SNMP

Content

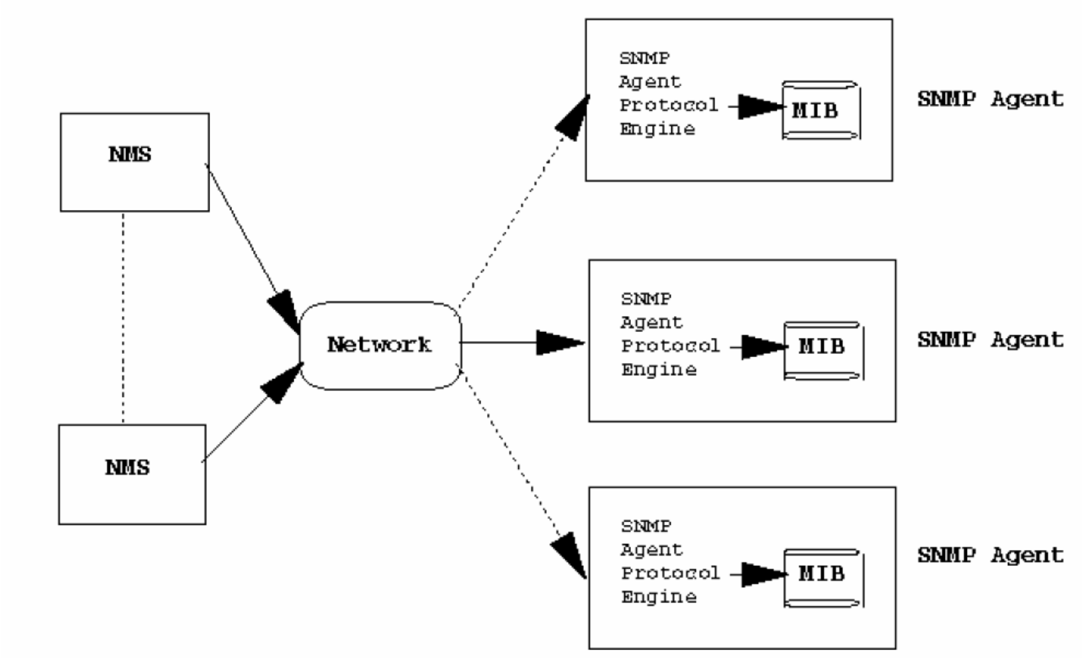
1. Introduction

Since its creation in 1988 as a short-term solution to manage elements in the growing Internet and other attached networks, Simple Network Management Protocol has achieved widespread acceptance and has become the *de facto* standard for internetwork management. SNMP was first defined by the IETF (Internet Engineering Task Force) in 1989, and was widely extended since. SNMP is applicable to TCP/IP networks, as well as other types of networks. SNMP was derived from its predecessor SGMP (Simple Gateway Management Protocol) and was intended to be replaced by a solution based on the CMIS/CMIP (Common Management Information Service/Protocol) architecture. This long-term solution, however, never received the widespread acceptance of SNMP.

2. SNMP architecture

SNMP is based on the manager/agent model consisting of a manager, an agent, a database of management information, managed objects and the network protocol. The manager provides the interface between the human network manager and the management system. The agent provides the interface between the manager and the physical device(s) being managed, such as bridges, hubs, routers or network servers, these managed objects might be hardware, configuration parameters, performance statistics, and so on… These objects are arranged in what is known as a virtual information database, called a Management Information Base, also called

MIB. SNMP allows managers and agents to communicate for the purpose of accessing these objects.



NMS : Network Management Station

Figure 1 : The model of network management architecture

SNMP uses five basic messages (Get, GetNext, GetResponse, Set, and Trap) to communicate between the manager and the agent. The Get and GetNext messages allow the manager to request information for a specific variable. The agent, upon receiving a Get or GetNext message, will issue a GetResponse message to the manager with either the information requested or an error indication as to why the request cannot be processed. A Set message allows the manager to request a change be made to the value of a specific variable in the case of an alarm remote that will operate a relay. The agent will then respond with a GetResponse message indicating the change has been made or an error indication as to why the change cannot be made. The Trap message allows the agent to spontaneously inform the manager of an 'important' event.

As you can see, most of the messages (Get, GetNext, and Set) are only issued by the SNMP manager. Because the Trap message is the only message capable of being initiated by an agent, it is the message used by Remote Telemetry Units (RTUs) to report alarms. This notifies the SNMP manager as soon as an alarm condition occurs, instead of waiting for the SNMP manager to ask.

The small number of commands used is only one of the reasons SNMP is "simple". The other simplifying factor is its reliance on an unsupervised or connectionless communication link. This simplicity has led directly to its widespread use, specifically in the Internet Network Management Framework. Within this framework, it is considered 'robust' because of the independence of the managers from the agents, e.g. if an agent fails, the manager will continue to function, or vice versa.

