

ENFORCING DYNAMIC HOST CONFIGURATION PROTOCOL

A

*Dissertation submitted to JAWAHARLAL NEHRU UNIVERSITY
in partial fulfillment of the requirements for the award of the degree of*

**Master of Technology
in
Computer Science and Technology**

By
Arvind Kumar



Jawaharlal Nehru University

School of Computer & System Sciences
Jawaharlal Nehru University
New Delhi -110067
January 2001

lib code
1802

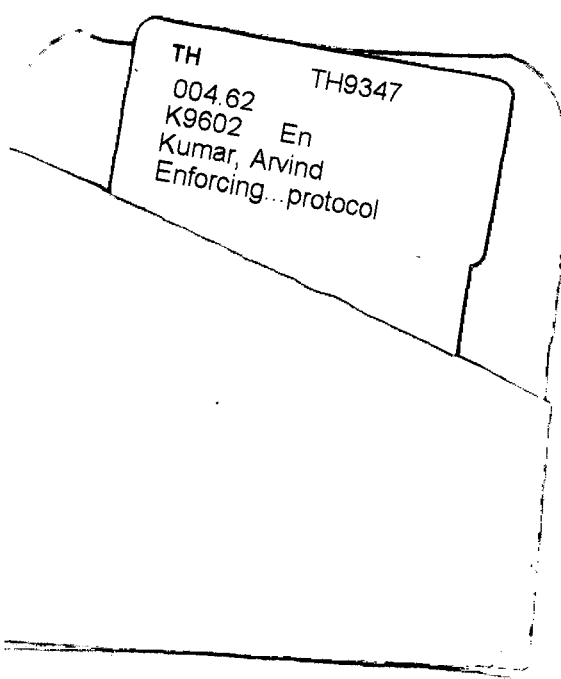
ENFORCING DYNAMIC HOST CONFIGURATION PROTOCOL

A

*Dissertation submitted to JAWAHARLAL NEHRU UNIVERSITY
in partial fulfillment of the requirements for the award of the degree of*

**Master of Technology
in
Computer Science and Technology**

By
Arvind Kumar



Jawaharlal Nehru University
Computer & System Sciences
Jawaharlal Nehru University
New Delhi -110067
January 2001




जवाहरलाल नेहरू विश्वविद्यालय
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067 (INDIA)
SCHOOL OF COMPUTER & SYSTEMS SCIENCES


Certificate

This is to certify that the dissertation entitled "**ENFORCING DYNAMIC HOST CONFIGURATION PROTOCOL**" which is being submitted by **ARVIND KUMAR** to the School of Computer & Systems Sciences, *JAWAHARLAL NEHRU UNIVERSITY* for the award of **MASTER OF TECHNOLOGY** in **COMPUTER SCIENCE & TECHNOLOGY** is a bonafide work carried out by him under our supervision.

This is original and has not been submitted in part or full to any university or institution for the award of any degree.


Prof. G.V. Singh
(Supervisor)

School of Comp. & Systems Sci.
Jawaharlal Nehru University
New Delhi


Dr. D.K. Lobiyal
(Supervisor)

School of Comp. & Systems Sci.
Jawaharlal Nehru University
New Delhi


Prof. C.P. Katti

Dean

School of Comp. & Systems Sci.
Jawaharlal Nehru University

Acknowledgements

First and foremost I would like to thank my supervisors Prof. G.V. Singh And Dr. D.K.Lobiyal, not only for their guidance of my work but also for patience and forbearance with which they dealt with my shortcomings. While typing and proof reading, which consider with most boring job in academics, they created intellectual excitement and devoted their energy and time in patient reading after draft. They give me a psychological boost during the depressed hours.

I also thank to Lab mates for providing friendly environment in the lab. I thank to Mr. Bhupendra, and Mr.Vijendra Chauhan for their cooperation and being such a good company in those long hours in the lab.

No endeavor is complete without the help of certain friends. A special mention is made here for Mr. Raja rai, Mr. J.K.Sharma and Mr. G.Majhi.

Last but not the least, I mention my parents, all my brothers and my sister in law.

If certain names are missing still their names are not missing from my heart.

Arvind Kumar

माँ की प्रेरणा से ..

दाऊ को समर्पित

CONTENTS

Topic	Page No.
1. Introduction	1
1.1 Assigning an IP address	2
1.2 Basics of DHCP	3
1.3 Problem Definition	5
1.4 Organization of the report	5
2. Dynamic Host Configuration Protocol: an Overview	6
2.1 Why DHCP	6
2.2 DHCP Message Format	6
2.3 DHCP option and message Type	8
2.4 Protocol Flow of DHCP	9
2.5 DHCP Sever and client interaction	11
2.6 Lease Termination	12
2.6.1 BOUND States	12
2.6.2 Client state	13
2.6.3 The DHCP lease	14
2.6.4 Getting the lease	14
2.6.6 Renewing the lease	15
2.6.6 Rebinding the lease	15
2.6.7 Eviction	15
3. Simple Network Management Protocol	17
3.1 Introduction	17
3.2 SNMP Basic Components	17
3.3 SNMP Basic Commands	19
3.4 SNMP Management Information Base (MIB)	20
3.5 Data Representation in SNMP	22
3.6 Structure of Management Information in SNMP	22
3.6.1 ASN1 Data Types	22
3.6.2 SMI-Specific Data Types	23

3.6.3 SNMP MIB Tables	23
3.7 SNMP Protocol Operations	24
3.8 SNMP Management	24
3.9 SNMP Security	24
3.10 SNMP Reference	25
3.10.1 SNMP Message Formats	25
3.10.2 SNMP Message Header	25
3.10.3 SNMP Protocol Data Unit (PDU)	26
3.10.4 Trap PDU Format	27
4. Data Structure	28
4.1 Introduction	28
4.2 Data Organization	30
4.2.1 Table- 1. Ip_mac_map_table	30
4.2.2 Table- 2. Mac_sw_port_map_table	31
4.2.3 Table- 3. Mac_hub_port_mapping	31
4.2.4 Table- 4 Switch_port_data_table	32
4.3 Organisation of Ip_mac_map_table	33
4.4 Organisation of mac_sw_port_map_table	34
4.5 Orgnisation of the mc_hub_port_map_table_node	34
4.6 Organisation of the switch_port_data_table	35
5. Algorithms	37
5.1 Algorithm for finding MAC Address	37
5.2 Algorithm to finding Switch information	38
5.3 Algorithm for finding Hub information	39
5.4 Algorithm for finding Rogue host	40
5.5 Algorithm for IP allocation	41
5.6 Algorithm for Update Data Information	42
5.7 Using updation algorithm	43
6. Conclusion	44
Reference	45

Abstract

DHCP provides configuration parameters to Internet hosts. DHCP consists of two components: a protocol for delivering host-specific configuration parameters from a DHCP server to a host and a mechanism for allocation of network address to the host. DHCP is built on a client server model, where designated DHCP server host allocates network address and deliver configuration parameters to dynamically configured hosts.

The design goal of DHCP was primarily that it should be a mechanism for dynamic allocation of addresses and not that it should be a mechanism for network management. By this design goal the developers meant that any mechanism to make the dynamic allocation of addresses by the DHCP server en-forcible over the network had to be external to the DHCP mechanism. This led to the problem of “rouge” hosts. Rouge host means a host that has not received a dynamic IP address from the DHCP server, which it was supposed to take. In such a case the network security could also be compromised as unauthorized hosts could breach the network.

For getting information about a network SNMP is the most used protocol. The use of this protocol can collect the information about network devices (hub, router, switch etc.) which support SNMP.

The information collected about the network devices gives the details of the hosts and their network access points. This information is compared with the information provided by DHCP to determine the rouge hosts. Since the network access point of the all the hosts are known, the rouge hosts at their network access point can be blocked.

The polling interval (i.e. the time period to get the entire information) of network data collection have been taken into consideration, so that the performance of the network does not go down

Various data structures in form of the tables that reflect the state of the network have been designed to store the information about the network devices. This includes router_data table, switch_data table, hub_data table, and remote_access_data table. Algorithms to detect the unauthorized host, to block a rogue host, and to update the various tables have been designed.

Chapter 1

Introduction

In this modern era, data communication is on the bloom. The requirement of this service has been increased exponentially. This demand has created some limitations for the service providers.

For global data communication each computer is assigned an address called the IP address. This IP address is 4 byte long. And this limits the address space to maximum of 2^{32} possible address. This space is insufficient to cater the increasing demand. There are much more computers to be connected. This need has been some how fulfilled by using many techniques, one of which is dynamic host control protocol (DHCP).

In DHCP, IP addresses are dynamically allocated to the hosts connected to the Local Area Network (LAN). This dynamic allocation is done by a server in the network called the DHCP server.

This server allocates the IP address to a computer (or host) dynamically and for a period of time. When this time period elapses then the IP address has been de-allocated and the host should send request for the reallocation of the IP address to itself. This time period is called a lease period.

Computer, on a network, used by a student in a university might have a short lease period. By contrast, a corporate network might use a lease period of one day or one week. To accommodate all possible environment's DHCP does not specify a fixed constant for the lease period. Instead, the protocol allows a client to request a specific lease period, and allows a server to inform the client of lease period it grants. Thus, a manager can decide how long each server should allocate an address to a client. In the extreme, DHCP reserve a value for infinity to permit a lease to last arbitrary long like the permanent address assignment used in BOOTP.

The BOOTstrap protocol (BOOTP) provides an alternative to RARP for a computer that needs to determine its IP address. BOOTP is more general than RARP because it uses UDP, making it possible to determine a router address, a server address, and the name of a program the computer should run. Finally, BOOTP allows administrator to establish a configuration database that maps a generic name, like “Unix”. The client uses the limited broadcast address to communicate with the server, and takes responsibility for retransmitting requests if the server does not respond. Retransmission uses an exponential backoff policy similar to Ethernet to avoid congestion.

Designed as a successor to BOOTP, the Dynamic Host Configuration Protocol (DHCP) extends BOOTP in several ways. Most important, DHCP permits a server to allocate IP addresses automatically or dynamically. Dynamic allocation is necessary for environments such as a wireless network where computers can attach and detach quickly. To use DHCP, a computer becomes a client. The computer can broadcast a request for DHCP server to obtain a lease on the advertised IP address.

When a client obtains an IP address, the client starts three timers. After the first timer expires, the client attempts to renew its lease. If a second timer expires before renewal completes, the client attempts to rebind its address from any server. If the final timer expires before a lease has been renewed, the client stops using the IP address and return to the initial state to acquire a new address. A finite state machine explains lease acquisition and renewal.

