Lecture 13:

SNMPv3

Prof. Shervin Shirmohammadi
SITE, University of Ottawa

Key Features of SNMPv3

• Modularization of documentation and architecture
  – Enables the use of SNMPv1 and SNMPv2 with the newly developed SNMPv3.

• SNMP engine defined
  – A model for the processing of SNMP messages.

• New Security features
  – Secure information to prevent tampering of data
  – Access control to determine proper access to MIB.
Documentation

- Better organized documentation architecture.

Architecture

- An SNMP entity is a node with an SNMP management element
  - either an agent or manager or both
- The architecture of an SNMP entity consists of the elements of the entity and the names associated with it.
- There are three names associated with an entity
  - Entities: e.g. SNMP engine (snmpEngineID)
  - Identities: e.g. Principal and security name
  - Management Information: Used when multiple objects are managed. E.g. Context engine

![Figure 7.2 SNMPv3 Architecture](image)
SNMP Engine ID

- Each SNMP engine has a unique ID: `snmpEngineID`
- Acme Networks {enterprises 696}
  - SNMPv1 `snmpEngineID`, Enterprise ID = ‘000002b8’H
  - SNMPv3 `snmpEngineID`, Enterprise ID = ‘800002b8’H
    (the 1st octet is 1000 0000)

```
SNMPv1
SNMPv2
SNMPv3
```

<table>
<thead>
<tr>
<th>Octet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1st bit</td>
</tr>
<tr>
<td>1</td>
<td>Enterprise ID (4 octets)</td>
</tr>
<tr>
<td>2</td>
<td>Format indicator (5th octet)</td>
</tr>
<tr>
<td>3</td>
<td>Format (variable number of octets)</td>
</tr>
</tbody>
</table>

![Figure 7.3 SNMP Engine ID](image)

Table 7.2 SNMPv3 Engine ID Format (5th Octet)

- **For SNMPv1 and SNMPv2:**
  - Octet 5 is the method that the enterprise used for deriving SNMP engine ID
  - Octet 6-12 is function of the method (i.e. an IP address)
- **Examples:** IBM host IP address 10.10.10.10
  - SNMPv1: 00 00 00 02 01 0A 0A 0A 0A 00 00 00
  - SNMPv3: 80 00 02 02 00 00 00 00 00 00 00 00 00 00 00 0A 0A 0A

```
<table>
<thead>
<tr>
<th>Octet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved, unused</td>
</tr>
<tr>
<td>1</td>
<td>IPv4 address (4 octets)</td>
</tr>
<tr>
<td>2</td>
<td>IPv6 (16 octets)</td>
</tr>
<tr>
<td>3</td>
<td>MAC address (6 octets)</td>
</tr>
<tr>
<td>4</td>
<td>Text, administratively assigned</td>
</tr>
<tr>
<td>5</td>
<td>Octets, administratively assigned</td>
</tr>
<tr>
<td>6-127</td>
<td>Reserved, unused</td>
</tr>
<tr>
<td>128-255</td>
<td>As defined by the enterprises</td>
</tr>
</tbody>
</table>
```

Engine ID Format (5th Octet)

- For SNMPv1 and SNMPv2:
  - Octet 5 is the method that the enterprise used for deriving SNMP engine ID
  - Octet 6-12 is function of the method (i.e. an IP address)
- **Examples:** IBM host IP address 10.10.10.10
  - SNMPv1: 00 00 00 02 01 0A 0A 0A 0A 00 00 00
  - SNMPv3: 80 00 02 02 00 00 00 00 00 00 00 00 00 00 00 0A 0A 0A
**Dispatcher**

- There is one dispatcher in an SNMP engine
- Handles multiple versions of SNMP messages
- Interfaces with the various modules: application modules, network modules, and message processing modules.
- Three components for three functions:
  - **Transport mapper** delivers messages over the transport protocol.
  - **Message Dispatcher** routes messages between network and appropriate module of Message Processing Subsystem (MPS).
  - **PDU dispatcher** handles messages between applications and MPS

---

**Message Processing Subsystem (MPS)**

- Contains one or more Message Processing Models (MPM)
- One MPM for each SNMP version
- SNMP version identified in the header
Security and Access Control

- Security at the message level
  - Authentication
  - Privacy of message via secure communication

- Flexible access control
  - Who can access?
  - What can be accessed?
  - Flexible MIB views?