3. The Management Information Base

The manager and agent use a Management Information Base and a relatively small set of commands to exchange information. The MIB is organized in a tree structure with individual variables, such as point status or description, being represented as leaves on the branches. A long numeric tag or object identifier (OID) is used to distinguish each variable uniquely in the MIB and in SNMP messages.

The MIB lists the unique object identifier of each managed element in an SNMP network. The SNMP manager can’t monitor devices unless it has compiled their MIB files. The MIB is also a guide to the capabilities of SNMP devices. For example, if MIB lists OIDs for Traps but not for GetResponse messages, it will report alarms, but will not respond to alarm polls.

Each SNMP element manages specific objects with each object having specific characteristics. Each object / characteristic has a unique object identifier consisting of numbers separated by decimal points (i.e., 1.3.6.1.4.1.2682.1). These object identifiers naturally form a tree as shown below. The MIB associates each OID with a readable label and various other parameters related to the object. The MIB then serves as a data dictionary or code book that is used to assemble and interpret SNMP messages.

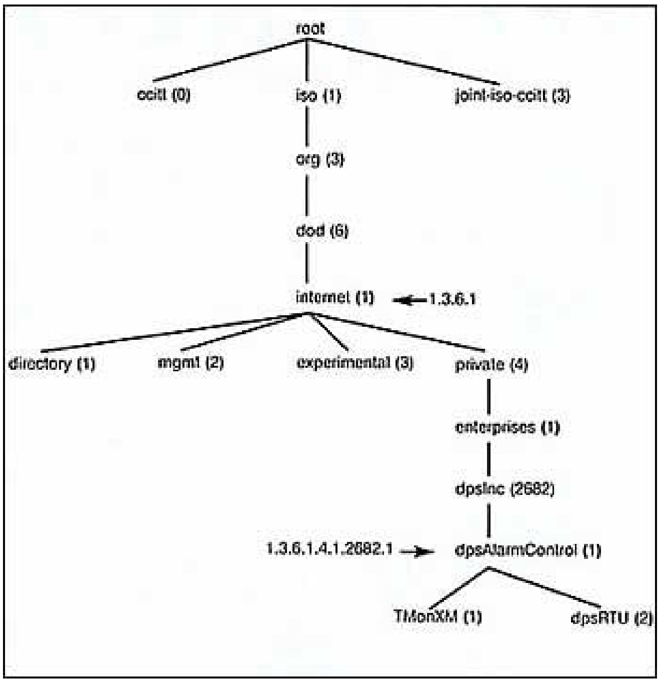


Figure 2 : The MIB tree structure

When an SNMP manager wants to know the value of an object / characteristic, such as the state of an alarm point, the system name, or the element uptime, it will assemble a Get packet that includes the OID for each object / characteristic of interest. The element receives the request and looks up each OID in its code book (MIB). If the OID is found (the object is managed by the element), a response packet is assembled and sent with the current value of the object / characteristic included. If the OID is not found, a special error response is sent that identifies the unmanaged object.

When an element sends a Trap packet, it can include OID and value information (bindings) to clarify the event. Remote units send a comprehensive set of bindings with each Trap to maintain traditional telemetry event visibility. Well-designed SNMP managers can use the bindings to correlate and manage the events. SNMP managers will also generally display the readable labels to facilitate user understanding and decision-making.

4. Packet types and structure

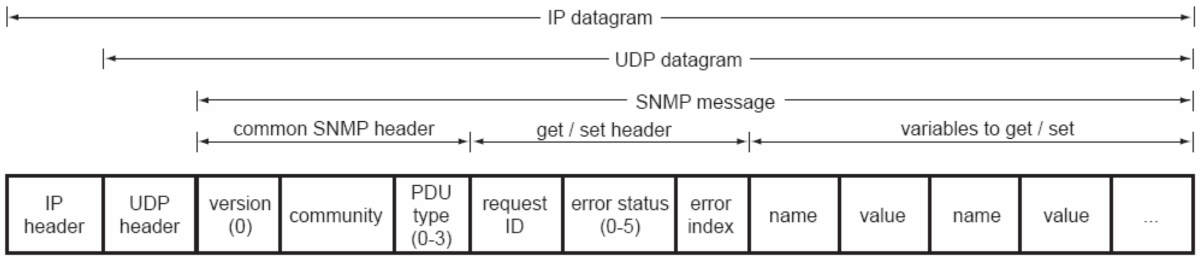


Figure 3 : The SNMP data packet is enclosed in the UDP data packet, which is enclosed in the IP data packet

Note : UDP (User Datagram Protocol) is the IP transport layer protocol that supports SNMP messages. Unlike TCP, UDP is a connectionless protocol. A UDP host places messages on the network without first establishing a connection with the recipient. UDP does not guarantee message delivery, but it's a lightweight protocol that can transport a large number of status messages without using too many network resources.