1.1 Assigning an IP address

There are generally three pieces of information a system needs in order to start communicating on a TCP/IP network : an IP address to uniquely identify a system on the network, a subnet mask to determine the subnet, and at least one default router address. If a machine does not need to communicate beyond its immediate subnet, the router address and the machine's own IP address are set the same. These three values represent the bare minimum needed for a system to participate in the TCP/IP world, and they are vital. However, with networks changing so quickly, manually programming these values into

each and every device attached to the network and programming them, as they can change quickly become tiresome.

1.2 Basics of DHCP

DHCP is extension of BOOTP. DHCP uses client-server model. DHCP has server and client model with relay agent same as BOOTP. It relays messages between the client and the server .

In DHCP all interactions are initiated by a client, and a server replies. DHCP has following implementation goals;

- DHCP should support dynamic allocation. DHCP is capable of leasing the address, and DHCP server should be able to recycle the IP address when the lease period is expired.
- A network administrator should not configure each client, and user interaction of each client should not be required.
- DHCP should also support static allocation and an infinite lease.
- DHCP should coexist with BOOTP and normal host.
- DHCP should not require a server on each subnet. Relay agent which is compatible with BOOTP can be used.

Only allocation model supported by BOOTP is static allocation. On the other hand, DHCP supports three models;

Manual allocation: Server allocates an IP address which has been chosen by the administrator.

Automatic allocation: Server allocates an IP address with infinite lease.

Dynamic allocation: Server chooses and allocates an IP address with finite lease.

In *manual allocation method*, the network administrator on the DHCP server manually configures the client's IP address in the server. When the client workstation makes the request for an IP address, the server looks at the MAC address (Media Access

Control address; manufacture's unique address of the network card) and assigns the client the manually set IP address.

In *automatic allocation method*, the DHCP client workstation is assigned an IP address when it first contacts the DHCP server. In this method the IP address is randomly assigned and is not set in the server. The IP address is permanently assigned to the DHCP client and is not reused by another DHCP client.

In *dynamic allocation method*, the DHCP server assigns an IP address to a requesting client workstation on a temporary basis. The IP address is leased to the DHCP client for a specified duration of time. When this lease expires, the IP address is revoked from the client and the client is required to surrender the address. If the DHCP client still needs an IP address to perform its functions, it can request another IP address.

DHCP server manages two database to recognize the network status: *Address Pool* is the database which holds IP addresses and other network configuration parameters. *Binding database* keeps mapping between an Ethernet address and entry of Address Pool. Dynamic allocation is the only one of the three mechanisms that allows automatic reuse of an address that is no longer needed by the client to which it was assigned. Thus, dynamic allocation is particularly useful for assigning an address to a client that will be connected to the network only temporarily or for sharing a limited pool of IP addresses among a group of clients that do not need permanent IP addresses. Dynamic allocation may also be a good choice for assigning an IP address to a new client being permanently connected to a network where IP addresses are sufficiently scarce. It is important to reclaim them when old clients are retired. Manual allocation allows DHCP to be used to eliminate the error-prone process of manually configuring hosts with IP addresses in environments where (for whatever reasons) it is desirable to manage IP address assignment outside of the DHCP mechanisms.

1.3 Problem Definition

The design goal of DHCP was primarily that it should be a mechanism for dynamic allocation of IP addresses and not that it should be a mechanism for network management. By this design goal the developers meant that any mechanism to make the dynamic allocation of address by the DHCP server en-forcible over the network had to be external to the DHCP mechanism. This led to the problem of “rogue” hosts. Rogue host means a host has not received a dynamic IP address from the DHCP server, which it was supposed to take. In such a case the network security could also be compromised as unauthorized hosts could breach the network.

- DHCP is a mechanism for dynamic allocation of address and not a mechanism for network management
- No standard mechanism is there to enforce the DHCP (i.e. If some host uses a IP address reserved for Dynamic IP address, there is no mechanism to stop the “Rouge” Host)
- Get the information from the network devices about the IP addresses and the MAC addresses on the network.
- Get information from the DHCP server about the IP Addresses allocated and the free IP addresses available with the DHCP server.

1.4 Organization of the report

The entire dissertation consists of six chapters. In the first chapter, problems are introduced. It describes about hosts, which can be unauthorized, and ways to find it. The second chapter describes about dynamic host configuration protocols, client server interaction, lease termination of the IP addresses, the ways to get IP address and renewal of IP address after lease time has been expired. In the third chapter there is discussion about SNMP and it's use for managing DHCP. Last two chapters specifies data structure and algorithms for enforcing DHCP.



Chapter 2

Dynamic Host Configuration Protocol: an Overview

2.1 Why DHCP

Computer networks always seem to be changing. Number of host in the network keeps on increasing while the number of IP addresses are limited. Allocating IP addresses statically will disallow new hosts to use the Internet services. Therefore, dynamic allocation of addresses will allow all hosts to use Internet services on time sharing basis.

2.2 DHCP Message Format

DHCP uses the BOOTP message format, but modifies the contents and meaning of some fields. DHCP uses the same packet format with BOOTP for its compatibility (Figure. 1). DHCP server can respond to BOOTP client and vice versa.

Op(1)		htype(1)		hlen(1)		hops(1)
Xid(4)						
secs (2)				flags (2)		
Ciaddr						
Yiaddr(4)						
Siaddr(4)						
Giaddr(4)						
Chaddr(16)						
Sname (64)						
File(128)						
Options(312)						

Figure- 1 (DHCP Packet Format)

In the figure, most of the fields in a DHCP message are identical to fields in BOOTP message. In fact, the two protocols are compatible; a DHCP server can change the meaning of two fields. First, DHCP interprets BOOTP's UNUSED field as a 16-bit FLAGS field.



Figure-2 (The format of the 16-bit FLAGS field in a DHCP message)

In fact, figure-1 shows that only high-order bit of the FLAGS field has been assigned a meaning. Because the DHCP request message contains the client's hardware address, a DHCP server normally sends its responses to the client using the hardware unicast. A client sets the high-order bit in the FLAGS field to request that the server respond using hardware broadcast instead of hardware unicast. To understand why a client might choose a broadcast response, recall that while a client communicates with a DHCP server, it does not yet have an IP address. If a datagram arrives via hardware unicast and the destination address does not match the computer's address, IP can discard the datagram. However, IP is required to accept and handle any datagram sent to the IP broadcast address. To ensure IP software accepts and delivers DHCP messages that arrive before the machine's IP address has been configured, a DHCP client can request that the server send responses using IP broadcast.

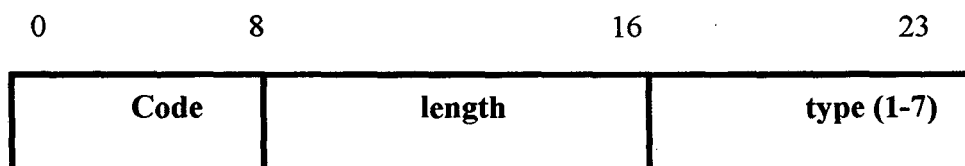
Table-1(given below) is a part of the description of fields in a DHCP message. BOOTP uses only two message types; **BootRequest** and **BootReply**. These message types are also used in DHCP for compatibility. However, DHCP puts another message type in the *options* field to distinguish various messages. There are seven message types for DHCP (Table- 2).

Field	Description
Op	Message type. BootRequest / BootReply
Xid	Transaction ID, a random number chosen by the

	client, used by the client and server to associate messages and responses between a client and server.
Ciaddr	Client IP address; only filled in if client can responds to ARP requests.
Yiaddr	'your' (client) IP address.
Giaddr	Relay agent IP address, used in booting via a relay-agent.
File	Boot file name.
Options	Optional parameters field.

Table 1: DHCP message fields' Description

2.3 DHCP option and message Type



Type Field	Corresponding DHCP message Type
1	DHCPDISCOVER
2	DHCPOFFER
3	DHCPREQUEST
4	DHCPDECLINE
5	DHCPACK
6	DHCPNACK
7	DHCPRELEASE

Figure-2 : (The Format of a DHCP message type option)

It is used to specify the DHCP message being sent. The table lists possible values of the third octet and their meaning.

<i>Message</i>	<i>Description</i>
Discover	A client broadcast to recognize servers.
Offer	Server response with proposal of parameters.
Request	The client broadcasts to the preferable server. An implicit decline to others.
Ack	The server assigns an IP address.
Nak,	The server rejects the request from the client.
Decline	The client found a problem with an assigned address.
Release	The client returns the assigned address before its lease expires.

Table 2: DHCP Message Type

2.4 Protocol Flow of DHCP

As stated above, DHCP is based on the client-server model. The process of an initial assignment can be divided into two phases; In the first step, client broadcasts a DhcpDiscover to collect proposals from servers. In the second step, the client chooses one offer from the server, and requests the server to assign the address.

DHCP protocol flows are different with getting a new address, and getting the previously assigned address (renewing). *Figure 2* given below shows the first case :

- DHCP client broadcasts a DhcpDiscover. The client may specify preference of a lease and/or a IP address.
- DHCP server receiving the DhcpDiscover may return DhcpOffer or may not return (Many servers may receives the DhcpDiscover). If a server decides to respond, server puts an available address into *yiaddr* field and broadcasts the DhcpOffer. At this point, there is no agreement of an assignment between the server and the client.
- Client gets one or more DhcpOffers, and chooses one server from them. The client puts the IP address of the server into the *Server identifier* option of a DhcpRequest and broadcasts it to the server.

- The server check the *Server identifier* option. If it does not match with its own address, the server consider it as an implicit decline. The selected server sends the DhcpAck (if its address is available) or the DhcpNak (for example, the address is already assigned to another client).
- The client which gets the DhcpAck starts using the IP address. If it gets DhcpNak, it restarts from step 1.
- The client finds a problem with the assigned address of DhcpAck, it sends DhcpDecline to the server, and restarts from step 1.
- The client can release the address before its lease expires by DhcpRelease. This process is not required absolutely.