SNMP Engine (identified by snmpEngineID)

Applications

Manager Apps

Agent Apps

Command Generator

Command Responder

Notification Originator

Notification Receiver

Proxy Forwarder Subsystem

Other

Application(s)

Example

get-request

get-response

trap generation

trap processing

get-bulk to get-next

Special application

Prof. Shervin Shirmohammadi

CEG 4395

13-9
More on Names

- An SNMP agent can monitor more than one network element (context)

<table>
<thead>
<tr>
<th>Entity / Identity / MI</th>
<th>Entity Name</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SNMP Engine ID</td>
<td>snmpEngineID</td>
<td>IP address</td>
</tr>
<tr>
<td>• Principal</td>
<td>principal</td>
<td>John Smith</td>
</tr>
<tr>
<td>• Security Name</td>
<td>securityName</td>
<td>Administrator</td>
</tr>
<tr>
<td>• Context Engine ID</td>
<td>contextEngineID</td>
<td>00 00 00 02 01 0A 0A 0A 0A 0A 00 00 00</td>
</tr>
<tr>
<td>• Context Name</td>
<td>contextName</td>
<td>00 00 00 02 01 0A 0A 0A 0A 0A 0A 00 00 00</td>
</tr>
</tbody>
</table>

Identity is composed of:

- **principal** - Who: person or group or application
- **securityName** - human readable name

Abstract Service Interface

- The subsystems in the SNMP entity communicate with each other over abstract service interfaces.
- An abstract service interface is a **conceptual interface** between modules, independent of implementation.
- Defines a set of **abstract primitives**
- Primitives are generally associated with the receiving sub-system except for the Dispatcher
  - Contain **IN** and **OUT** parameters
  - Can return **Status** information / result
- Dispatcher primitives associated with
  - receiving messages to and sending from applications
  - registering and un-registering application modules
  - transmitting to and receiving messages from the network
Abstract Service Interface

<table>
<thead>
<tr>
<th>Module</th>
<th>Primitive</th>
<th>Service Provided</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispatcher</td>
<td>sendPdu</td>
<td>Request from application to send a PDU to a remote entity</td>
</tr>
<tr>
<td>Dispatcher</td>
<td>processPdu</td>
<td>Processing of incoming message from remote entity</td>
</tr>
<tr>
<td>Dispatcher</td>
<td>returnResponsePdu</td>
<td>Request from application to send a response PDU</td>
</tr>
<tr>
<td>Dispatcher</td>
<td>processResponsePdu</td>
<td>Processing of incoming response from a remote entity</td>
</tr>
<tr>
<td>Dispatcher</td>
<td>registerContextEngineID</td>
<td>Register request from a Context Engine</td>
</tr>
<tr>
<td>Dispatcher</td>
<td>unregisterContextEngineID</td>
<td>Unregister request from a Context Engine</td>
</tr>
</tbody>
</table>
Dispatcher sendPDU Primitive

- sendPdu (destined for a remote entity) request is sent by the application module, **command generator**, and is associated with the receiving module, the dispatcher.
- After the message is transmitted over the network, the dispatcher sends a handle to the command generator for tracking the response.
- sendPdu is the **IN** parameter.
- sendPduHandle is the **OUT** parameter, shown as coupled to the **IN** parameter.

