Overview

• We have covered the organization and information models of SNMPv1.
• Here we will address the SNMPv1 communication and functional models
• SNMPv1 does not formally define a functional model
  – What was the functional model?
    • Deals with the user oriented requirements: (configuration, fault, performance, security, and accounting)
  – The functions are actually built in the community-based access policy of the SNMP administrative model.
SNMP Architecture

SNMP Manager

SNMP Application

Management Data

Get-Request

Get-Next-Request

Set-Request

Get-Response

Trap

SNMP

UDP

IP

DLC

PHY

SNMP Agent

SNMP Application

Get-Request

Get-Next-Request

Set-Request

Get-Response

Trap

SNMP

UDP

IP

DLC

PHY

Physical Medium

Figure 4.9 SNMP Network Management Architecture

SNMP Messages

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

- UDP is unreliable, so what happens if these messages are lost?
SNMP Trap Messages

- **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
    - Example SNMP Manager and SNMP Agent
      - The Pairing of these 2 names is known as a community
  - **SNMP protocol entities**
    - Communication processes (PDU handlers)
    - Peer processes that support application entities
SNMP Community

- **Security** in SNMPv1 is community-based
  - Not very secure, as we shall see
- 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
  - A local concept: local to the managed station
- Application could have multiple community names
- Communication is not secured in SNMPv1 – no encryption

**Figure 5.1 SNMP Community**

Managed station might give different access rights and views to different managers
Community Profile

- **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 + access mode
  - Operations on an object determined by community profile and the access privileges of the object (from the MIB definition)
  - There are a total of four access privileges
    - not-accessible
    - read-only
    - write-only
    - read-write
  - Some objects, such as table and table entry are non-accessible

Access Policy

- **Access Policy** is basically SNMP Community paired with Community Profile.
- **Example**:
  - 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
Access Mode, Access Policy, MIB Access

<table>
<thead>
<tr>
<th>MIB ACCESS Category</th>
<th>SNMP Access Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ-ONLY</td>
<td>READ-ONLY</td>
</tr>
<tr>
<td>READ-WRITE</td>
<td>READ-WRITE</td>
</tr>
<tr>
<td>WRITE-ONLY</td>
<td>WRITE-ONLY</td>
</tr>
<tr>
<td>not accessible</td>
<td>not accessible</td>
</tr>
</tbody>
</table>

Community Profile 1
Community Profile 2
Manager 2 (Community 2)
Agent 1
Agent 2
Agent 3
Community 2
Community Profile 3
Community Profile 4
Manager 3 (Community 1, Community 2)
Agent 1
Agent 2
Community Profile 1
Community Profile 2
Agent 1
Agent 2
Manager 1 (Community 1)

Generalized Administration Model

Manager 1 (Community 1)
Agent 1
Community Profile 1
Community Profile 2
Agent 2
Manager 3 (Community 1, Community 2)
Agent 3
Community Profile 3
Community Profile 4
Agent 1
Manager 2 (Community 2)
Proxy Access Policy

- 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.
- Proxy agent is programmed to know which MIB objects can be viewed (MIB view) and what’s their access mode.

![Figure 5.4 SNMP Proxy Access Policy](image)

Protocol Entities

- Protocol entities support application entities
- Communication between remote peer processes
- Message consists of
  - Version identifier
  - Community name
  - Protocol Data Unit
    - PDU format depends on the message (GET, SET, …)
- Message encapsulated and transmitted

![Figure 5.5 Encapsulated SNMP Message](image)
Get and Set PDU

PDU Types: enumerated INTEGER

<table>
<thead>
<tr>
<th>PDU Type</th>
<th>RequestID</th>
<th>Error Status</th>
<th>Error Index</th>
<th>VarBind 1 name</th>
<th>VarBind 1 value</th>
<th>...</th>
<th>VarBind n name</th>
<th>VarBind n value</th>
</tr>
</thead>
</table>

=0 for get and set

PDU Type

Get and Set Type PDUs

Prof. Shervin Shirmohammadi
CEG 4395
9-15

Trap PDU

- Enterprise (based on sysObjectID) 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
- time stamp indicates elapsed time since last re-initialization