Let's examine the communication between managers and agents. Basic serial telemetry protocols, are byte-oriented, with a single byte exchanged to communicate. Expanded serial telemetry protocols, are packet oriented with packets of bytes exchanged to communicate. The packets contain header, data and checksum bytes. SNMP is also packet oriented with the following SNMP v1 packets (Protocol Data Units or PDUs) used to communicate:

Get
GetNext
Set
GetResponse
Trap

The manager sends a Get or GetNext to read a variable or variables and the agent's response contains the requested information if managed. The manager sends a Set to change a variable or variables and the agent's response confirms the change if allowed. The agent sends a Trap when a specific event occurs. Figure 3 (above) shows the packet formats. Each variable binding contains an identifier, a type and a value (if a Set or GetResponse). The agent checks each identifier against its MIB to determine whether the object is managed and changeable (if processing a Set). The manager uses its MIB to display the readable name of the variable and sometimes interpret its value.

5. Layered communication

We continue to examine the Simple Network Management Protocol focusing specifically on the layered communication model used to exchange information. The last section focused on the structure of SNMP messages, however an SNMP message is not sent by itself. It is wrapped in the User Datagram Protocol (UDP), which in turn is wrapped in the Internet Protocol (IP). These are commonly referred to as layers and are based on a four-layer model. SNMP resides in what is called the Application layer, UDP resides in the Transport layer and IP resides in the Internet layer. The fourth layer is the Network Interface layer where the assembled packet is actually interfaced to some kind of transport media (for example, twisted pair copper, co-axial or fiber). This multi-layer model isolates the tasks of communication and ultimately assists in designing and implementing a network.

➢ Traversing the Layers

To illustrate the function of this layered model, let’s look at a single SNMP Get request from the agent’s perspective.

The SNMP manager wants to know what the Agent’s System Name is and prepares a Get message for the appropriate OID. It then passes the message to the UDP layer. The UDP layer adds a data block that identifies the manager port to which the response packet should be sent and the port on which it expects the SNMP agent to be listening for messages. The packet thus formed is then passed to the IP layer. Here a data block containing the IP and Media Access addresses of the manager and the agent is added before the entire assembled packet gets passed to the Network Interface layer. The Network Interface layer verifies media access and availability and places the packet on the media for transport.
After working its way across bridges and through routers based on the IP information, the packet finally arrives at the agent.
Here it passes through the same four layers in exactly the opposite order as it did at the manager. First, it is pulled off the media by the Network Interface layer. After confirming that the packet is intact and valid, the Network Interface layer simply passes it to the IP layer. The IP layer verifies the Media Access and IP address and passes it on to the UDP layer where the target port is checked for connected applications. If an application is listening at the target port, the packet is passed to the Application layer. If the listening application is the SNMP agent, the Get request is processed. The agent response then follows the identical path in reverse to reach the manager.

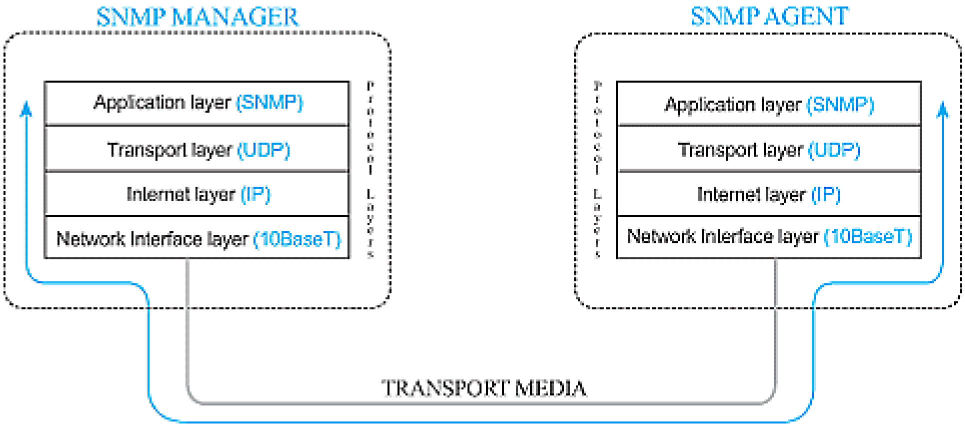


Figure 4 : Protocol layers in SNMP communication

6. References

ftp://ftp.rfc-editor.org/in-notes/rfc1157.txt
http://www.snmp.com/
http://www.DpsTelecom.com/Snmp/Tutorial
http://www2.rad.com/networks/1999/snmp/index.htm
http://www.unix.org.ua/orelly/perl/sysadmin/appe_01.htm
http://www.snmp.com/
http://www2.rad.com/networks/1995/snmp/snmp.htm