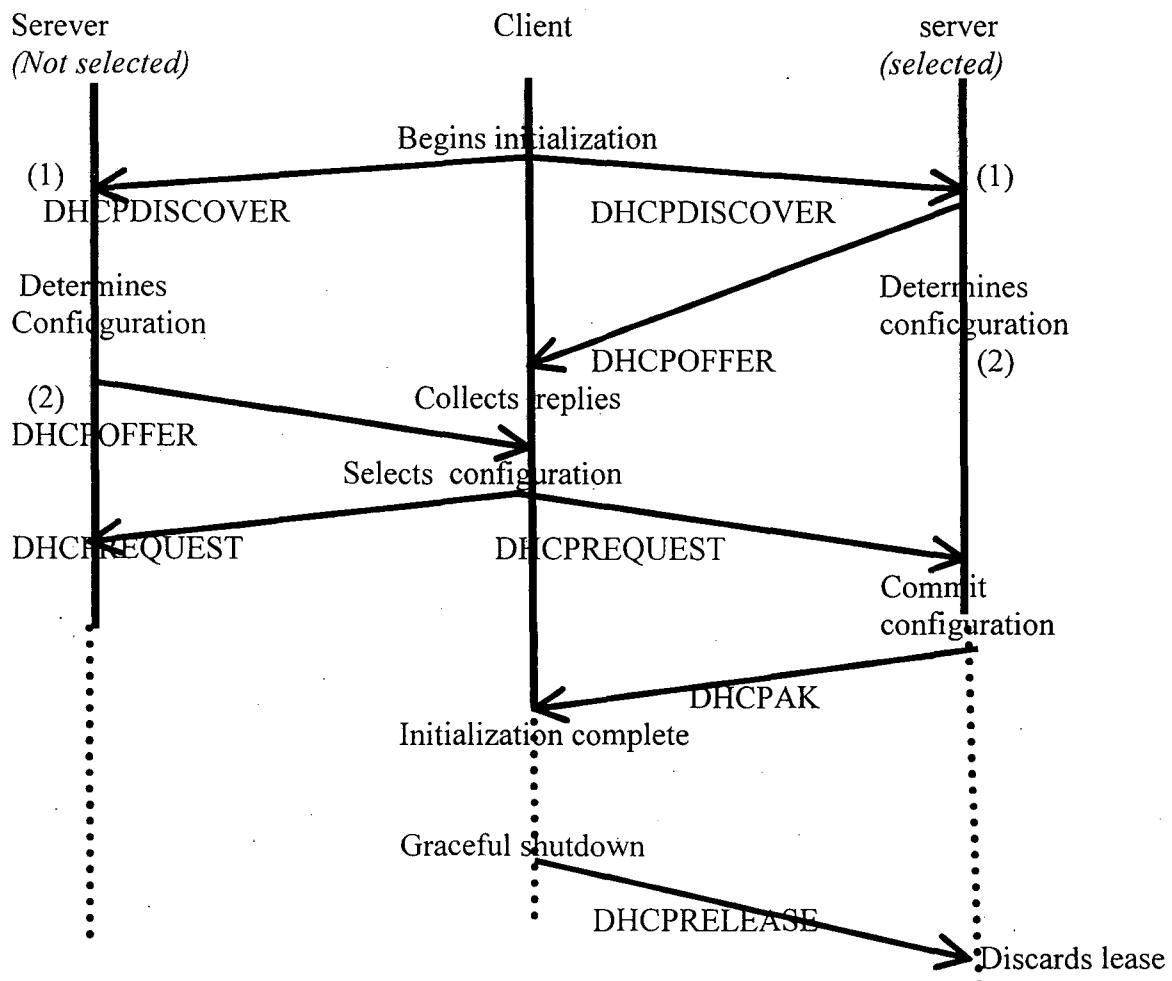


Figure-2: (Initial Address Assignment)

In the second case, the client requests for renewal of the already assigned address. Steps are given below:

- A DHCP client broadcasts a DhcpRequest with *Requested IP address* option is the previously assigned address.
- A DHCP server which has a binding of the address returns DhcpAck or DhcpNak. Remaining are same as the first case.

2.5 DHCP Sever and client interaction

DHCP provides a standards based method of centrally mapping and dynamically setting critical configuration parameters for an IP host on a network. In a working DHCP environment, hosts may be added to the network or moved between networks.

Host that implement DHCP uses the protocol to configure some or all of their IP interface. Each interface is configured independently.

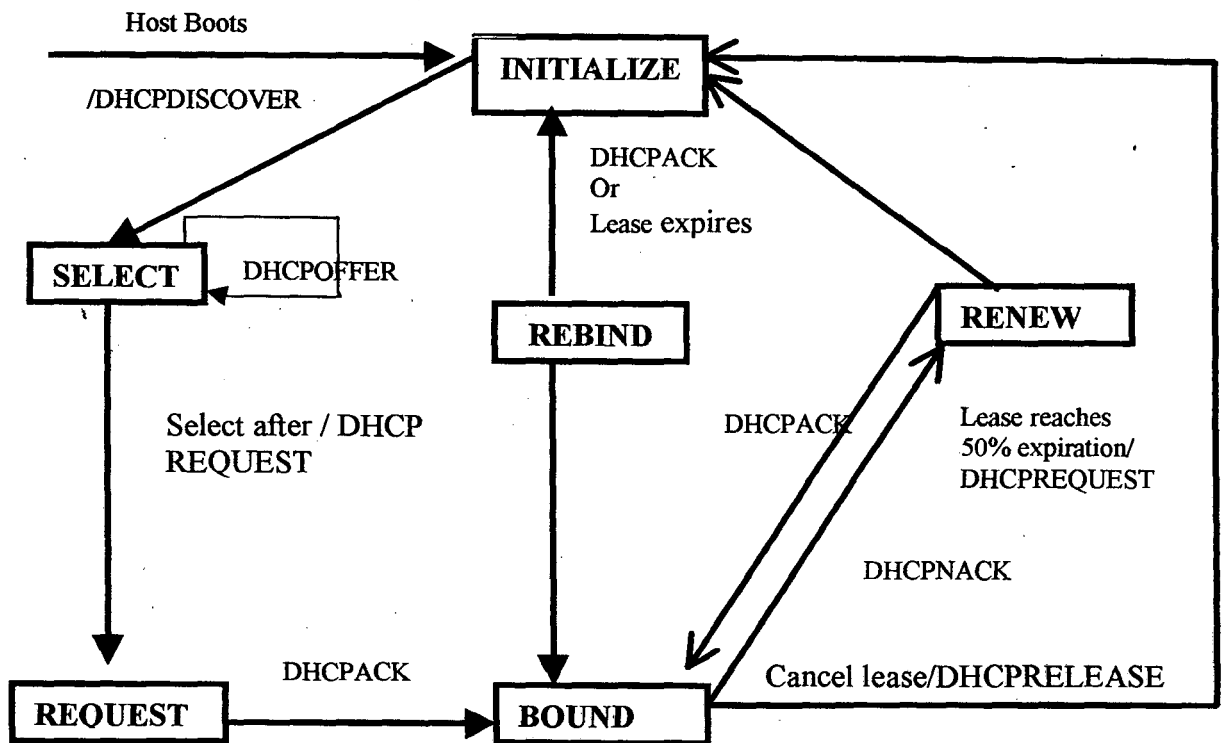


Figure-4: (The six main states of DHCP client and transition among)

2.6 Lease Termination

2.6.1 BOUND States

A client typically remains in the BOUND states while it uses the IP address it has acquired. If a client has secondary storage, the client can store the IP address it was assigned, and request the same address, when it restart again. In some cases, a client in the same BOUND state may discover that it no longer needs in IP address. Example, suppose a user attaches a portable computer to a network, uses DHCP to acquire an IP address, and then uses TCP/IP to read electronic mail. The user may not know how long reading mail will require, or the portable computer may allow the server to choose a lease period. In any case, DHCP specifies a minimum lease period of one hour. If after obtaining an IP

address, the user discovers that no email messages are waiting to be read the user may choose to shutdown the portable computer and move to another location.

When it no longer needs a lease, DHCP allows a client to terminate a lease without waiting for the lease to expire. Such termination is helpful in case where neither the client nor the server can determine appropriate lease duration at the time the lease is granted because it allows a server to choose a reasonably long lease period. Termination is specially important if the number of IP addresses available are much smaller than the number of computers that attach to the network. If each client terminates its lease as soon as the IP address is no longer needed, the server will be able to assign the address to another client.

To terminate a lease, a client sends a DHCPRELEASE message to the server. Releasing an address is a final action that prevents the client from using the address further. Thus, after transmitting the release message, the client must not send any other datagram that use the address. In figure- 4 a host that sends a DHCPRELEASE leaves the BOUND state, and sends a DHCPRELEASE again before it can use IP.

2.6.2 Client state

A DHCP client is in one of the six states at all times. The states are defined by RFC 2131 and determine how the client interacts with the DHCP server, and how the server responds. The states are described below.

INIT : The client has not had a lease and is locating a DHCP server.

INIT-REBOOT : The client has had a lease and is restarting its IP interface.

SELECTING : The client has received lease offers from one or more DHCP servers.

BOUND : The client has been successfully configured and has a valid lease.

RENEWING : The client is attempting to renew the lease with the same DHCP server.

REBINDING : The client could not renew the lease with original server and is attempting to find another.

2.6.3 The DHCP lease

The configuration parameters managed by DHCP are valid for a period of time called a lease. The duration of the lease is set when the client accepts a configuration offer from a DHCP server. The protocol provides a method for a client to request a specific length for the lease terms. The client either accepts or rejects the parameters offered by a server.

2.6.4 Getting the lease

DHCP works with lease times using numbers of seconds as opposed to specific dates and times. This avoids the need to synchronize clocks among multiple hosts on a network.

When a client configures an IP interface using DHCP, three values related to lease time are determined by the server and sent to the client using these options:

- Option 51 IP address lease time
- Option 58 Renewal time value (T1)
- Option 59 Rebinding time value (T2)

The RFC for DHCP (RFC 2131) suggests that the values for T1 and T2 default to 50% and 87.5% of the lease duration, respectively. Some DHCP server implementations only allow you to configure option 51 and derive the other values according to the suggested defaults. On other servers, such as AIX, OS/2 and Shadow IP server, you are able to specify all three options. The RFC mandates that the lease time must be greater than T2, and that T2 must be greater than T1.

Although a client's DHCP implementation might use these values to calculate specific dates and/or times for lease events to occur, conceptually it's easier to think of them as simple counters. The counters are set to the specified number of seconds at the beginning of the lease, and decrement at the rate of one per second.

2.6.6 Renewing the lease

When the T1 counter reaches zero, the client's state changes from BOUND to RENEWING. At this time the client attempts to renew its lease with the same DHCP server that originally provided the configuration. If it is successful, the counter are reset to freshly calculated values. The client now returned to the BOUND state.

If the client is not able to contact the DHCP server, it continue to try periodically, While the client is in the renewing state, no disruption of IP traffic through the interface occurs.

