UNDERSTANDING AND USING COMPUTER NETWORKS

Uday Pabrai

System Integration Support/ACCESS, Fermilab

Cary Dowat

Internal Networking/Data Communications, Fermilab

Janet Carlson

System Integration Support/ACCESS, Fermilab

Fermi National Accelerator Laboratory

Batavia, Illinois

G. Drzewiński

Computing Division Document Number: GG0009

Preface

In the 1970's a few proprietory network architectures existed. In the 80's we witnessed a period where a number of standards were defined for open systems computing. The 80's was also a period where we were beginning to see a migration away from centralized computing that consisted of mainframes and supercomputers to regional data and computer centers that also included high-performance workstation farms. This migration was made possible mainly by the growth in network and computing technology.

Most computer manufacturers today are migrating from their proprietory network architectures to a series of International Standards for network architectures. These standards are known as the Open Systems Interconnect Reference Model (OSI/RM). Today even the smallest of organizations is dependent on the network for a number of transactions. Be it file transfer, remote login, or electronic mail, the end-user is increasingly interacting with the underlying network. The structure of the text is as follows:

Part 1: Introduction
Chapter 1 provides information on local area networks at Fermilab and also describes all the wide area networks that can be accessed from Fermilab. Chapter 2 introduces the OSI/RM and explains concepts and terminology used in the text.

Part 2: Network Technologies
Chapter 3 describes Ethernet. Chapter 4 studies the emerging metropolitan area network technology, FDDI.

Part 3: Architectures, Protocols, & Gateways
This part provides information on the Internet (DARPA) network architecture, DIGITAL Network Architecture (DNA), DECnet, and BITNET. Interlink's VM-DECnet gateway is described and protocols explained include NFS, DFS, DNS, and packet-switched network protocols such as X.25.

Part 4: PC Networks & Protocols
This part describes KERMIT and Appletalk.

Part 5: Security
Chapter 17 treats the newly defined ISO security architecture. Chapter 18 studies security issues as they relate to UNIX systems.

A number of chapters are titled either *Understanding ...* or *Using ....* Readers are encouraged to read the chapters titled *Understanding ...* to get more infor-

mation on network architectures, protocols, and technology. Chapters titled *Using ...* provide information on network transactions such as electronic mail, remote login, file transfer, and other end-user network applications.

For example, *Chapter 5: Understanding TCP/IP* provides information on the Internet (DARPA) architecture and protocols such as TCP, IP, UDP, SMTP, and ARP. It also describes BIND, subnetting, domain name servers and Internet gateways. This is in contrast to *Chapter 6: Using TCP/IP* which provides information on end-user network commands and applications such as TELNET, FTP, RLOGIN, and RSHELL. This chapter further explains usage of these and other commands for operating systems such as UNIX, VMS, and VM.

I would like to thank the following individuals for their comments and suggestions: Judy Nicholls, Mark Leininger, Roy Thatcher, Jack Pfister, Lauri Loebel, Joy Hathaway, and Mike Isely. I would also like to thank Maribel Jaquez for helping with diagrams and figures. I am grateful to the Fermilab Data Communications Group especially Mark Kaletka, Phil Demar, Reggie Gibbons, Darryl Wohlt, and Vyto Grigaliunas. I would also like to thank William Lidinsky, Mohnish Pabrai, and Jyoti Plaha for their encouragement and support.

Last but not the least, I would like to thank my wife, Tina - for being very understanding for all those weekends and evenings that I was away from home working on this text. I am very grateful to my parents, OmPrakash and Shakuntala, and my wife Tina for always being a source of inspiration and love.

Uday Pabrai

Dedication

To my parents, OmPrakash and Shakuntala, and my wife, Tina - for their inspiration, encouragement, love and support.

U.P.

To my mother, Winifred.

C.D.

To my parents, Laurie and Kathy, and the rest of my family, especially Mark - for their patience, tolerance and encouragement.

J.C.

