUENCE OF	**30**	Opaque	**44**

SMI encoding method - examples

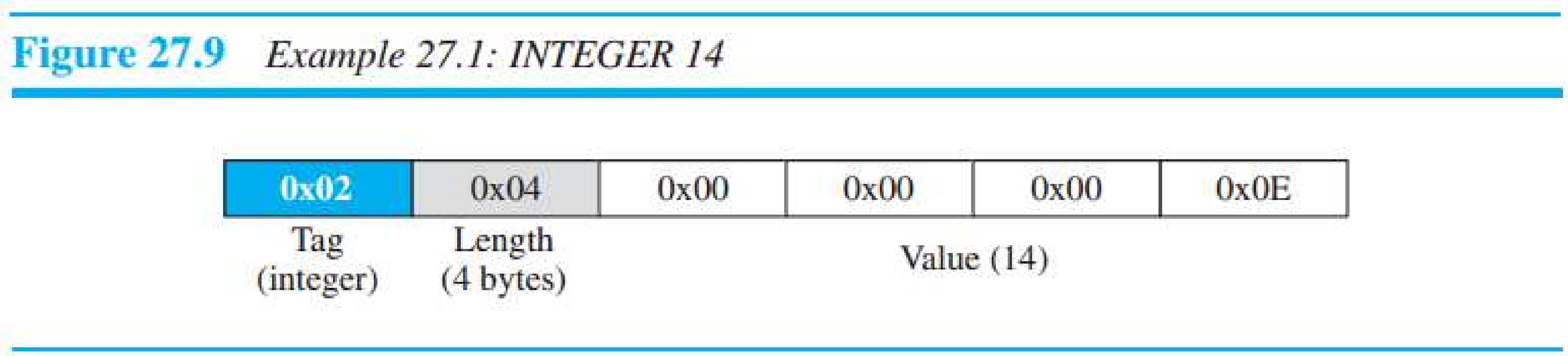


Figure 27.9 *Example 27.1: INTEGER 14*

0x02	0x04	0x00	0x00	0x00	0x0E
Tag (integer)	Length (4 bytes)	Value (14)			