2.6.6 Rebinding the lease

Since T2 must be greater than T1, the T2 counter only expires if the client was not able to renew the lease with the original DHCP server. As a result of this failure, the client's state changes from RENEWING to REBINDING.

At this point, the client tries to get a lease from any DHCP server. If the operation is successful, the counters are set to the values specified in the new lease. The client then returns to the BOUND state.

If the client is unable to contact any DHCP server, it continues to try for the remainder of the current lease. As with the RENEWING state, a REBINDING client is still able to transmit and receive using its IP address with no interruption. The change in state from BOUND to RENEWING to REBINDING is transparent to the applications that are using the IP interface.

2.6.7 Eviction

While you won't find the term eviction in any DHCP RFC, it does accurately describe what happens to a host's IP interface if the lease counter reaches zero. At this point, the lease has expired and the interface can not transmit or receive datagrams using

the previously assigned IP address. Any applications on the host that had been using the interface receive errors.

It is important to note that eviction only occurs when a client has not able to contact any DHCP server on the network and has made multiple attempts over a considerable amount of time.

*** **

Chapter 3

Simple Network Management Protocol

3.1 Introduction

Network management protocol allows a manager to monitor and control routers and Hosts. A network management client program executing on the manager's workstation contact one or more servers, called agents, running on the computers to be controlled. Because an Internet consist of heterogeneous machines and networks, TCP/IP management software executes as application programs and uses Internet transport protocols for communication between clients and servers.

SNMP defines a low- level management protocol that provides two basic operations; fetch a value from a variable or store a value into a variable. In SNMP, all operations occur as side-effects of storing values into variable. The SNMP define the format of message that travel between a manager's computer and managed entity.

The current standard TCP/IP network management protocol is the simple Network Management Protocol. The Simple Network Management Protocol (SNMP) is an application-layer protocol that facilitates the exchange of management information between network devices. It is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite. SNMP enables network administrators to manage network performance, find and solve network problems, and plan for network growth.

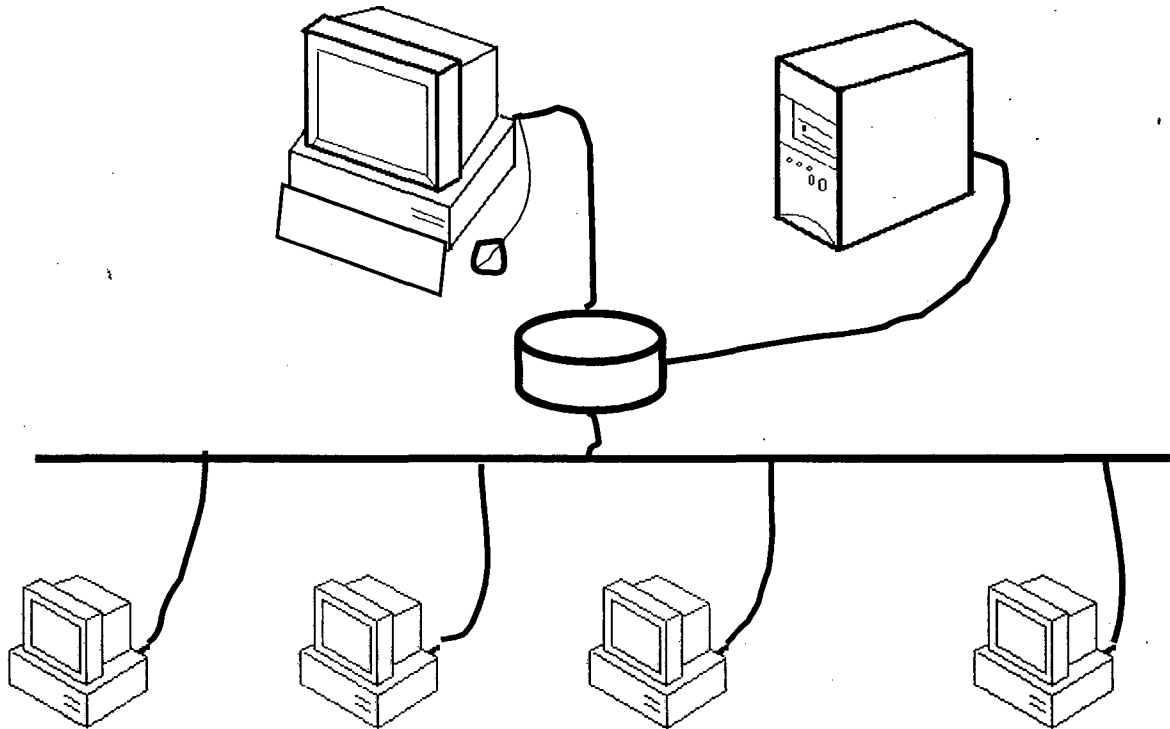


Figure -1 (SNMP facilitates the exchange of network information between devices)

3.2 SNMP Basic Components

An SNMP managed network consists of three key components: managed devices, agents, and network-management systems (NMSs). A managed device is a network node that contains an SNMP agent and resides on a managed network. Managed devices collect and store management information and make this information available to NMSs using SNMP. Managed devices, sometimes called network elements, can be routers and access servers, switches and bridges, hubs, computer hosts, or printers.

An agent is a network-management software module that resides in a managed device. An agent has local knowledge of management information and translates that information into a form compatible with SNMP.

An NMS executes applications that monitor and control managed devices. NMSs provide the bulk of the processing and memory resources required for network management. One or more NMSs must exist on any managed network.

RELATIONSHIP BETWEEN THESE THREE COMPONENTS.

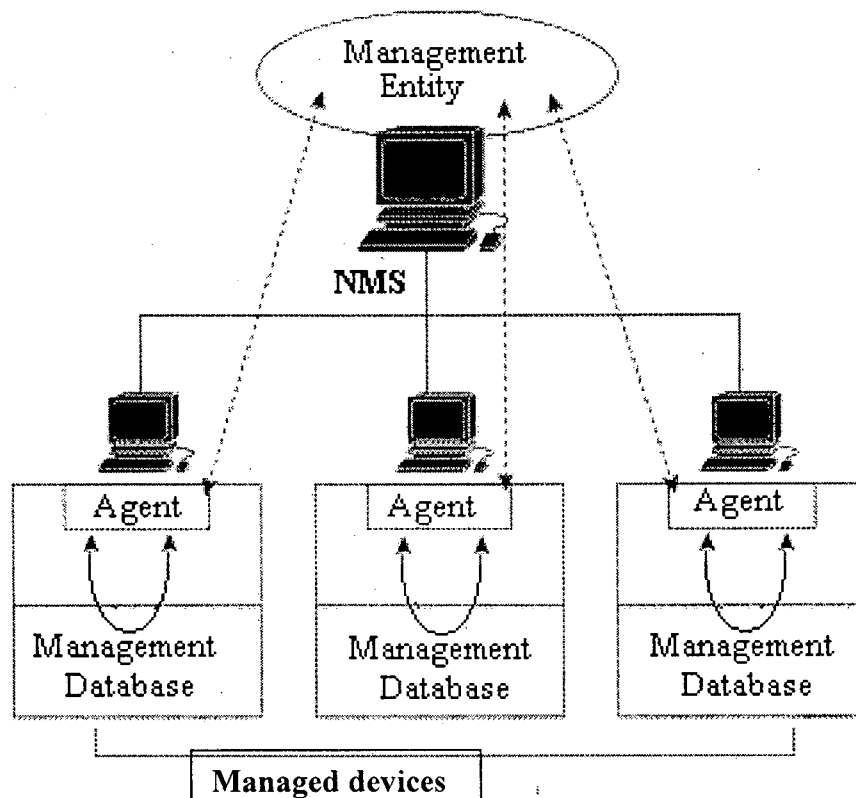


Figure-2 (An SNMP managed network consists of managed devices, agents, and NMSs)

3.3 SNMP Basic Commands

Managed devices are monitored and controlled using four basic SNMP commands: read, write, trap, and traversal operations.

- The read command is used by an NMS to monitor managed devices. The NMS examines different variables that are maintained by managed devices.
- The write command is used by an NMS to control managed devices. The NMS changes the values of variables stored within managed devices.
- The trap command is used by managed devices to asynchronously report events to the NMS.

When certain types of events occur, a managed device sends a trap to the NMS. Traversal operations are used by the NMS to determine which variables a managed device supports and to sequentially gather information in variable tables, such as a routing table.

