

# Network Management

## Lecture 5

### SNMPv1 Network

### Management:

### Communication and Functional Models

---

# Objectives

- Communication model: Administrative and messages
- Administrative structure
  - Community-based model
  - Access policy
  - MIB view
- Message PDU
- SNMP protocol specifications
- SNMP operations
- SNMP MIB
- SNMP functional model

# SNMP Architecture

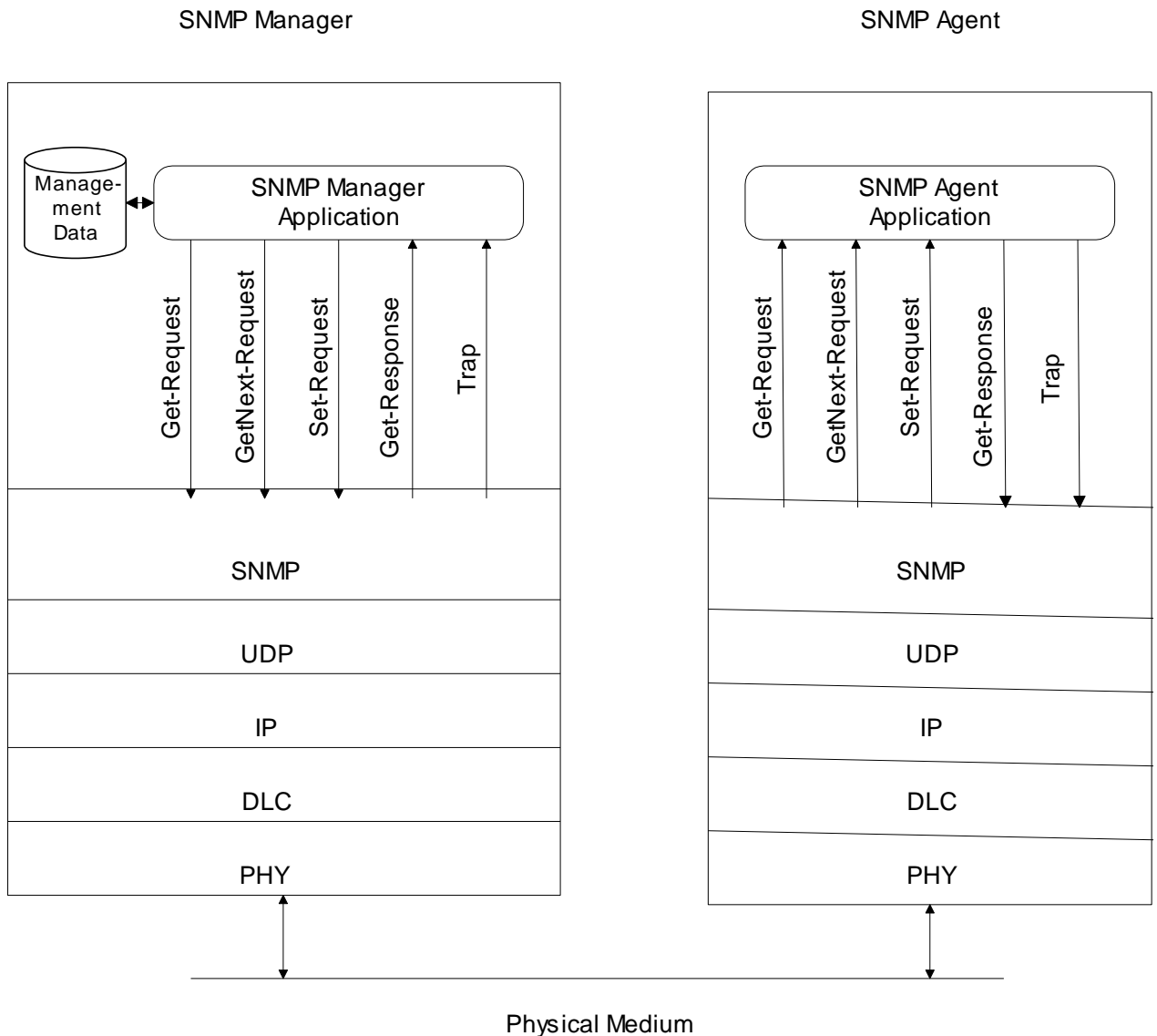


Figure 4.9 SNMP Network Management Architecture

## Notes

- Truly *simple* network management protocol
- Five messages, 3 from manager and 2 from agent

---

# SNMP Messages

- Get-Request
- Get-Next-Request
- Set-Request
- Get-Response
- Trap
  - Generic trap
  - Specific trap
  - Time stamp

---

## Notes

- Generic trap
  - coldStart
  - warmStart
  - linkDown
  - linkUp
  - authenticationfailure
  - egpNeighborLoss
  - enterpriseSpecific
- Specific trap
  - For special measurements such as statistics
- Time stamp: Time since last initialization

---

# Administrative Model

- Based on community profile and policy
- SNMP Entities:
  - SNMP application entities
    - Reside in management stations and network elements
    - Manager and agent
  - SNMP protocol entities
    - Communication processes (PDU handlers)
    - Peer processes that support application entities