<table>
<thead>
<tr>
<th>PDU Type</th>
<th>Enterprise</th>
<th>Agent Address</th>
<th>Generic Trap Type</th>
<th>Specific Trap Type</th>
<th>Timestamp</th>
<th>VarBind 1 name</th>
<th>VarBind 1 value</th>
<th>...</th>
<th>VarBind n name</th>
<th>VarBind n value</th>
</tr>
</thead>
</table>

Generic Trap Type

<table>
<thead>
<tr>
<th>Description (brief)</th>
</tr>
</thead>
<tbody>
<tr>
<td>coldStart(0)</td>
</tr>
<tr>
<td>warmStart(1)</td>
</tr>
<tr>
<td>linkDown(2)</td>
</tr>
<tr>
<td>linkUp(3)</td>
</tr>
<tr>
<td>authenticationFailure(4)</td>
</tr>
<tr>
<td>egpNeighborLoss(5)</td>
</tr>
<tr>
<td>enterpriseSpecific(6)</td>
</tr>
</tbody>
</table>
SNMP Operations

- Reading values using Get-Request

Manager Process

<table>
<thead>
<tr>
<th>GetRequest (sysDescr.0)</th>
<th>Agent Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetResponse (sysDescr.0 = &quot;SunOS&quot;)</td>
<td></td>
</tr>
<tr>
<td>GetRequest (sysObjectID.0)</td>
<td></td>
</tr>
<tr>
<td>GetResponse (sysObjectID.0 = enterprises.11.2.3.10.1.2)</td>
<td></td>
</tr>
<tr>
<td>GetRequest (sysUpTime.0)</td>
<td></td>
</tr>
<tr>
<td>GetResponse (sysUpTime.0 = 2247349530)</td>
<td></td>
</tr>
<tr>
<td>GetRequest (sysContact.0)</td>
<td></td>
</tr>
<tr>
<td>GetResponse (sysContact.0 = &quot;&quot;)</td>
<td></td>
</tr>
<tr>
<td>GetRequest (sysName.0)</td>
<td></td>
</tr>
<tr>
<td>GetResponse (sysName.0 = &quot;noc1&quot;)</td>
<td></td>
</tr>
<tr>
<td>GetRequest (sysLocation.0)</td>
<td></td>
</tr>
<tr>
<td>GetResponse (sysLocation.0 = &quot;)&quot;</td>
<td></td>
</tr>
<tr>
<td>GetRequest (sysServices.0)</td>
<td></td>
</tr>
<tr>
<td>GetResponse (sysServices.0 = 72)</td>
<td></td>
</tr>
</tbody>
</table>

- This returns one row, we have to repeat this for all the rows to read the whole table.

  - Not very efficient!

Lexicographic Order

- 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

<table>
<thead>
<tr>
<th>Numerical Order</th>
<th>Lexicographic order</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1118</td>
</tr>
<tr>
<td>3</td>
<td>115</td>
</tr>
<tr>
<td>9</td>
<td>126</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>34</td>
<td>22</td>
</tr>
<tr>
<td>115</td>
<td>250</td>
</tr>
<tr>
<td>126</td>
<td>2509</td>
</tr>
<tr>
<td>250</td>
<td>3</td>
</tr>
<tr>
<td>321</td>
<td>321</td>
</tr>
<tr>
<td>1118</td>
<td>34</td>
</tr>
<tr>
<td>2509</td>
<td>9</td>
</tr>
</tbody>
</table>
A More Complex MIB Example

Figure 5.14 MIB Example for Lexicographic Ordering

Get-Next-Request

• Uses logographical ordering
• Reads **columns** in order.
• Note that non-accessible objects (T and E) are not included in the operations.
Figure 5.15 Get-Next-Request Operation for MIB in Figure 5.12

Figure 5.16 GetNextRequest Example with Indices
Sniffer Data

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

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

SNMP MIB

Figure 5.21 SNMP Group
SNMPv1 Functional Model

- **Configuration**: setRequest
- **Fault**: trap messages
- **Performance**: counters maintained by agent; it’s manger’s duty to get values and measure performance.
- **Security**: administrative framework (not very secure at all)
- **Accounting**: N/A