3.4 SNMP Management Information Base (MIB)

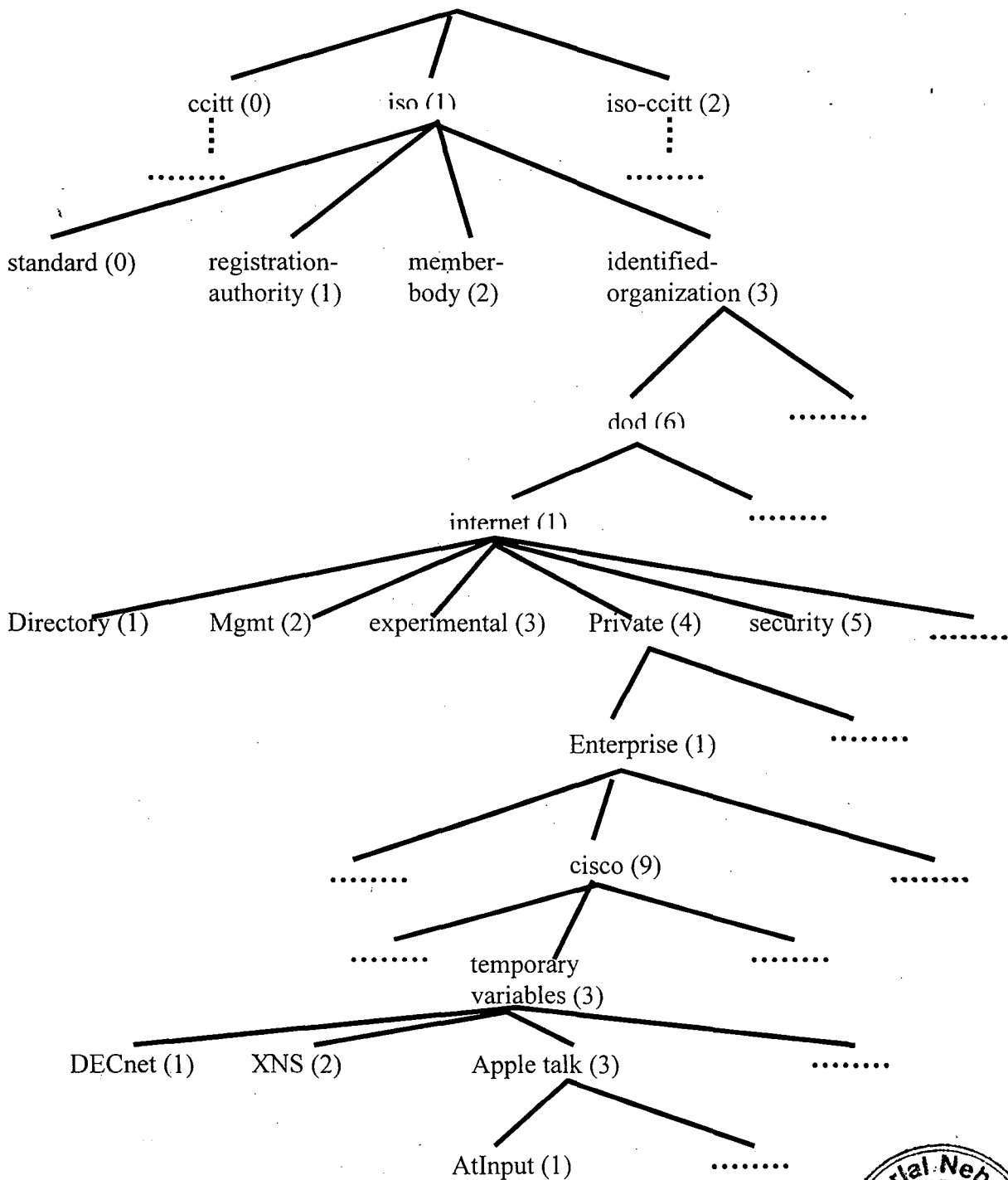
A Management Information Base (MIB) is a collection of information that is organized hierarchically. MIBs are accessed using a network-management protocol such as SNMP. They are comprised of managed objects and are identified by object identifiers. A managed object (sometimes called a MIB object, an object, or a MIB) is one of any number of specific characteristics of a managed device. Managed objects are comprised of one or more object instances, which are essentially variables.

Two types of managed objects exist: scalar and tabular. Scalar objects define a single object instance. Tabular objects define multiple related object instances that are grouped together in MIB tables.

An example of a managed object is at Input, which is a scalar object that contains a single object instance, the integer value that indicates the total number of input AppleTalk packets on a router interface.

An object identifier (or object ID) uniquely identifies a managed object in the MIB hierarchy. The MIB hierarchy can be depicted as a tree with a nameless root, the levels of which are assigned by different organizations.

The MIB tree illustrates the various hierarchies assigned by different organizations.



TH-9347

Figure-3 (MIB Tree)



The top-level MIB object IDs belong to different standards organizations, while lower-level object IDs are allocated by associated organizations. Vendors can define

private branches that include managed objects for their own products. MIBs that have not been standardized typically are positioned in the experimental branch.

The managed object at Input can be uniquely identified either by the object name--
-iso.identified-organization.dod.internet.private.enterprise.cisco.temporaryvariables.
AppleTalk.atInput - or by the equivalent object descriptor: 1.3.6.1.4.1.9.3.3.1.

3.5 Data Representation in SNMP

SNMP must account for and adjust to incompatibilities between managed devices. Different computers use different data-representation techniques, which can compromise the ability of SNMP to exchange information between managed devices. SNMP uses a subset of Abstract Syntax Notation One (ASN.1) to accommodate communication between diverse systems.

3.6 Structure of Management Information in SNMP

The Structure of Management Information (SMI) defines the rules for describing management information, using Abstract Syntax Notation One (ASN.1). The SMI makes three key specifications: ASN.1 data types, SMI-specific data types, and SNMP MIB tables.

3.6.1 ASN1 Data Types

The SNMP SMI specifies that all managed objects have a certain subset of Abstract Syntax Notation One (ASN.1) data types associated with them. Three ASN.1 data types are required: name, syntax, and encoding. The name serves as the object identifier (object ID). The syntax defines the data type of the object (for example, integer or string). The SMI uses a subset of the ASN.1 syntax definitions. The encoding data describes how information associated with a managed object is formatted as a series of data items for transmission over the network.

3.6.2 SMI-Specific Data Types

The SNMP SMI specifies the use of a number of SMI-specific data types, which are divided into two categories: simple data types and application-wide data types.

Three simple data types are defined in the SNMP SMI, all of which are unique values: integers, octet strings, and object IDs. The integer data type is a signed integer in the range of -2,147,483,648 to 2,147,483,647. Octet strings are ordered sequences of zero to 65,535 octets. Object IDs come from the set of all object identifiers allocated according to the rules specified in ASN.1.

Seven application-wide data types exist in the SNMP SMI: network addresses, counters, gauges, time ticks, opaque, integers, and unsigned integers. Network addresses represent an address from a particular protocol family. SNMP supports only 32-bit IP addresses. Counters are nonnegative integers that increase until they reach a maximum value and then return to zero. In SNMP, a 32-bit counter size is specified. Gauges are nonnegative integers that can increase or decrease but retain the maximum value reached. A time tick represents a hundredth of a second since some event. An opaque represents an arbitrary encoding that is used to pass arbitrary information strings that do not conform to the strict data typing used by the SMI. An integer represents signed integer-valued information. This data type redefines the integer data type, which has arbitrary precision in ASN.1 but bounded precision in the SMI. An unsigned integer represents unsigned integer-valued information and is useful when values are always nonnegative. This data type redefines the integer data type, which has arbitrary precision in ASN.1 but bounded precision in the SMI.

3.6.3 SNMP MIB Tables

The SNMP SMI defines highly structured tables that are used to group the instances of a tabular object (that is, an object that contains multiple variables). Tables are composed of zero or more rows, which are indexed in a way that allows SNMP to retrieve or alter an entire row with a single *Get*, *GetNext*, or *Set* command.

3.7 SNMP Protocol Operations

SNMP is a simple request-response protocol. The network-management system issues a request, and managed devices return responses. This behavior is implemented by using one of four protocol operations: *Get*, *GetNext*, *Set*, and *Trap*. The *Get* operation is used by the NMS to retrieve the value of one or more object instances from an agent. If the agent responding to the *Get* operation cannot provide values for all the object instances in a list, it does not provide any values. The *GetNext* operation is used by the NMS to retrieve the value of the next object instance in a table or list within an agent. The *Set* operation is used by the NMS to set the values of object instances within an agent. The *Trap* operation is used by agents to asynchronously inform the NMS of a significant event.

3.8 SNMP Management

Network management protocols specify communication between the network management client program. The Manager invokes and network management server program on a host or router.

SNMP is a distributed-management protocol. A system can operate exclusively as either an NMS or an agent, or it can perform the functions of both. When a system operates as both an NMS and an agent, another NMS might require that the system query managed devices and provides a summary of the information learned, or that it reports locally stored management information. Defining the form and meaning of messages exchanged and the representation of names and values in those messages, network management. Protocol also defines administrative relationship among routes being managed that is, they provide for authentication of managers. One might expect network management protocols to contain a large number of commands. For example, supported commands that allowed the manager to: reboot the system, add or delete routes, disable or enable a particular network interface, or remove cached address bindings

3.10 SNMP Security

SNMP lacks any authentication capabilities, which results in vulnerability to a variety of security threats. These include masquerading, modification of information, message sequence and timing modifications, and disclosure. Masquerading consists of an unauthorized entity attempting to perform management operations by assuming the identity of an authorized management entity. Modification of information involves an unauthorized entity attempting to alter a message generated by an authorized entity so that the message results in unauthorized accounting management or configuration management operations. Message sequence and timing modifications occur when an unauthorized entity reorders, delays, or copies and later replays a message generated by an authorized entity. Disclosure results when an unauthorized entity extracts values stored in managed Objects, or learns of noticeable events by monitoring exchanges between managers and agents. Because SNMP does not implement authentication, many vendors do not implement Set operations, thereby reducing SNMP to a monitoring facility.

3.10 SNMP Reference:

3.10.1 SNMP Message Formats

SNMP messages contain two parts: a message header and a protocol data unit.



Figure. 4 (An SNMP message consists of a header and a PDU)

3.10.2 SNMP Message Header

SNMP message headers contain two fields: Version Number and Community Name. The following descriptions summarize these fields:

- *Version Number*---Specifies the version of SNMP used.

- **Community Name**---Defines an access environment for a group of NMSs. NMSs within the community are said to exist within the same administrative domain. Community names serve as a weak form of authentication because devices that do not know the proper community name are precluded from SNMP operations.

3.10.3 SNMP Protocol Data Unit (PDU)

SNMP PDUs contain a specific command (Get, Set, and so on) and operands that indicate the object instances involved in the transaction. SNMP PDU fields are variable in length, as prescribed by Abstract Syntax Notation One (ASN.1). Figure. 5 illustrates the fields of the SNMP *Get*, *GetNext*, *Response*, and *Set* PDUs transactions.

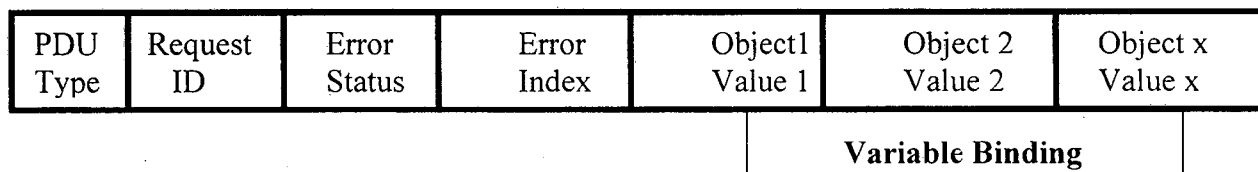


Figure. 5: (SNMP *Get*, *GetNext*, *Response*, and *Set* PDUs contain the same fields)

The following descriptions summarize the fields illustrated in Figure. 5:

PDU Type- Specifies the type of PDU transmitted.

Request ID- Associates SNMP requests with responses.

Error Status- Indicates one of a number of errors and error types. Only the response operation sets this field. Other operations set this field to zero.

Error Index- Associates an error with a particular object instance. Only the response operation sets this field. Other operations set this field to zero.

Variable Bindings- Serves as the data field of the SNMPv1 PDU. Each variable binding associates a particular object instance with its current value (with the exception of Get and GetNext requests, for which the value is ignored).