---

## Notes

---

# SNMP Community

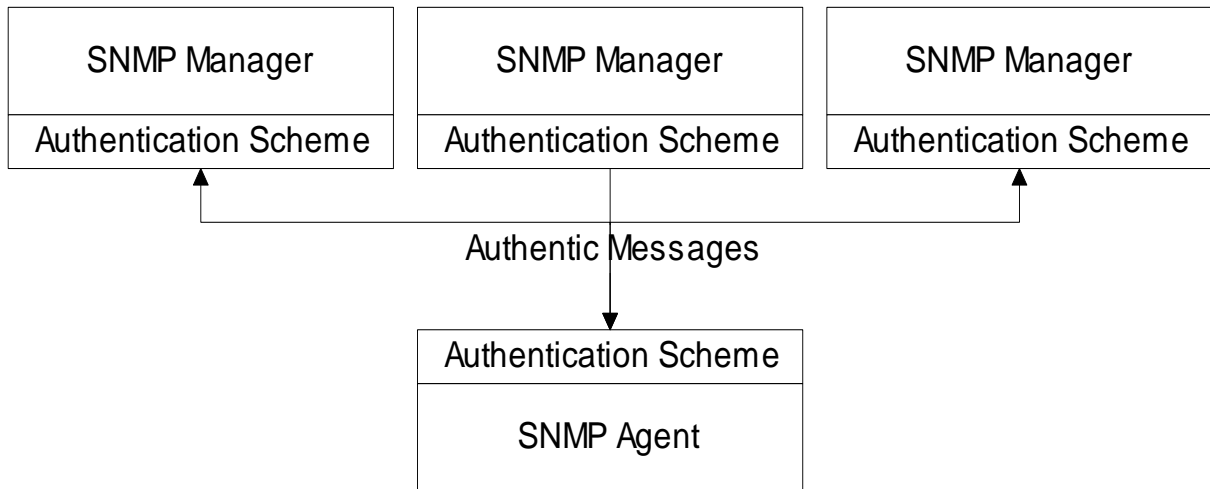


Figure 5.1 SNMP Community

---

## Notes

- Security in SNMPv1 is community based
- Authentication scheme in manager and agent
- Community: Pairing of two application entities
- Community name: String of octets
- Two applications in the same community communicate with each other
- Application could have multiple community names
- Communication is not secured in SNMPv1 - no encryption

---

# Community Profile

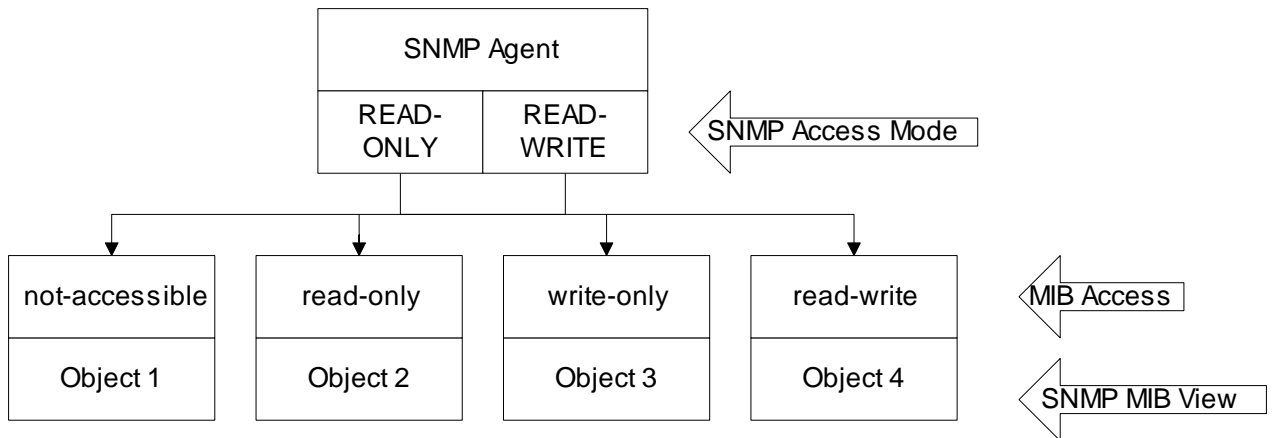


Figure 5.2 SNMP Community Profile

---

## Notes

- MIB view
  - An agent is programmed to view only a subset of managed objects of a network element
- Access mode
  - Each community name is assigned an access mode: read-only and read-write
- Community profile: MIB view + SNMP access mode
- Operations on an object determined by community profile and the access mode of the object
- Total of four access privileges
- Some objects, such as table and table entry are non-accessible

---

# Administrative Model

- Administrative model is SNMP access policy
- SNMP community paired with SNMP community profile is SNMP access policy

---

## Notes

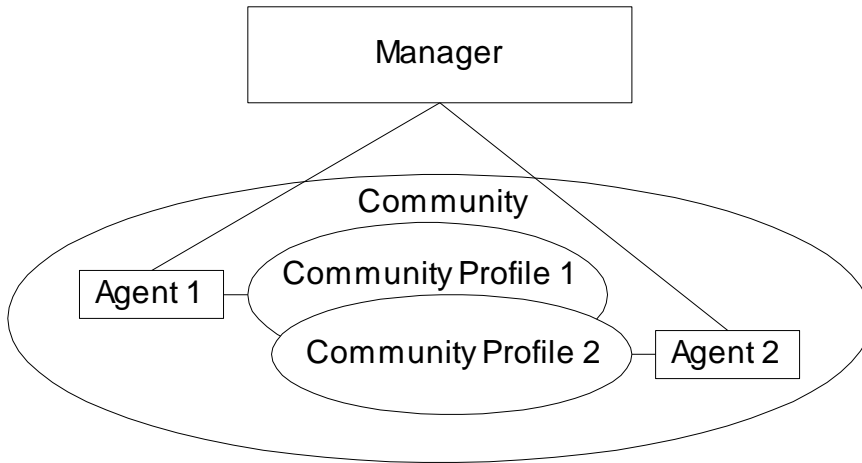
Parameters:

- Community / communities
- Agent / Agents
- Manager / Managers



---

# Access Policy



---

## Notes

- Manager manages Community 1 and 2 network components via Agents 1 and 2
- Agent 1 has only view of Community Profile 1, e.g., Cisco components
- Agent 2 has only view of Community Profile 2, e.g., 3Com components
- Manager has total view of both Cisco and 3Com components

---

# Generalized Administrative Model

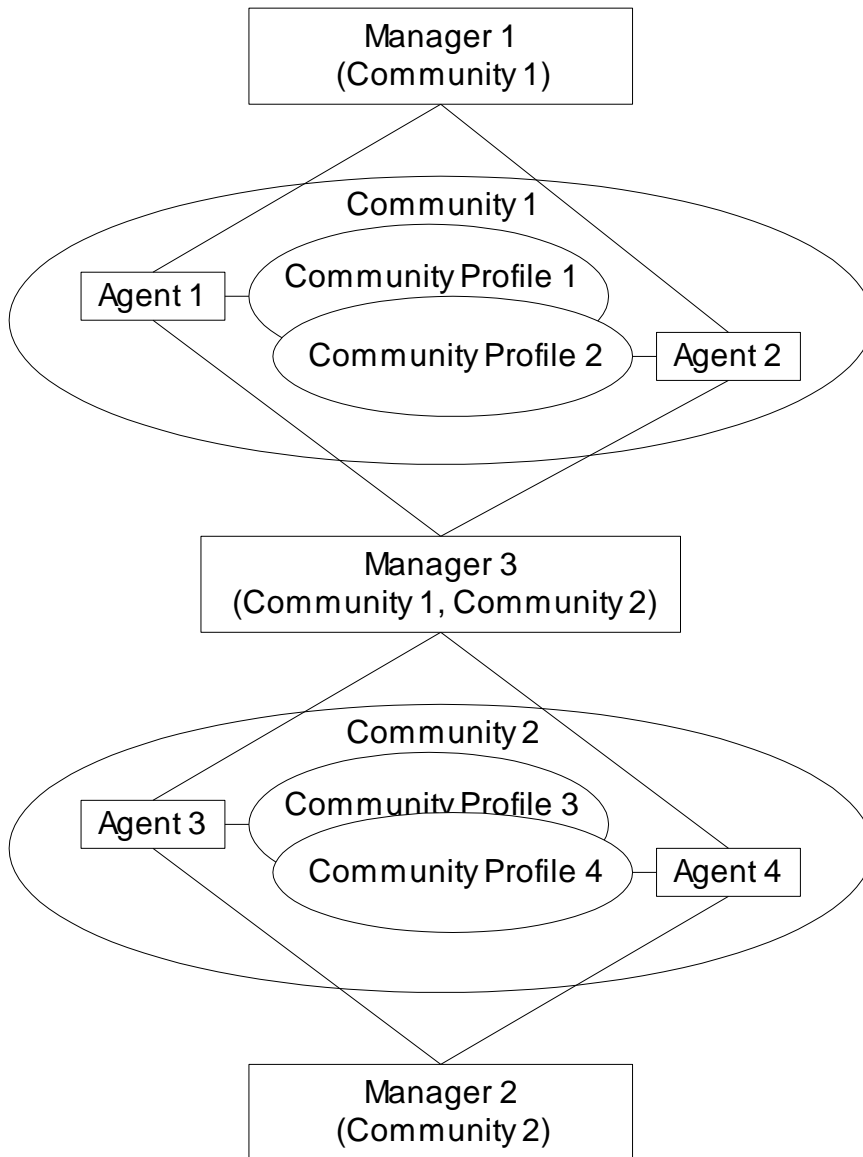


Figure 5.3 SNMP Access Policy

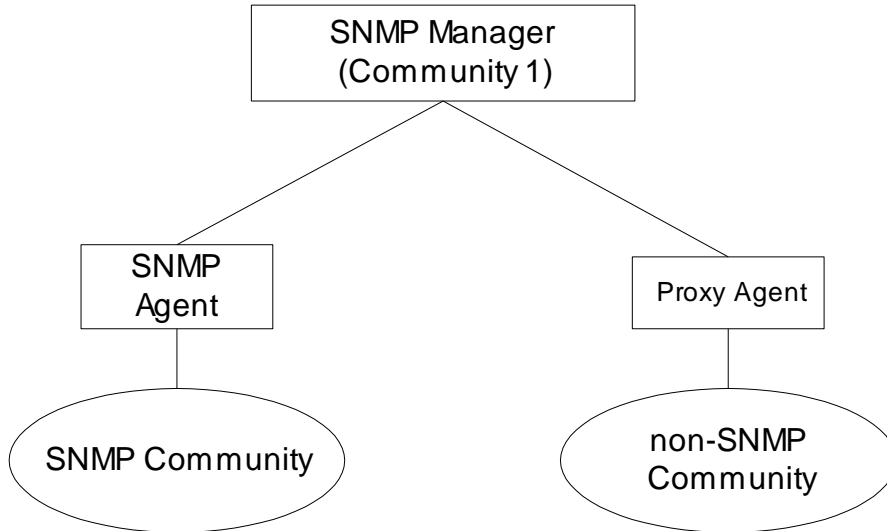
---

## Notes

- Manager 1 manages community 1, manager 2 community 2, and manager 3 (MoM) both communities 1 and 2