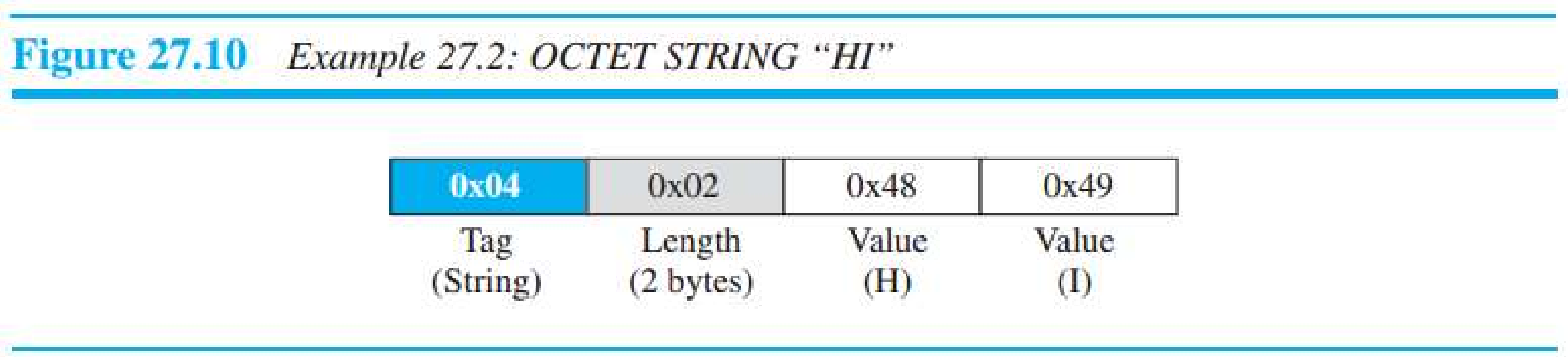


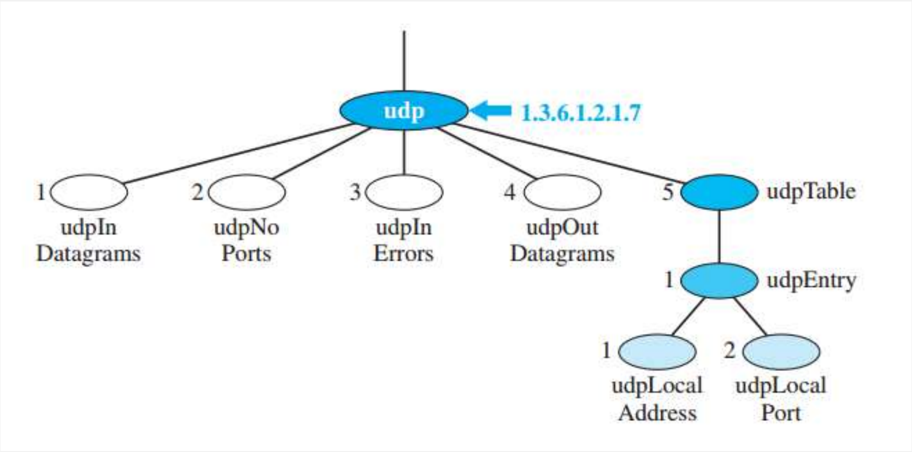
Figure 27.10 *Example 27.2: OCTET STRING "HI"*

0x04	0x02	0x48	0x49
Tag (String)	Length (2 bytes)	Value (H)	Value (I)

MIB2 (MIB, version 2)

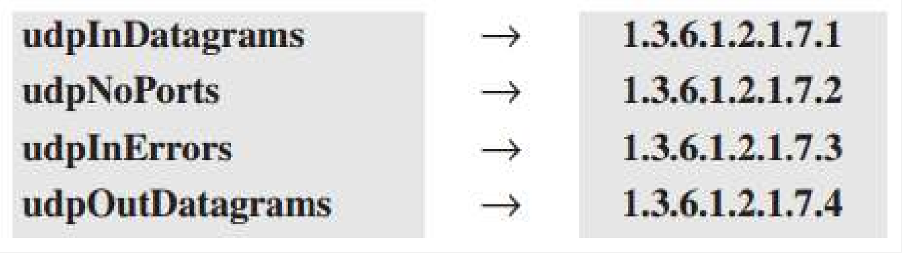
- Each agent has its own MIB2 (collection of objects that the manager can manage)
- The objects in MIB2 are categorized under several groups:
  - system (sys) - defines general information about the node/system
  - interface (if) - defines information about all the interfaces of the node
  - address translation (at) - defines the information about the ARP table
  - ip - defines information related to IP
  - icmp - defines information related to ICMP
  - tcp - defines general information related to TCP
  - udp - defines general information related to UDP
  - egp - related to the operation of EGP
  - transmission (trans) - related to the specific method of transmission (future use)
  - snmp - defines general information related to SNMP itself