3.10.4 Trap PDU Format

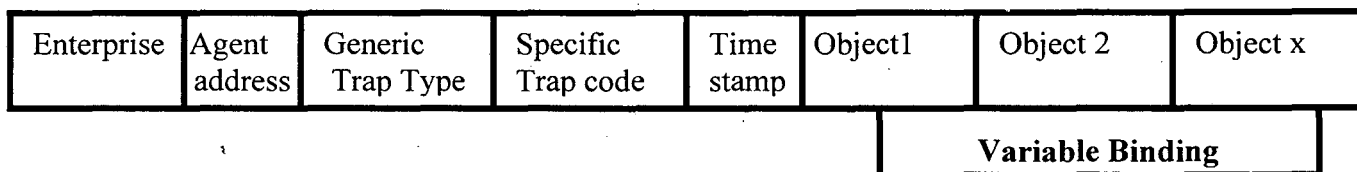


Figure- 6: (The SNMP Trap PDU consists of eight fields)

The following descriptions summarize the fields illustrated in Figure. 6 :

Enterprise- Identifies the type of managed object generating the trap.

Agent Address- Provides the address of the managed object generating the trap.

Generic Trap Type- Indicates one of a number of generic trap types.

Specific Trap Code- Indicates one of a number of specific trap codes.

Time Stamp- Provides the amount of time that has elapsed between the last network re-initialization and generation of the trap.

Variable Bindings-The data field of the SNMP Trap PDU. Each variable binding associates a particular object instance with its current value.

❖ ❖ ❖ ❖ ❖

Chapter 4

Data Structure

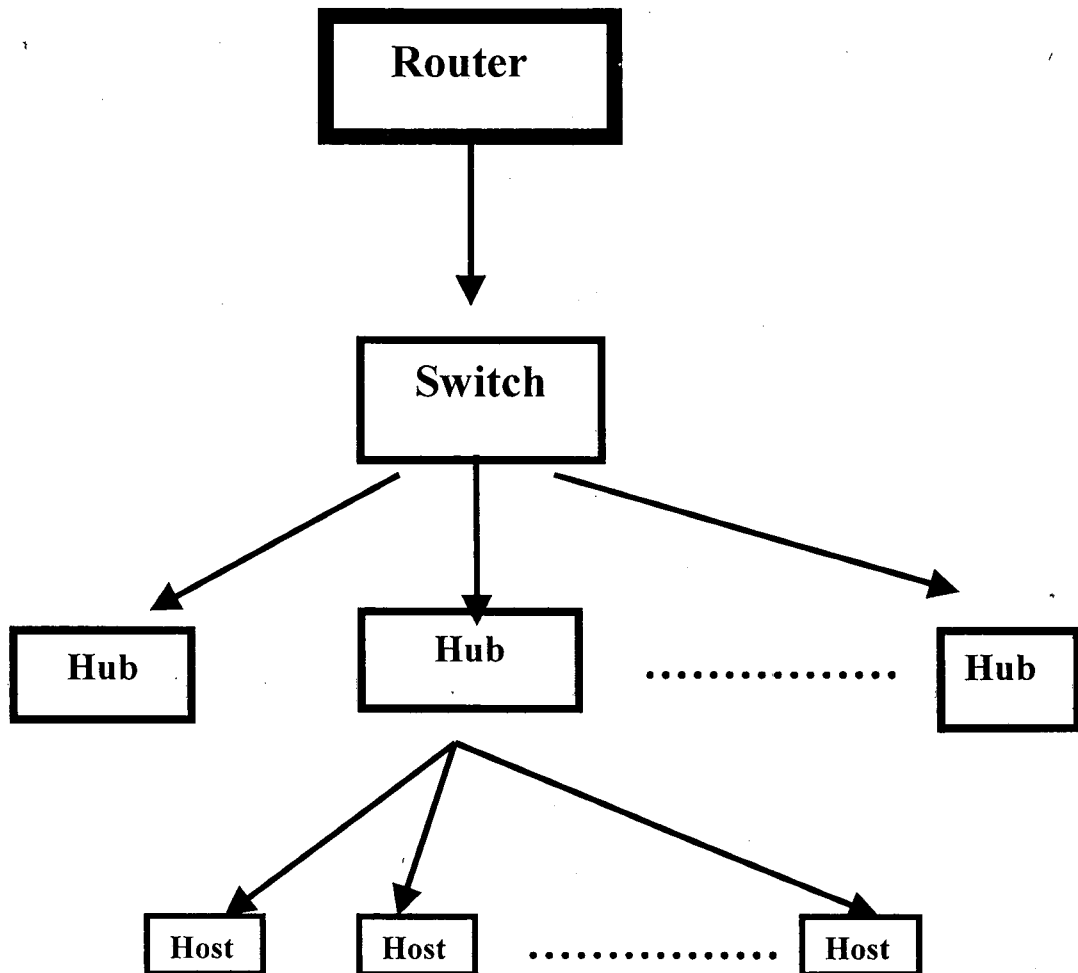


Figure -1 (Structure of the Devices)

4.1 Introduction

In present work three tables are used for finding IP address information of the host. The first table (router table) contains data information of IP addresses and their corresponding MAC addresses. Second table (switch table) is used for finding switch Port number corresponding to the MAC address of the host. This table takes MAC address from the Router table and the port number can be determined from the switch table. The

third table is used for finding Hub information. This table takes port number from the Switch table, and corresponding to that Port number, the Hub identity, Hub unit identity and MAC address of the host, and the hub port are retrieved from the Hub table.

The simple data structure for implementing the table can be an array. But it has the following disadvantages.

- To determine the size of the array is difficult task, because the number of IP address and MAC address differ from network to network.
- Allocating a large amount of memory to an array in advance will result into wastage of memory.

Another solution of this problem is to use a link list. But it also have same disadvantages:

- Some amount of memory is wasted in keeping the links.
- Traversing of the list is needed to access an element.

Therefore, a better solution is used that combines the advantage of both array and the linked list. It also reduces the disadvantages in the former solution.

The data structure used here is shown in the given figure.

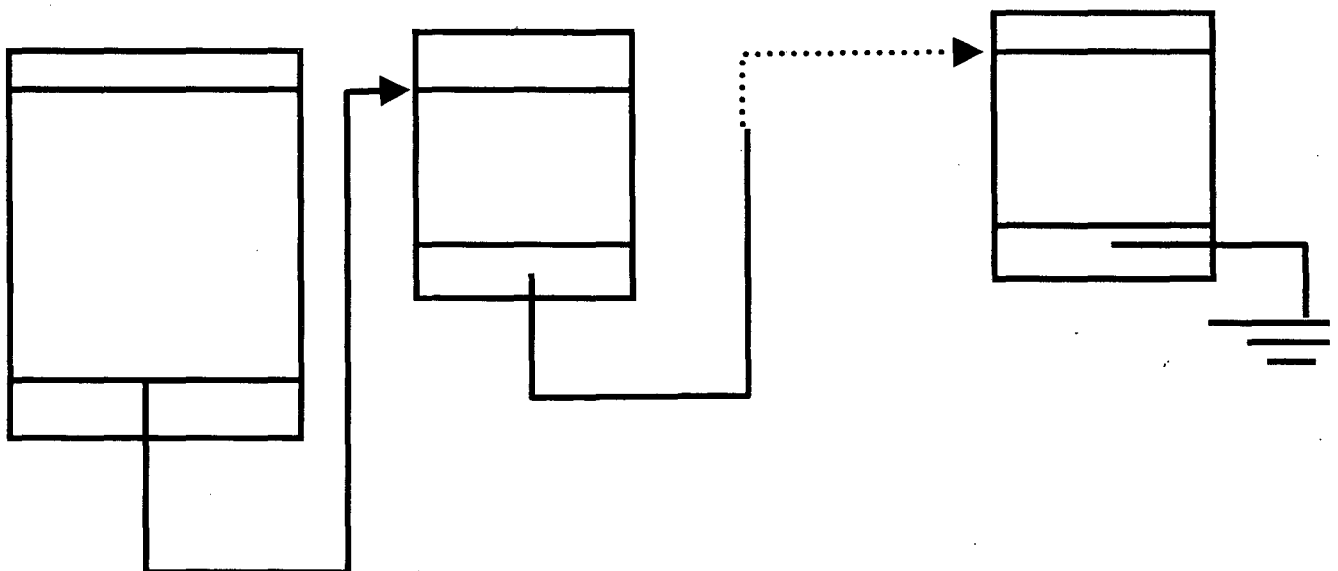


Figure-2 (Layout for proposed table structures)

Here, the data structure is a fixed list of fixed sized blocks. Each block has fixed number of elements and the last element of a block contains a link to another block. As the size of the table varies blocks can be added or removed. To access an element inside a block may require traversing of a few blocks to move to the block containing the specific element. Elements inside a block are accessed directly.

4.2 Data Organization

As mentioned above the data required for finding unauthorized IP address are stored in the tables. These tables are shown below:

- Ip_mac_map_table
- Mac_sw_port_map_table
- Mac_hub_port_mapping
- Switch_port_data_table

4.2.1 Table- 1. Ip_mac_map_table (Table for router information)

This table which exists on the DHCP server is a copy of the IP addresses to MAC

```

    struct ipmacmaptablenode *next_block;
};
typedef struct ipmacmaptablenode Ip_mac_map_table_node;
Ip_mac_map_table_node *ip_mac_map_table;

```

4.2.2 Table- 2. Mac_sw_port_map_table (Table for switch information)

This table also contain two fields for storing data information. i.e. mapping of the MAC address of host and Port number of the switch.

Table-2

```

struct macswportmapping
{
    Macaddr macaddr;
    Portno portno;
};

typedef struct macswportmapping Mac_sw_port_mapping;

struct macswportmaptablenode
{
    Mac_sw_port_mapping
    mac_sw_port_mapping[MAX_ENTRY_PER_MAC_SW_PORT_BLOCK];
    struct macswportmaptablenode *next_block;
};

typedef struct macswportmaptablenode Mac_sw_port_map_table_node;

Mac_sw_port_map_table_node *mac_sw_port_map_table;

```

4.2.3 Table- 3. Mac_hub_port_mapping (Table for Hub information)

This table contain the mapping of Hub Id, Hub unit number, Hub port number, for the MAC addresses of the hosts.