![Diagram](Image)

**Figure 7.4(b) Abstract Service Interface for sendPdu**

Command Generator Application

- Generates
  - get-request
  - get-next-request
  - get-bulk
  - set-request
- Processes the response
**Command Responder Application**

- Processes the get and set requests destined for a remote entry.
- Prepares get-response message

**Notification / Proxy**

- **Notification originator**
  - Generates trap and inform messages
  - Determine target, SNMP version, and security
  - Decides name of the context that has the information.

- **Notification receiver**
  - Registers with SNMP engine
  - Receives notification messages

- **Proxy forwarder**
  - Exhibits Proxy server functionality
  - Handles only SNMP messages (different than Proxy Agent) by
    - Command generator
    - Command responder
    - Notification generator
    - Report indicator
  - Uses the translation table in the proxy group MIB
SNMPv3 MIB

- SNMPv3 MIB developed under `snmpModules` of the same convention that SNMPv2 has defined.
- Security placeholder not used.

![Figure 6.31 SNMPv2 Internet Group](image)

snmpModules Objects

- `snmpFrameworkMIB` describes SNMP management architecture
- `snmpMPDMIB` identifies objects in the message processing and dispatch subsystems
- `snmpTargetMIB` and `snmpNotificationMIB` used for notification generation
- `snmpProxyMIB` defines translation table for proxy forwarding
- `snmpUsMIB` defines user-based security model objects
- `snmpVacmMIB` defines objects for view-based access control

![Figure 7.7 SNMPv3 MIB](image)
SNMPv3 Target MIB

• Target MIB contains **two tables:**
  
  • **Target address table** contains:
    – addresses of the targets for notifications (see notification group)
    – information for establishing the transport parameters
    – reference to the second table, target parameter table
  
  • **Target parameter table** contains security parameters for authentication and privacy

SNMP Target Address Table

*Table 7.4*

<table>
<thead>
<tr>
<th>OID</th>
<th>Description (brief)</th>
</tr>
</thead>
<tbody>
<tr>
<td>snmpTargetObjects 2</td>
<td>Table of transport addresses</td>
</tr>
<tr>
<td>snmpTargetAddrTable 1</td>
<td>Row in the target address table</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 1</td>
<td>Locally administered name associated with this entry</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 2</td>
<td>Transport type of the addresses</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 3</td>
<td>Transport address</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 4</td>
<td>Expected maximum round-trip time</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 5</td>
<td>Number of retries</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 6</td>
<td>List of tag values used to select the target addresses for a particular operation</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 7</td>
<td>Value that identifies an entry in the snmpTargetParams Table</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 8</td>
<td>Storage type for this row</td>
</tr>
<tr>
<td>snmpTargetAddrEntry 9</td>
<td>Status of the row</td>
</tr>
</tbody>
</table>
SNMP Target Parameters

Table 7.5

<table>
<thead>
<tr>
<th>OID</th>
<th>Description (brief)</th>
</tr>
</thead>
<tbody>
<tr>
<td>snmpTargetObjects 3</td>
<td>Table of SNMP target information to be used</td>
</tr>
<tr>
<td>snmpTargetParamsTable 1</td>
<td>A set of SNMP target information</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 1</td>
<td>Locally administered name associated with this entry</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 2</td>
<td>Message processing model to be used</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 3</td>
<td>Security model to be used</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 4</td>
<td>Security name of the principal</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 5</td>
<td>Level of security</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 6</td>
<td>Storage type for the row</td>
</tr>
<tr>
<td>snmpTargetParamsEntry 7</td>
<td>Status of the row</td>
</tr>
</tbody>
</table>

SNMPv3 Notification MIB

- Notification group contains three tables:
  - **Notify table** contains groups of management targets to receive notifications and the type of notifications
  - The target addresses to receive notifications that are listed in target address table (see target group) are tagged here
  - **Notification filter profile table** defines filter profiles associated with target parameters
  - **Notification filter table** contains table profiles of the targets

Figure 7.9 SNMP Notification Tables