Accessing MIB Variables



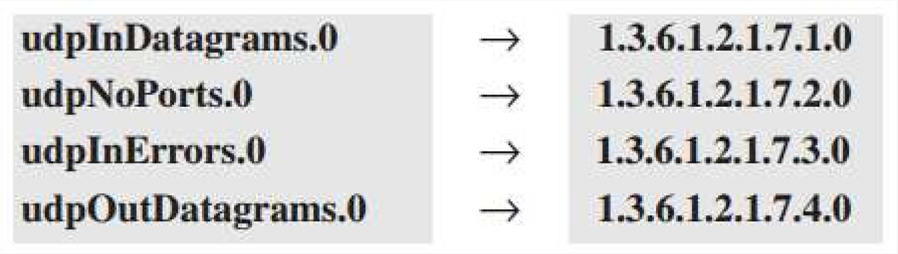
- Simple variables

ID of the group, followed by ID of the variable



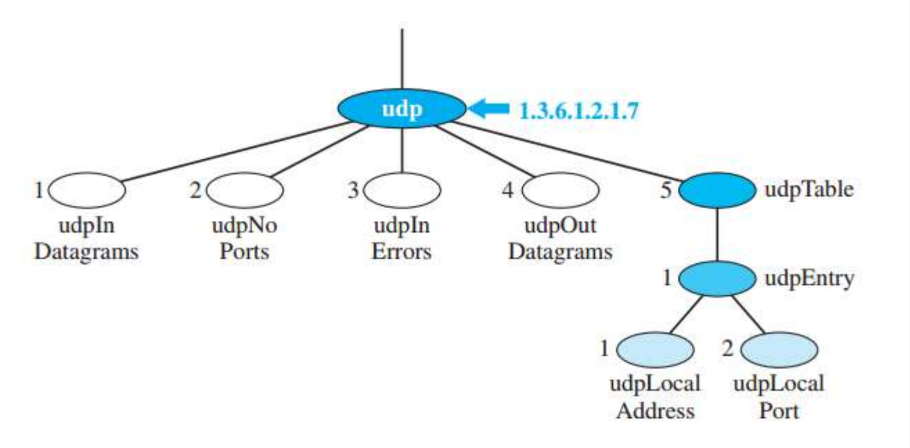
udpInDatagrams	→	1.3.6.1.2.1.7.1
udpNoPorts	→	1.3.6.1.2.1.7.2
udpInErrors	→	1.3.6.1.2.1.7.3
udpOutDatagrams	→	1.3.6.1.2.1.7.4

To access content of the variable, add an instance suffix (0 for a simple variable)



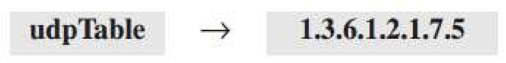
udpInDatagrams.0	→	1.3.6.1.2.1.7.1.0
udpNoPorts.0	→	1.3.6.1.2.1.7.2.0
udpInErrors.0	→	1.3.6.1.2.1.7.3.0
udpOutDatagrams.0	→	1.3.6.1.2.1.7.4.0

Accessing MIB Variables

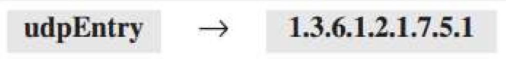


- Tables

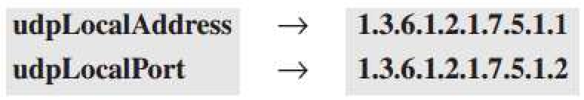
  Use the table ID. If the entry is not a leaf in the tree structure, define each entity in the entry.



udpTable → 1.3.6.1.2.1.7.5

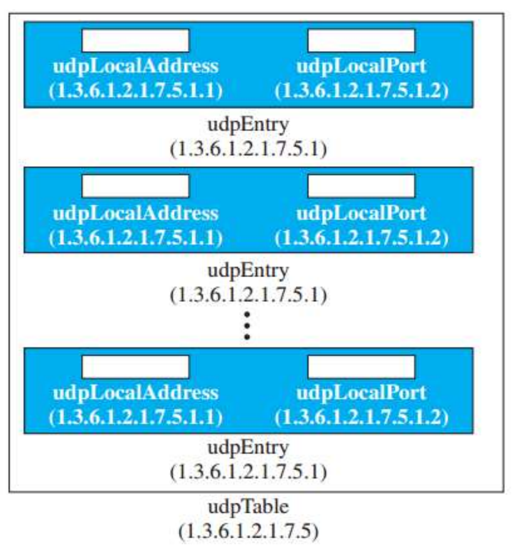


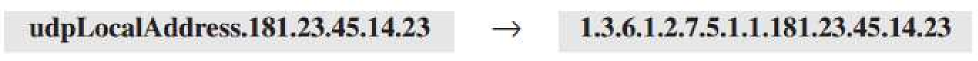
udpEntry → 1.3.6.1.2.1.7.5.1



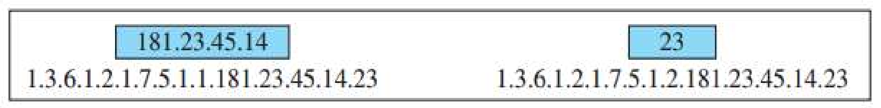
udpLocalAddress → 1.3.6.1.2.1.7.5.1.1
udpLocalPort → 1.3.6.1.2.1.7.5.1.2

  The indices are based on the value of one or more fields in the entries. In our example, the udpTable is indexed based on both the local address and the local port number.