---

# Proxy Access Policy



**Figure 5.4 SNMP Proxy Access Policy**

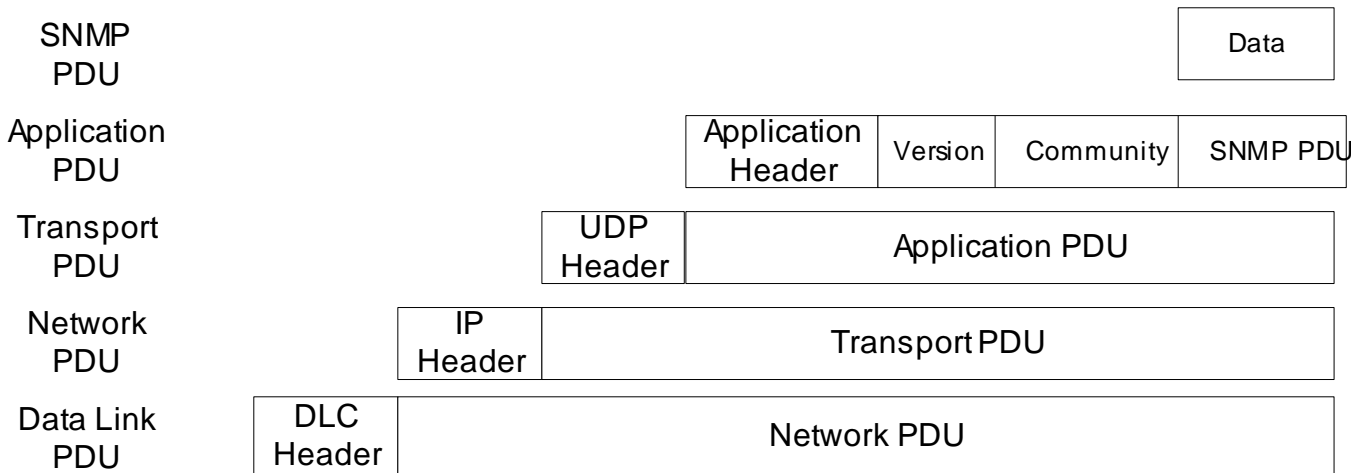
---

## Notes

- Proxy agent enables non-SNMP community elements to be managed by an SNMP manager.
- An SNMP MIB is created to handle the non-SNMP objects.

---

# Protocol Entities



**Figure 5.5 Encapsulated SNMP Message**

---

## Notes

- Protocol entities support application entities
- Communication between remote peer processes
- Message consists of:
  - Version identifier
  - Community name
  - Protocol Data Unit
- Message encapsulated and transmitted

---

# Get and Set PDU

PDU Type	RequestID	Error Status	Error Index	VarBind 1 name	VarBind 1 value	...	VarBind n name	VarBind n value
----------	-----------	--------------	-------------	----------------	-----------------	-----	----------------	-----------------

Figure 5.8 Get and Set Type PDUs

---

## Notes

- VarBindList: multiple instances of VarBind pairs

PDUs ::=

```
CHOICE {
    get-request           GetRequest-PDU,
    get-next-request     GetNextRequest-PDU,
    get-response         GetResponse-PDU,
    set-request          SetRequest-PDU,
    trap                 Trap-PDU
}
```

PDU Types: enumerated INTEGER

```
get-request           [0]
get-next-request     [1]
set-request          [2]
get-response         [3]
trap                 [4]
```

---

# Error in Response

```
ErrorStatus ::=
  INTEGER {
    noError(0)
    tooBig(1)
    noSuchName(2)
    bad value(3)
    readOnly(4)
    genErr(5)
  }
```

Error Index: No. of VarBind that the first error occurred

---

## Notes

---

# Trap PDU

PDU Type	Enterprise	Agent Address	Generic Trap Type	Specific Trap Type	Timestamp	VarBind 1 name	VarBind 1 value	...	VarBind n name	VarBind n value
----------	------------	---------------	-------------------	--------------------	-----------	----------------	-----------------	-----	----------------	-----------------

**Figure 5.8 Get and Set Type PDUs**

**Table 5.1 Generic Traps**

Generic Trap Type	Description (brief)
coldStart(0)	Sending protocol entity is reinitializing itself; agent's configuration or protocol entity implementation may be altered
warmStart(1)	Sending protocol entity is reinitializing itself; agent configuration or protocol entity implementation not altered
linkDown(2)	Failure of one of the communication links
linkUp(3)	One of the links has come up
authenticationFailure(4)	Authentication failure
egpNeighborLoss(5)	Loss of EGP neighbor
enterpriseSpecific(6)	Enterprise-specific trap