Table-3

```
struct machubportmapping
{
    Macaddr macaddr;
    Hubid hubid;
    Unitid unitid;
    Portno portno;
};

typedef struct machubportmapping Mac_hub_port_mapping;

struct machubportmaptablenode
{
    Mac_hub_port_mapping
    mac_hub_port_mapping[MAX_ENTRY_PER_MAC_HUB_PORT_BLOCK];
    struct machubportmaptablenode *next_block;
};

typedef struct machubportmaptablenode Mac_hub_port_map_table_node;
```

4.2.4 Table- 4 Switch_port_data_table (Table for Host information)

This table contains two fields, Port no and a pointer to a hub table corresponding switch port number. This table used for finding host information from the hub table. This table is shown below:

Table- 4

```
struct swportdata
{
    Portno sw_port_no;
    Mac_hub_port_map_table_node *mac_hub_port_map_table;
};

typedef swportdata Sw_port_data;
```

```

struct swportdatablock
{
    Sw_port_data      sw_port_data[MAX_SW_PORT_PER_BLOCK];
    struct swportdatablock *next_block;
};
typedef struct swportdatablock Sw_port_data_block;

Sw_port_data_block *switch_port_data_table;

```

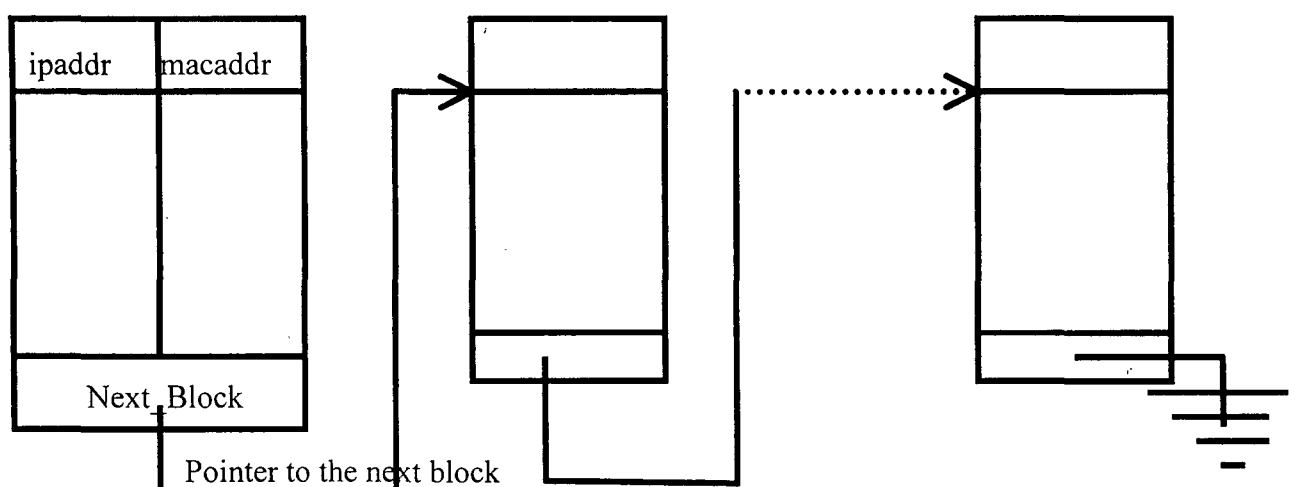
4.3 Organisation of Ip mac map table

This table capture the IP address and corresponding MAC address at router level. The DHCP allocates the IP address to the host and the hosts using the IP address communicate. This table keeps the IP address as well as the corresponding MAC address. Given an IP address the corresponding MAC address can be found out from the table.

This is a dynamic data structure. It is design as a linked list of a number of blocks. Each block is a set of IP address and MAC address pair.

The addr This filed contains the IP address of the host.

Macaddr This field contain the MAC address of the host corresponding to the IP address.



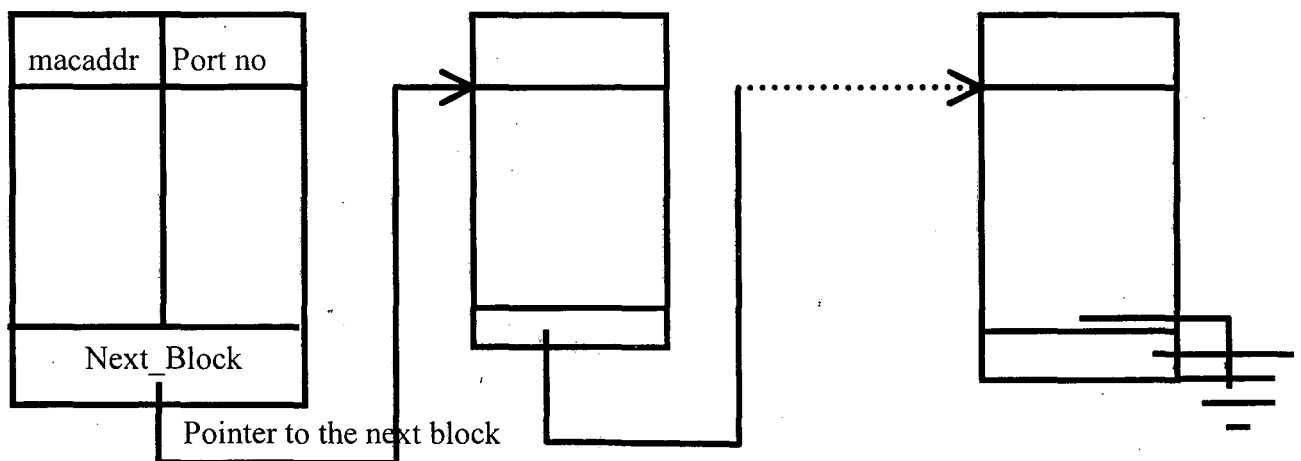
Layout of the Ip_mac_map_table(for router information)

4.4 Organisation of mac sw port map table

This is the data structure for switch level information. Once the MAC address corresponding to some IP address is found out, it is the switch to tell, at which port the same MAC address is there. It is also a dynamic data structure. It is a linked list of Blocks. Each block is a set of macaddress and port no. from this table. The port no. corresponding to the given MAC address is found out using the port no. of the switch. Then the Hub to which the host (MAC address) is attached can be found out. The fields in the table are:

Macaddr : This field contains the MAC address of the host.

Port no. : This tells the port no. of the switch on which the host is attached.



Layout of the mac_sw_port_map_table

4.5 Organisation of the mc hub port map table node

The hub level information are stores in this table. From table- 2 the port no. and the Mac Address is known corresponding to port no. the Mac address corresponding hub or hub unit at which the host resides is localized.

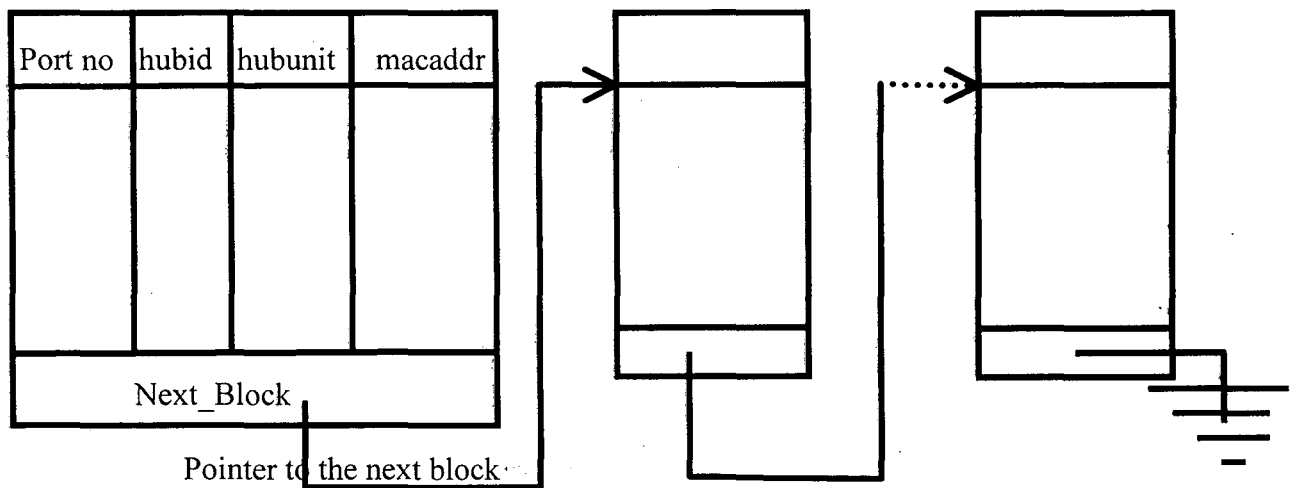
A dynamic data structure is implemented as a linked list. This is a list of blocks. Each block is a set of following variables:

Port no. : This tells the port no. of the hub the host is attached to

Hubid: This field contains the hub id of the hub corresponding to the Mac address of the host.

Hubunit: This field contains the hub unit of the hub corresponding to the Mac address of the Host.

Macaddr This field contain the MAC address of the host.