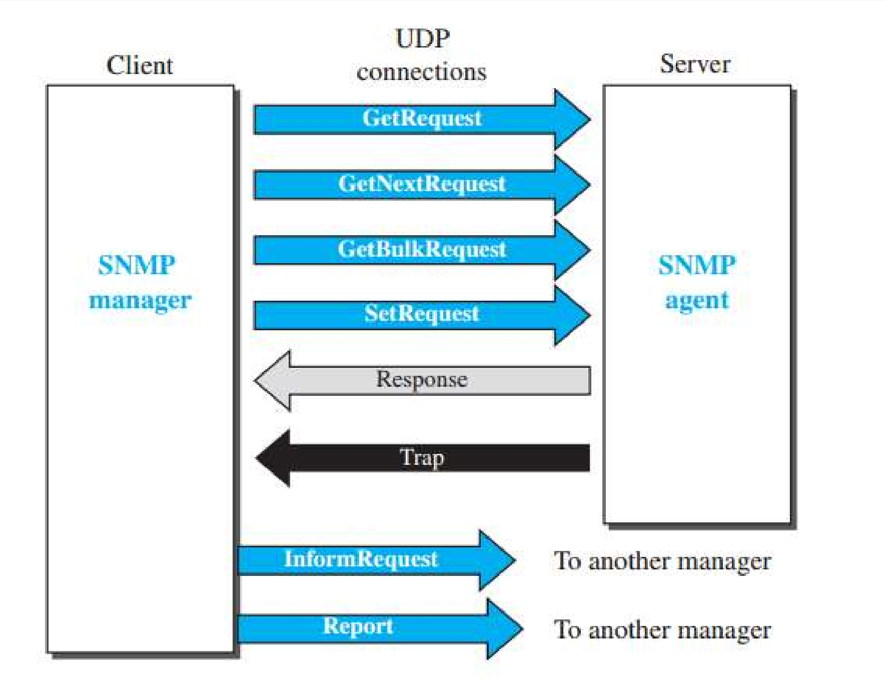


udpLocalAddress.181.23.45.14.23 → 1.3.6.1.2.7.5.1.1.181.23.45.14.23



SNMPv3 - PDUs

- Defines 8 types of Protocol Data Units (PDUs):
  - GetRequest
  - GetNextRequest
  - GetBulkReqest
  - SetRequest
  - Response
  - Trap
  - InformRequest
  - Report