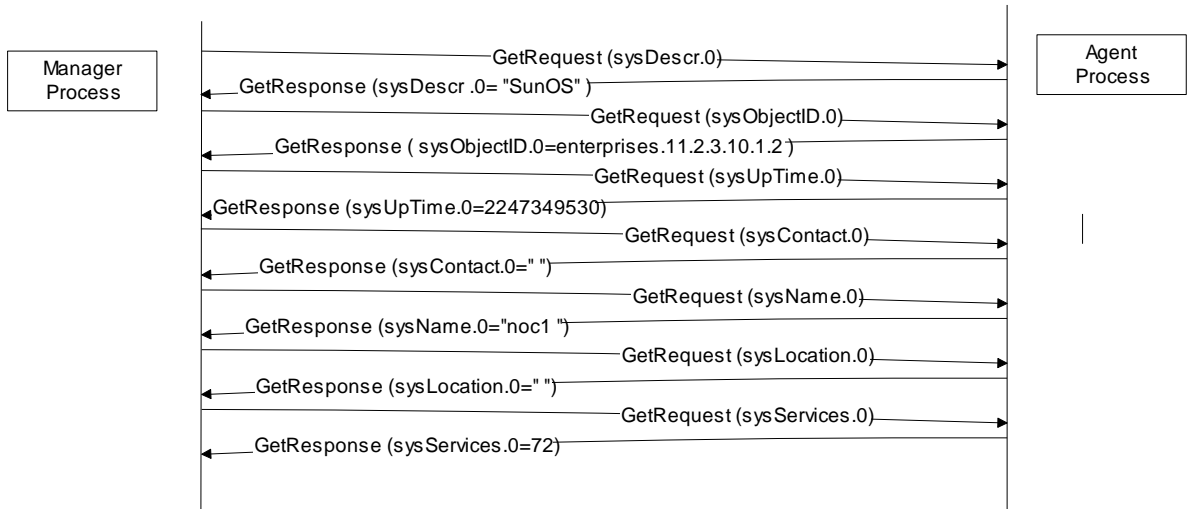
---

## Notes

- Enterprise and agent address pertain to the system generating the trap
- Seven generic traps specified by enumerated INTEGER
- Specific trap is a trap not covered by enterprise specific trap
- Timestamp indicates elapsed time since last re-initialization

---

# SNMP Operations



**Figure 5.10 Get-Request Operation for System Group**

---

## Notes



---

# MIB for Get-Next-Request

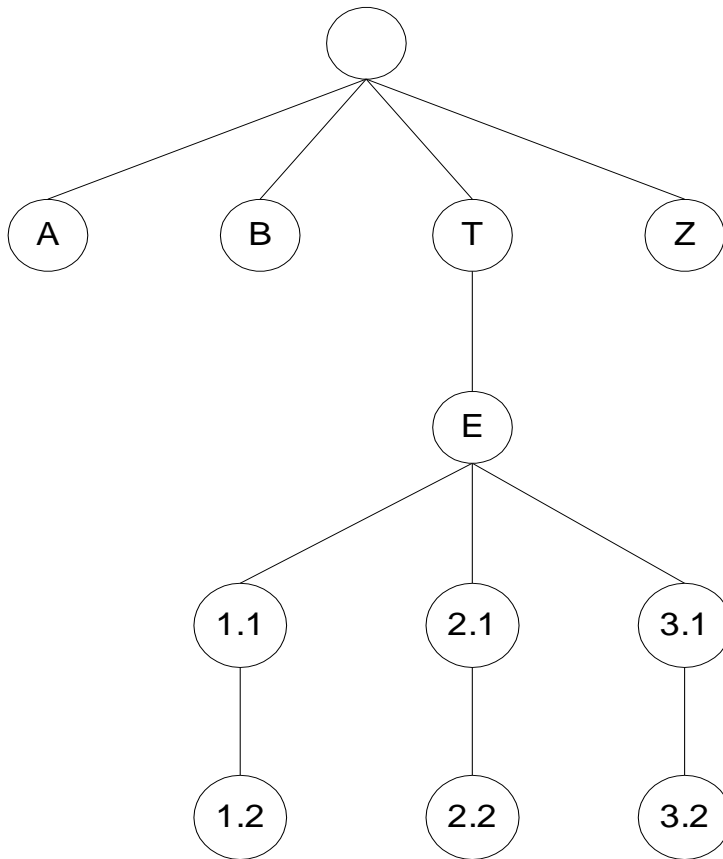


Figure 5.12 MIB for Operation Examples in Figures 5.13 and 5.15

---

## Notes

---

# Lexicographic Order

Table 5.2 Lexicographic-Order Number Example

Numerical Order	Lexicographic order
1	1
2	1118
3	115
9	126
15	15
22	2
34	22
115	250
126	2509
250	3
321	321
1118	34
2509	9

---

## Notes

- Procedure for ordering:
  - Start with leftmost digit as first position
  - Before increasing the order in the first position, select the lowest digit in the second position
  - Continue the process till the lowest digit in the last position is captured
  - Increase the order in the last position until all the digits in the last position are captured
  - Move back to the last but one position and repeat the process
  - Continue advancing to the first position until all the numbers are ordered
- Tree structure for the above process