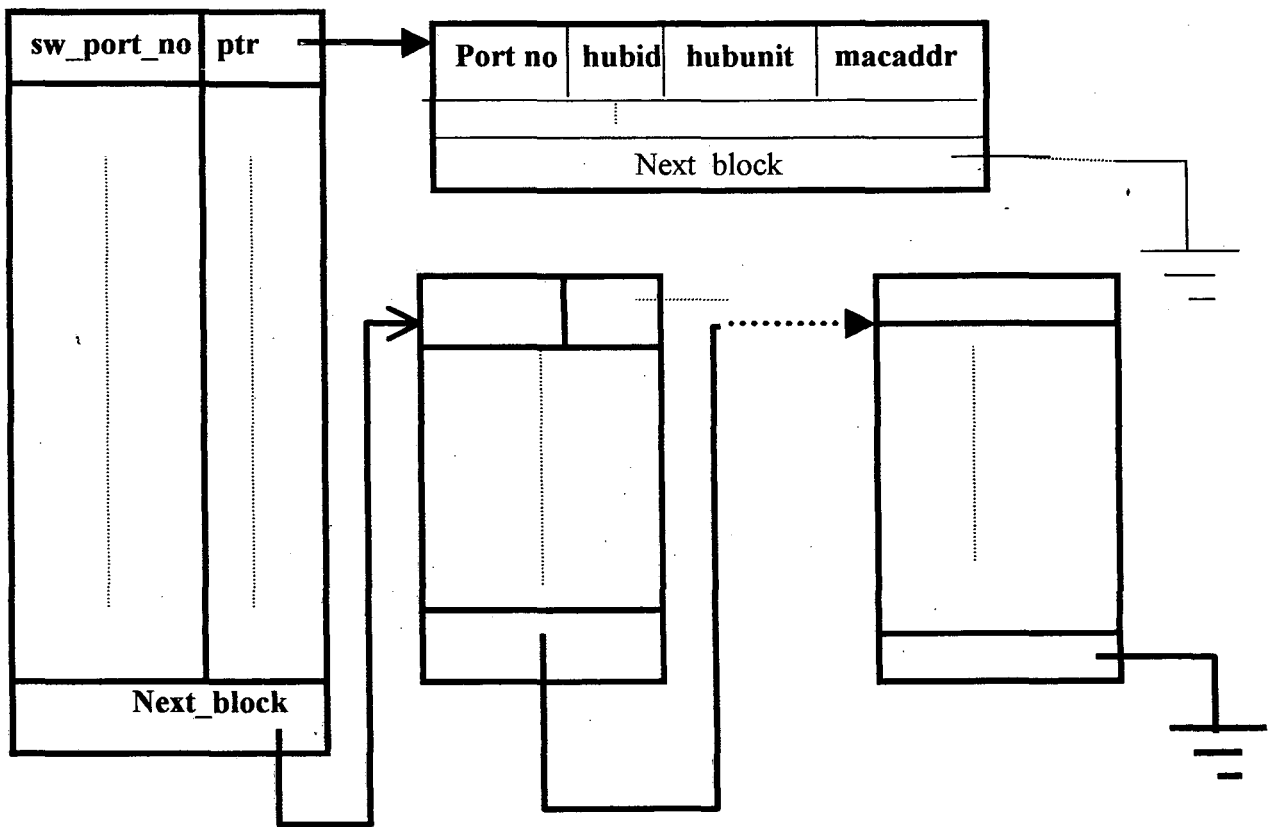
Layout of the `mc_hub_port_map_table_node`.(for Hub information)

4.6 Organisation of the switch port data table

Corresponding to switch port no. the hub can be known. The hub id and the mac address can locate the hub port at which the host is situated. This table contains two fields:

Sw_port_no: This field is a port no.

Macaddr: This field has a macaddress of the host.



Layout of the `switch_port_data_table`

*** **

Chapter 5

Algorithms

5.1 Algorithm For Finding MAC Address

Given the Ip address this algorithms finds the corresponding Mac address from the Ip_mac_map_table.

```
int find_mac_address(ip_addr,mac_addr_ptr)
    Ipaddr  ip_addr;
    Macaddr *mac_addr_ptr;
{
    int i;
    Ip_mac_map_table_node *temp_ptr;
    temp_ptr = ip_mac_map_table;
    while(temp_ptr != NULL)
    {
        i = 0;
        while(i < MAX_ENTRY_PER_IP_MAC_BLOCK)
        {
            if(ip_addr == temp_ptr->ip_mac_mapping[i].ipaddr)
            {
                *mac_addr_ptr = temp_ptr->ip_mac_mapping[i].macaddr;
                return(TRUE);
            }
            i++;
        }
        temp_ptr = temp_ptr->next_block;
    }
    return(FALSE);
}
```

5.2 Algorithm to finding Switch information

Given the Mac address this algorithms provide the corresponding information about the switch port no.

```
int find_sw_port_address(mac_addr,sw_port_no_ptr)
    Macaddr mac_addr;
    Portno *sw_port_no_ptr;
{
    int i;
    Mac_sw_port_map_table_node *temp_ptr;
    temp_ptr = mac_sw_port_map_table;
    while(temp_ptr != NULL)
    {
        i = 0;
        while(i < MAX_ENTRY_PER_MAC_SW_PORT_BLOCK)
        {
            if(mac_addr == temp_ptr->mac_sw_port_mapping[i].macaddr)
            {
                *sw_port_no_ptr = temp_ptr->mac_sw_port_mapping[i].portno;
                return(TRUE);
            }
            i++;
        }
        temp_ptr = temp_ptr->next_block;
    }
    return(FALSE);
}
```

5.3 Algorithm for finding hub information

Given the Mac address and switch port no. this algorithms provides the corresponding hub information i.e. hubid, unitid in case the hub is cascaded and port no. of that host (of which the Mac addresses is given) on this hub.

```
int find_hub_parameters(mac_addr, sw_port_no, unit_id_ptr, hub_id_ptr,
hub_port_no_ptr)

Macaddr mac_addr;
Portno sw_port_no;
Unitid *unit_id_ptr;
Hubid *hub_id_ptr;
Portno *hub_port_no_ptr;
{
int i;
Sw_port_data_block *temp_ptr;
temp_ptr = switch_port_data_table;
while(temp_ptr != NULL)
{
i = 0;
while(i < MAX_SW_PORT_PER_BLOCK)
{
if( sw_port_no == temp_ptr->sw_port_data[i].sw_port_no)
{
temp_ptr1 = temp_ptr->sw_port_data[i].mac_hub_port_map_table;
while(temp_ptr1 != NULL)
{
j = 0;
while(j < MAX_ENTRY_PER_MAC_HUB_PORT_BLOCK)
{
if(mac_addr == temp_ptr1->mac_hub_port_mapping[j].macaddr)
{
```



```

        *unit_id_ptr = temp_ptr1->mac_hub_port_mapping[j].unitid;
        *hub_id_ptr = temp_ptr1->mac_hub_port_mapping[j].hubid;
        *hub_port_no_ptr = temp_ptr1->mac_hub_port_mapping[j].portno;
        return(TRUE);
    }
    j++;
}
temp_ptr1 = temp_ptr1->next_block;
}
}
i++;
}
temp_ptr = temp_ptr->next_block;
}
return(FALSE);
}

```

5.4 Algorithm for finding rogue host

Given the IP address this algorithm provides all the information required to reach that host. This information contains the switch port no. the hub_id, the unit_id in case the hub is cascaded and the port no. on that hub.

```
int find_rogue_host( ip_address,sw_port_no_ptr,hub_id_ptr,unit_id_ptr,hub_port_no_ptr)
```

```

Ipaddr ip_address;
Portno *sw_port_no_ptr;
Portno *hub_port_no_ptr;
Unitid *unit_id_ptr;
Hubid *hub_id;
{
    Macaddr mac_address;
    if(find_mac_address(ip_address,&mac_address))

```

```

{
    if(find_sw_port_address(mac_address, sw_port_no_ptr))
    {
        if(find_hub_parameters(mac_address,*sw_port_no_ptr,unit_id_ptr,hub_id_ptr,hub_port_no_ptr));
        {
            return(TRUE);
        }
        else return(FALSE);
    }
    else return(FALSE);
}
else return(FALSE);
}
}

```

5.5 Algorithm for IP allocation

This algorithm allocate an IP address to a host from the available address pool.

```
int ip_allocation process()
```

```

{
    Ipaddr ip_address;
    Portno sw_port_no;
    Hubid hub_id;
    Unitid unit_id;
    Portno hub_port_no;

```

step 1 find free ip address from local table

step 2 use ARP to see whether the IP has been allocated or not;

step 3 if(ARP response arrives)/* then it implies that status of the IP address has been wrongly updated*/

```

{
    if(find_rogue_host(ip_address,&sw_port_no,&hub_id,&unit_id,&hub_port_no))

```

```

    {
        block_host( sw_port_no, hub_id, unit_id, hub_port_no);
    }
    else update_tables();

    if(find_rogue_host(ip_address,&sw_port_no,&hub_id,&unit_id,&hub_port_no))
    {
        block_host( sw_port_no, hub_id, unit_id, hub_port_no);
    }
    step 4 Allocate the IP and send response to client;
}

```

5.6 Algorithm For Update Data Information

This algorithm is used to update the data in tables used in the present work.

```

int update_tables(one_stretch)
{
    get next ip_mac mapping pair in router;
    search entry in table1
    if entry not found in table 1;
    {
        insert entry in table 1;
        get mac, switch port no mapping from switch;
        update table2;
        get mac, hub_id mapping from <network???\>
        get mac, unit_id from <network???\>
        get mac, portno mapping from hub;
        update table4()
    }
    else
    {

```

```

get mac, switch port no mapping from switch;
if entry is not found in table2
{
    get mac, hub_id mapping from <network????>
    get mac, unit_id from <network??>
    get mac, portno mapping from hub;
    update table 4;
    else
    {

        get mac, hub_id mapping from <network????>
        get mac, unit_id from <network??>
        get mac, portno mapping from hub;
        if entry not found in table 4
        {
            update table3()
        }
    }
}
}

```

5.7 Using updation algorithm

This algorithm will be used in the following situations:

- 1) at the idle time of network. (one_stretch == FALSE)
- 2) problem in finding rogue host. (one_stretch == TRUE)
- 3) This algo will not be completed in one step
but in case (2) it will be done in one step.

*** **

Chapter 6

Conclusion

The present work made an attempt to improve the performance of a network by controlling the DHCP IP allocation with the emphasis on blocking the rogue hosts. This work has been carried out with a case study done for the JNU campus wide area network. But this can work conveniently for other similar networks. Some of salient features of the work are given below:

- The server will disallow “Rogue” hosts
- Optimized polling interval
- No manual intervention
- The solution is application specific
- Table updation during leisure time.

Though the various data structures have been designed, they can further be improved to optimize the space and efficiency. Due to the time constraints the work could not be fully implemented. With the comprehensive data structures and the algorithms suggested, the work can easily be implemented in the language suitable to the implementers.

Since the work has been designed to provide application specific solution, it can easily be enhanced to give a standard solution.

- Efforts for standard solution can be made



Reference

1. Andrew S. Tanenbaum, "Computer Networks," Third Edition, Prentice Hall of India, 1999.
2. Case J., "A Simple Network Management Protocol," Request for Comments: 1157, May 1990.
3. Douglas E. Comer, "Internetworking With TCP/IP," Vol. 1: Principles, Protocols, and Architecture, Third Edition, Prentice Hall of India, 1999.
4. Douglas E. Comer, "Computer Networks And Internets," Second Edition, Prentice Hall of India, 2000.
5. Droms R., "Dynamic Host Configuration Protocol", Request for Comments: 2131, March 1997.
6. Srinivasan Cheenu, "Multiple-Protocol Label Switching Router MIB," Internet Draft-Draft-ietf-mpls-lsr-mib-00.txt, Jan 1999.
7. Stallings William, "Data and Computer Communications," Fifth Edition, Prentice Hall of India, July 1998.
8. W. Richard Stevens, "TCP/IP Illustrated," Volume 1: The Protocols, Addison-Wesley, 2000.

❖ ❖ ❖ ❖ ❖