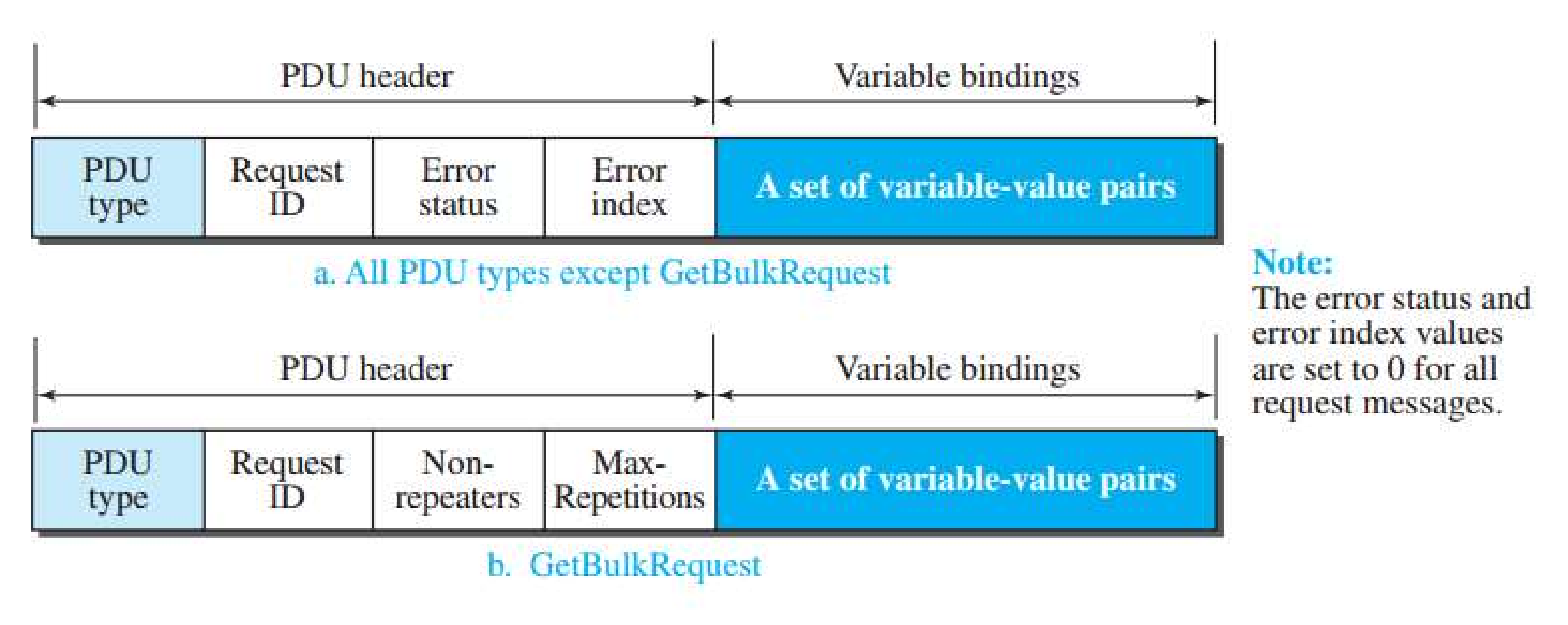
PDU Types

- GetRequest:
  - ➢ To retrieve the value of a variable or a set of variables.
- GetNextRequest:
  - ➢ To retrieve the value of a variable, where the retrieved value is the value of the object following the defined ObjectId in the PDU.
  - ➢ Mostly used to retrieve the values of the entries in a table.
- GetBulkRequest:
  - ➢ To retrieve a large amount of data.
  - ➢ Can be used instead of multiple GetRequest and GetNextRequest PDUs.
- SetRequest:
  - ➢ To set (store) a value in a variable.

PDU Types

- Response:
  - ➢ Contains the value(s) of the variable(s) requested by the manager.
- Trap:
  - ➢ To report an event.
  - ➢ Ex: If the agent is rebooted, it informs the manager and reports the time of rebooting.
- InformRequest:
  - ➢ To get the value of some variables from agents under the control of the remote manager.
- Report:
  - ➢ To report some types of errors between managers. Not yet in use.

SNMP PDU Format



PDU type	Request ID	Error status	Error index	A set of variable-value pairs

a. All PDU types except GetBulkRequest

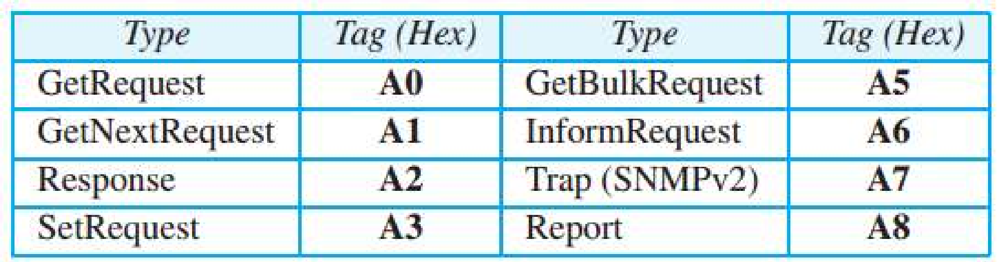
PDU type	Request ID	Non-repeaters	Max-Repetitions	A set of variable-value pairs

b. GetBulkRequest

Note:
The error status and error index values are set to 0 for all request messages.

SNMP PDU Format - Fields

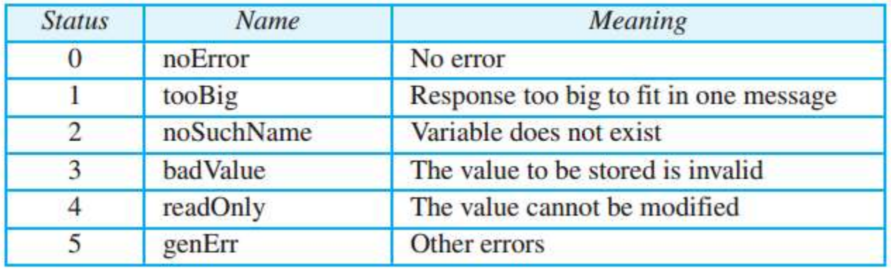
- *PDU type*: defines the type of the PDU.