---

# MIB Lexicographic Order

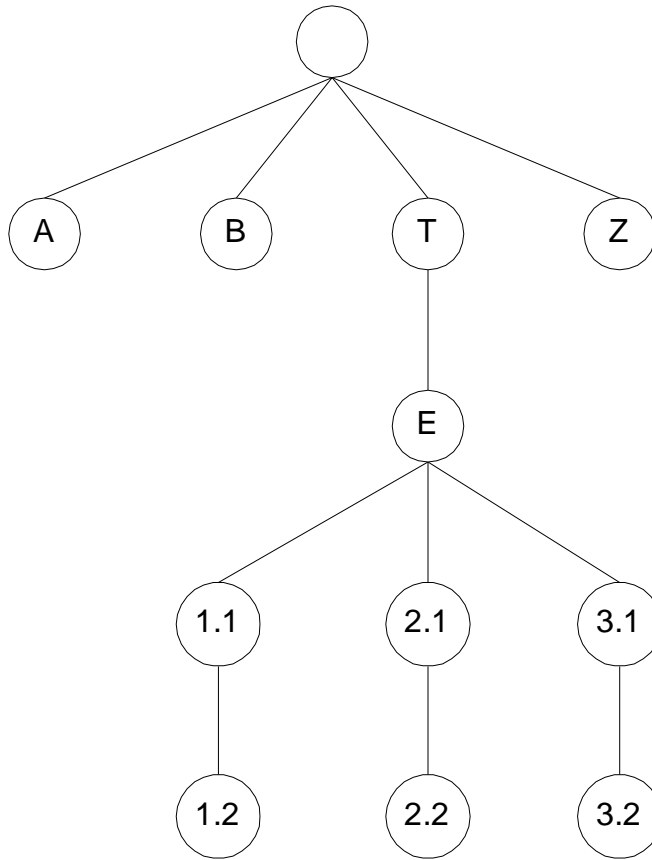


Figure 5.12 MIB for Operation Examples in Figures 5.13 and 5.15

---

## Notes

A	3.1
B	3.2
T	Z
E	
1.1	
1.2	
2.1	
2.2	

---

# A More Complex MIB Example

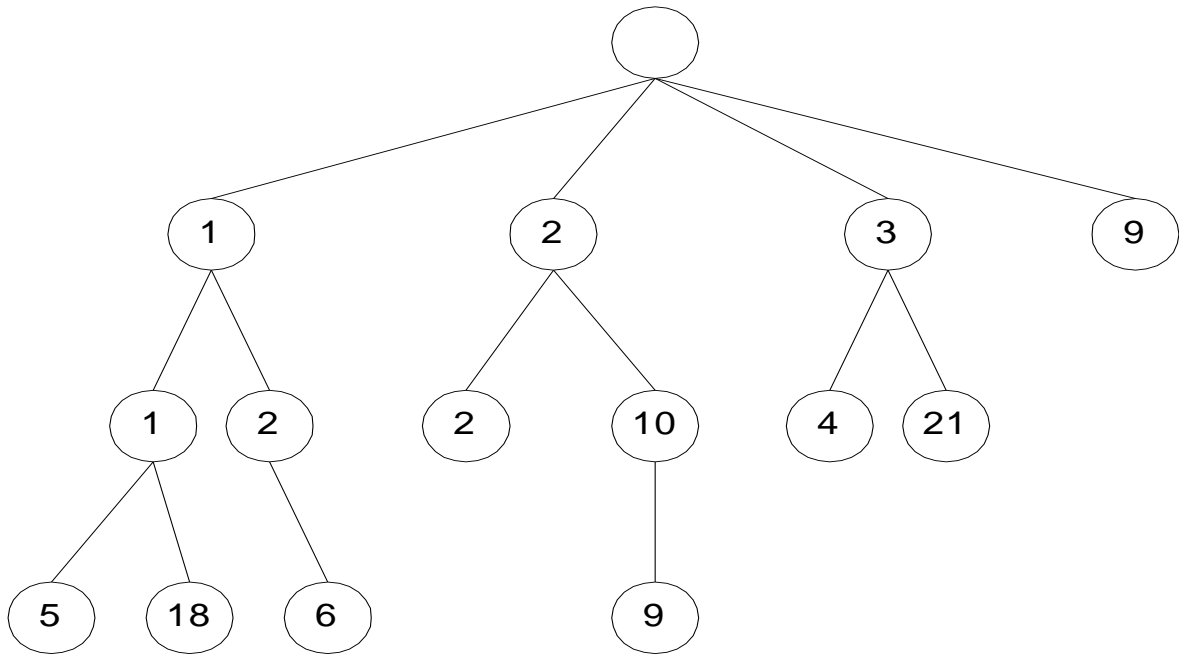


Figure 5.14 MIB Example for Lexicographic Ordering

---

## Notes

1
1.1
1.1.5
1.1.18
1.2
1.2.6
2
2.2
2.10
2.10.9
3
3.4
3.21
9

---

# Get-Next-Request Operation

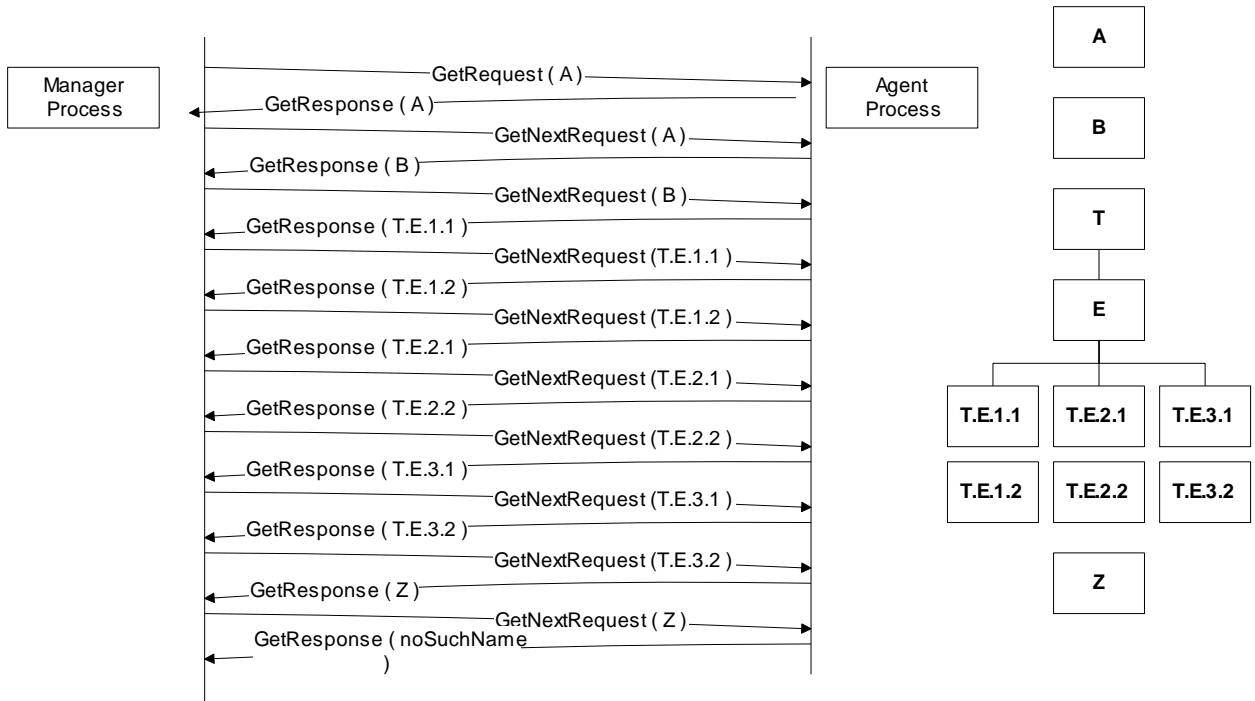


Figure 5.15 Get-Next-Request Operation for MIB in Figure 5.12

---

## Notes

# Get-Next-Request Operation

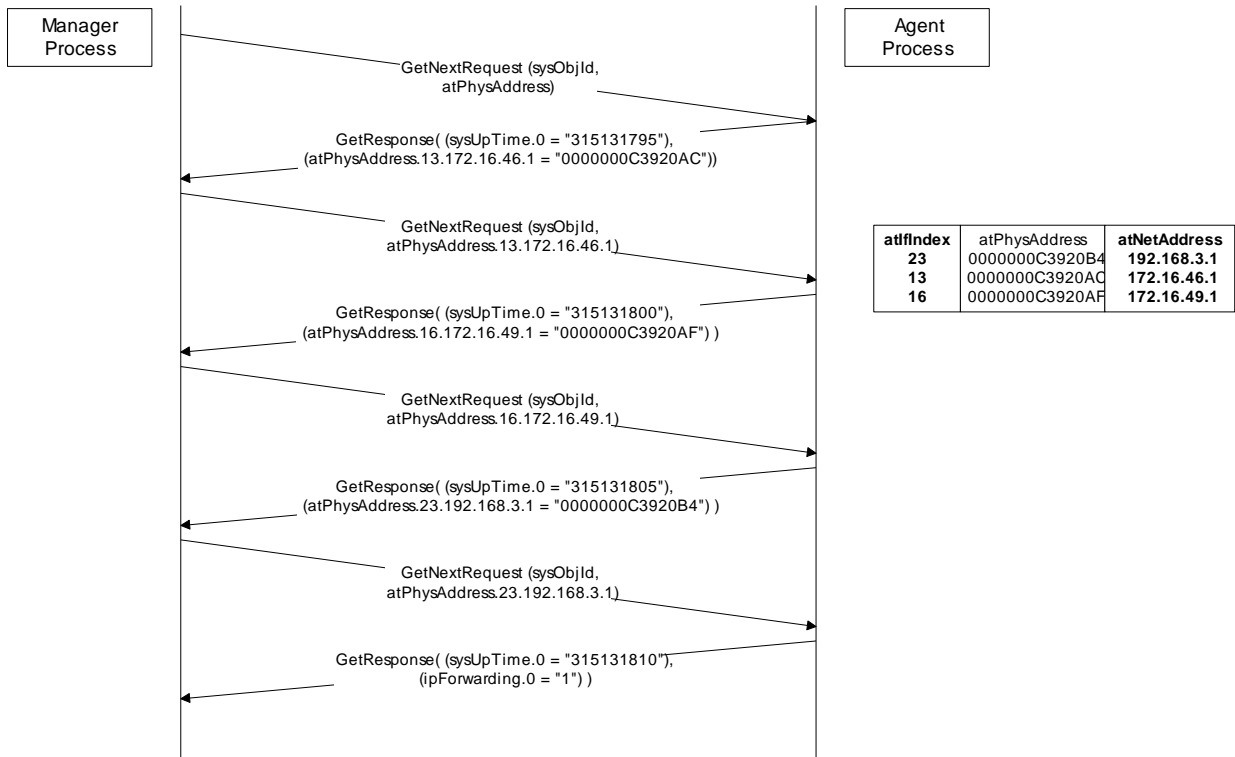


Figure 5.16 GetNextRequest Example with Indices

## Notes

---

# Sniffer Data

```
14:03:36.788270 noc3.btc.gatech.edu.164 >
noc1.btc.gatech.edu.snmp:
Community = public
GetRequest(111)
Request ID = 4
system.sysDescr.0
system.sysObjectID.0
system.sysUpTime.0
system.sysContact.0
system.sysName.0
system.sysLocation.0
system.sysServices.0
```

**Figure 5.17(a) Get-Request Message from Manager-to-Agent**

---

```
14:03:36.798269 noc1.btc.gatech.edu.snmp >
noc3.btc.gatech.edu.164:
Community = public
GetResponse(196)
Request ID = 4
system.sysDescr.0 = "SunOS noc1 5.5.1 Generic_103640-08
sun4u"
system.sysObjectID.0 = E:hp.2.3.10.1.2
system.sysUpTime.0 = 247396453
system.sysContact.0 = "Brandon Rhodes"
system.sysName.0 = "noc1"
system.sysLocation.0 = "BTC NM Lab"
system.sysServices.0 = 72
```

**Figure 5.17(b) Get-Response Message from Agent-to-Manager (After)**

---

# SNMP MIB

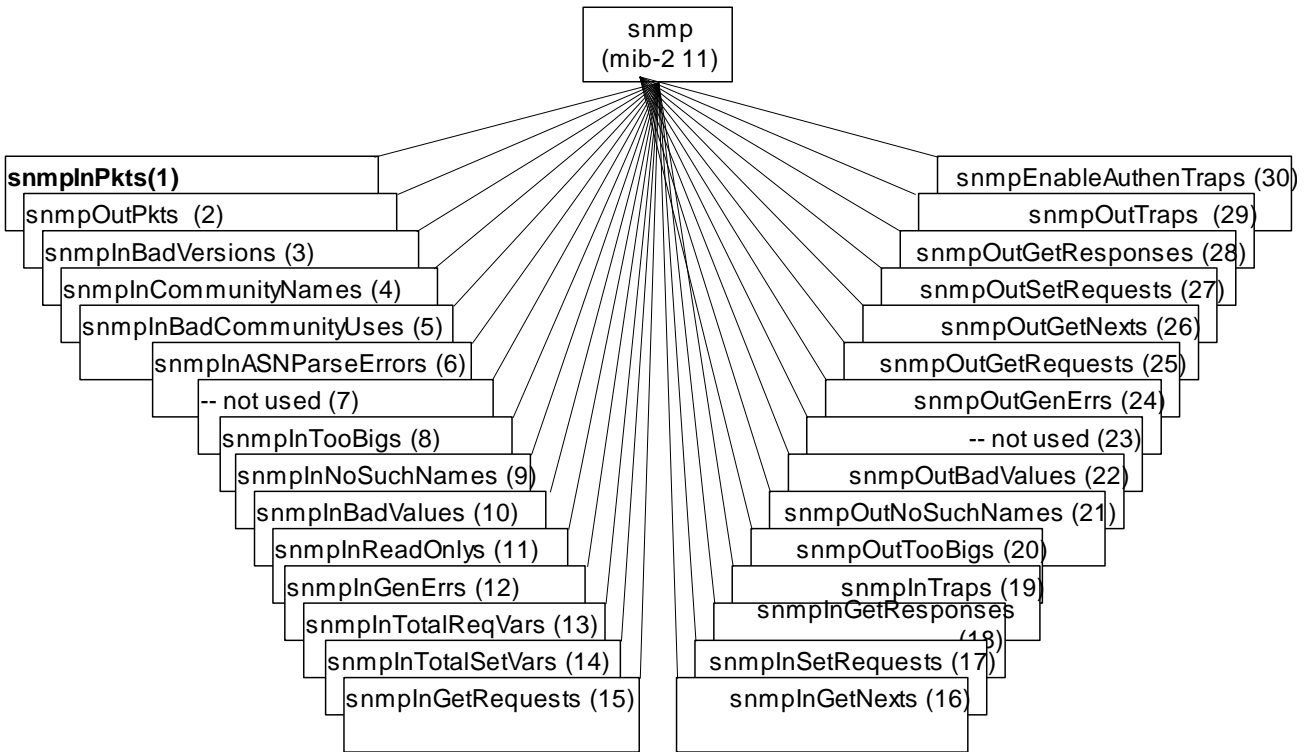


Figure 5.21 SNMP Group

---

## Notes

- SNMPv1 MIB has too many objects that are not used
- SNMPv2 obsoleted a large number of them

Note: Most of the MIB objects were not used and hence deprecated in SNMPv2