Type	*Tag (Hex)*	*Type*	*Tag (Hex)*
GetRequest	**A0**	GetBulkRequest	**A5**
GetNextRequest	**A1**	InformRequest	**A6**
Response	**A2**	Trap (SNMPv2)	**A7**
SetRequest	**A3**	Report	**A8**

- *Request ID*: a sequence number used by the manager in a request PDU and repeated by the agent in a response. Used to match a request to a response.
- *Error status:* an integer that is used only in response PDUs to show the types of errors reported by the agent.

Table 27.4 *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

SNMP PDU Format - Fields

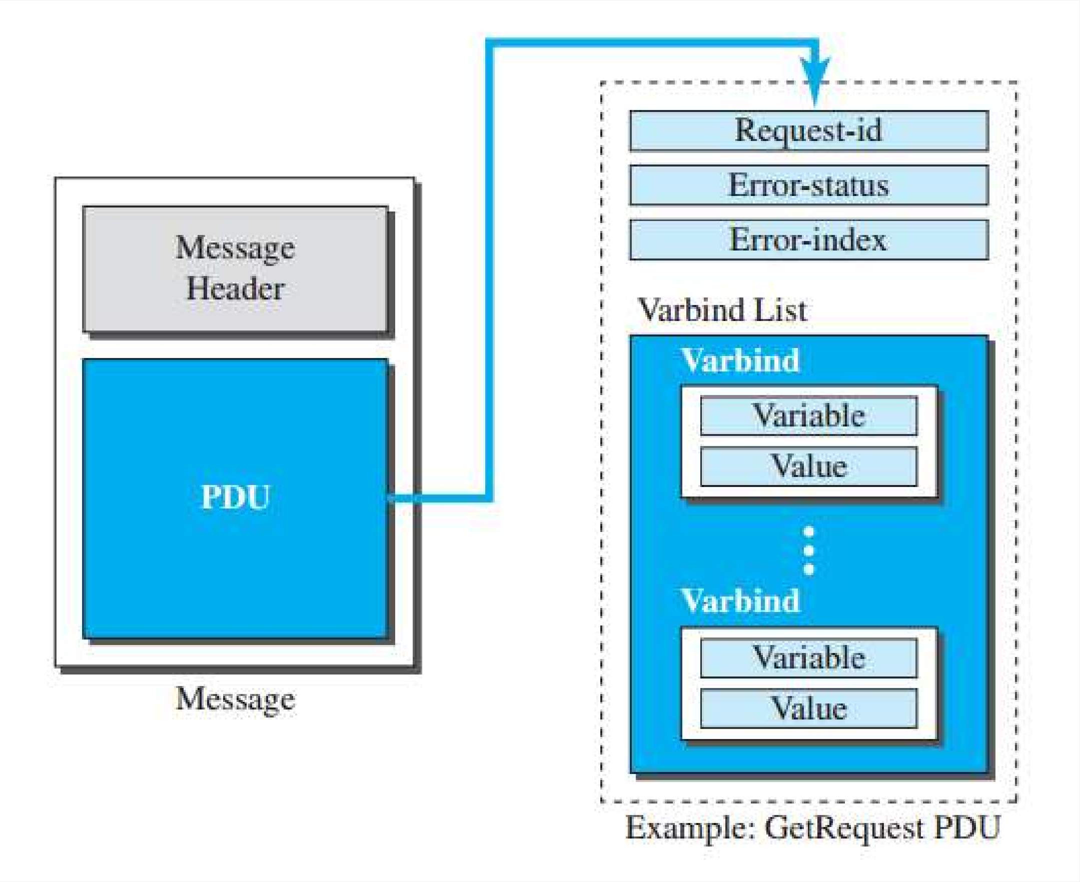
- *Non repeaters:* used only in a GetBulkRequest PDU. Defines the number of non-repeating (regular objects) at the start of the variable-value list.

- *Error index:* an offset that tells the manager which variable caused the error.

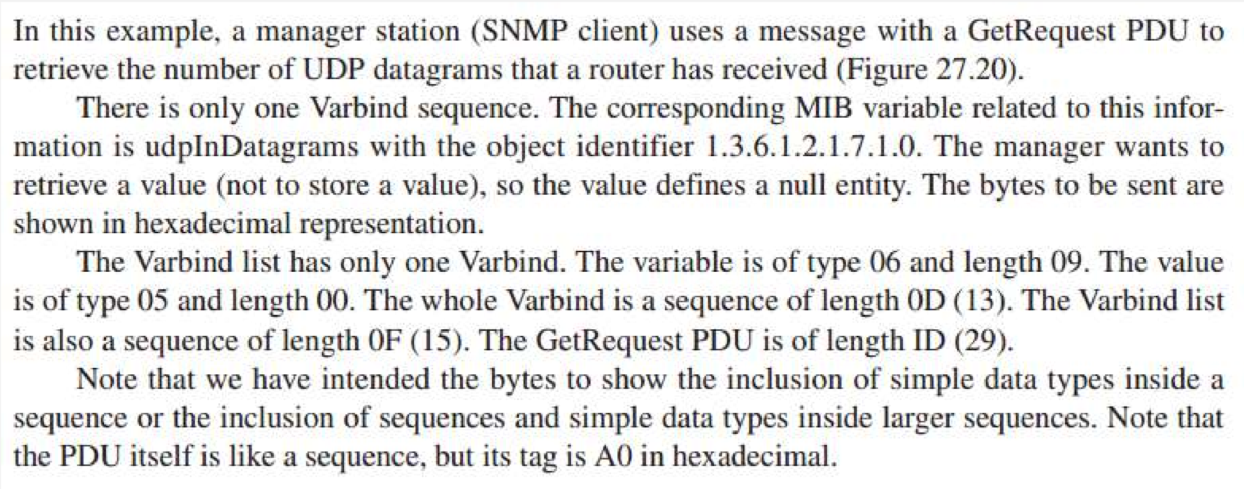
- *Max-repetitions*: used only in a GetBulkRequest PDU. Defines the maximum number of iterations in the table to read all repeating objects.

- *Variable-value pair list:* a set of variables with the corresponding values the manager wants to retrieve or set. The values are null in request PDUs.

SNMP Message



SNMP Message Format - example



In this example, a manager station (SNMP client) uses a message with a GetRequest PDU to retrieve the number of UDP datagrams that a router has received (Figure 27.20).

There is only one Varbind sequence. 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. The bytes to be sent are shown in hexadecimal representation.

The Varbind list has only one Varbind. 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).

Note that we have intended the bytes to show the inclusion of simple data types inside a sequence or the inclusion of sequences and simple data types inside larger sequences. Note that the PDU itself is like a sequence, but its tag is A0 in hexadecimal.

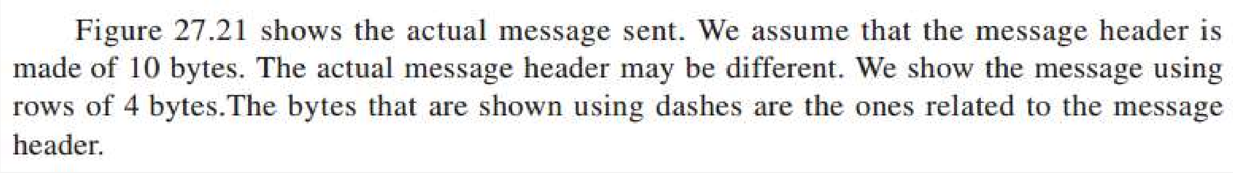


Figure 27.21 shows the actual message sent. We assume that the message header is made of 10 bytes. The actual message header may be different. We show the message using rows of 4 bytes.The bytes that are shown using dashes are the ones related to the message header.

Figure 27.20 *Example 27.5*

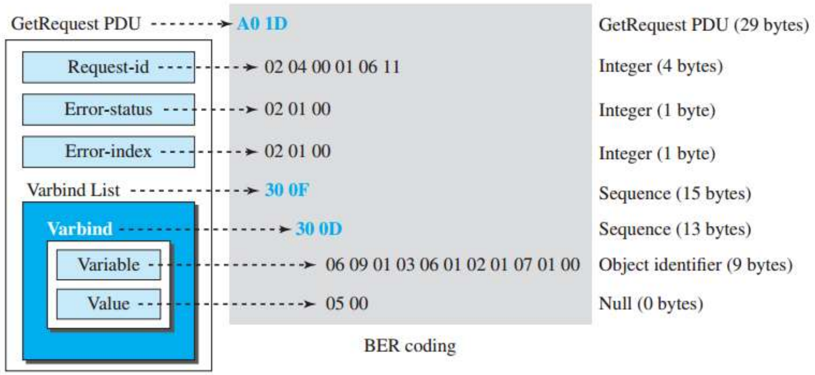
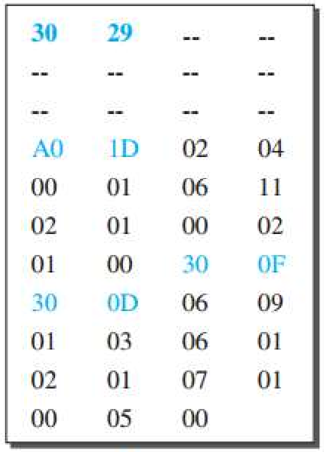


Figure 27.21 *Actual message sent for Example 27.5*



30	29	--	--
--	--	--	--
--	--	--	--
A0	1D	02	04
00	01	06	11
02	01	00	02
01	00	30	0F
30	0D	06	09
01	03	06	01
02	01	07	01
00	05	00	

Message

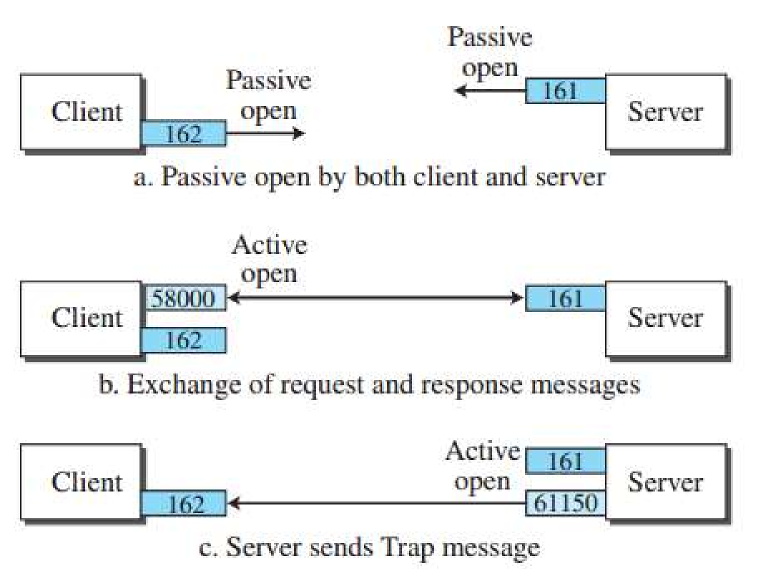
Note:
The byte values are in hexadecimal.

UDP Ports

- SNMP uses the UDP services on two well-known ports:
  - Port 161 – server (agent)
  - Port 162 – client (manager)

Both the client and the server use well-known ports.

Both the client and the server are running infinitely.



a. Passive open by both client and server

b. Exchange of request and response messages

c. Server sends Trap message

Security in SNMPv3

- Allows a manager to choose one or more levels of security when accessing an agent.
- Different aspects of security can be configured by the manager to allow message authentication, confidentiality, and integrity.
- Allows remote configuration of security aspects.

Summary

- Managers and Agents
- Management Components
- Management Overview
- SMI
- MIB
- SNMP

Test Your Understanding

- List the three protocols required for network management.
- _______ defines the general rules for declaring and defining objects and values.
- _______ creates a collection of named objects, their types and their relationships with each other.
- There are _______ types of PDUs in SNMPv3.
- _________ is the only PDU type with a different PDU format.

References

- *Data Communication and Networking*, Behrouz A. Forouzan (Chapter 27.2)

THANK YOU!