This Page Intentionally Left Blank

Table of Contents

Part I: INTRODUCTION

Part II: NETWORK TECHNOLOGIES

Part III: ARCHITECTURES, PROTOCOLS, AND GATEWAYS

Part V: SECURITY

List of Figures

List of Tables

This Page Intentionally Left Blank

Part I
INTRODUCTION

This Page Intentionally Left Blank

Chapter 1

Fermilab and Networking

Computer networking has changed enormously over the past decade and continues to do so. Computers ranging from personal computers to supercomputers are more likely to be a part of a network than not. Worldwide electronic mail is a daily reality for millions of people. Networks have become an essential tool for users of all kinds.

The Fermilab user community is very dependent upon networks and networking concepts. This chapter is meant to provide a brief overview of the key systems at Fermilab along with a brief description of the local (on-site) network. The last section covers briefly the wide area networking possibilities available from Fermilab.

1.1. Key Systems

The following list is an overview of the key systems here on-site at Fermilab.

FNBIT: Provides access to BITNET.

FNGATE: DECnet to ULTRIX gateway.

FNMFE: Provides access to MFEnet.

TCP/IP route (default)
is ESnet but packets can go many of several different directions.

CISC1.FNAL.GOV:
This is called the "firewall router". This router can shield the on-site network from external networks if required.

IBM/AMDAHL IBM and AMDAHL front-ends provide 3270 and ASCII ports for up to 500 terminals and printers (primarily in the Hi-Rise) while Interlink and BTI provide DECnet and TCP/IP access, respectively, over Ethernet for the AMDAHL.

1.2. Local Network Description

The local network at Fermilab is very impressive. The total length of cable is in excess of 10 Km and may even approach 15 Km. Four different types of Ethernet media are used:

- Standard
- ThinWire
- Twisted pair
- Fiber

The local network is comprised of approximately 40 Ethernets. Each of these Ethernets is attached to the local network by means of a bridge or router. There are about 37 bridges and 2 routers (used to connect networks instead of bridges) on site. The bridge topology is shown in Figure 5-10 in the *Understanding TCP/IP* chapter. The total number of nodes on the local network is approximately 800. The Ethernet topology of the on-site network can be found in the *Understanding Ethernet* chapter of this document.

System Type	%
DEC	82
UNIX	8
non-DEC terminal servers	8
PC/MAC	5
all other	1

Table 1-1: Node System Type Distribution

1.2.1. Traffic Statistics

Long-term average packet "source" rates vary across Ethernets, ranging from 480 packets per second (pps) on FNALD to less than 100 pps on many Ethernets. The following list contains the long-term average of packets "sourced" on key Ethernets:

FNALD	480 pps
FNAL	150 pps
FCC "HUB"	240 pps
High Rise	200 pps
D0	30 pps
CDF	70 pps

In the next year, the VMS cluster has plans to migrate to FDDI. FDDI is a standard, commercially-available technology with widening vendor support. It is designed specifically for robustness and reliability in a LAN environment. Fermilab is extremely well positioned for FDDI. Fiber is already installed and a backbone network can and will be installed quickly. Figure 4-3 in the *Understanding FDDi* chapter shows the FDDI topology at Fermilab.

1.3. Wide Area Network Description

In addition to its complex, yet highly function local network, Fermilab is a member of other remote networks. At present, there are five T1 links providing highspeed connectivity to other networks. The major networks that are reachable directly from Fermilab are listed below.

- Energy Sciences NETwork (ESnet)
- BITNET
- High Energy Physics NETwork (HEPnet)
- Magnetic Fusion Energy NETwork (MFEnet)

Many more networks are reachable from Fermilab, but Fermilab is only directly connected to those listed above. For example, Fermilab's access to the National Science Foundation NETwork (NSFnet) is through CICnet. NSFnet can also be reached through 4 different connections on ESnet. X.25 connectivity is also provided to Fermilab network users.

The port selector system includes 6 major and 5 minor switching nodes, connected by 14 T1 links over twisted pair, broadband and fiber links. The system provides ports for over 6000 asynch lines (terminal + computer) at very low cost. The *Understanding HEPnet* chapter contains a figure displaying the ESnet backbone T1 connections. Figure 1-1 shows some of the connections accessible through Fermilab.

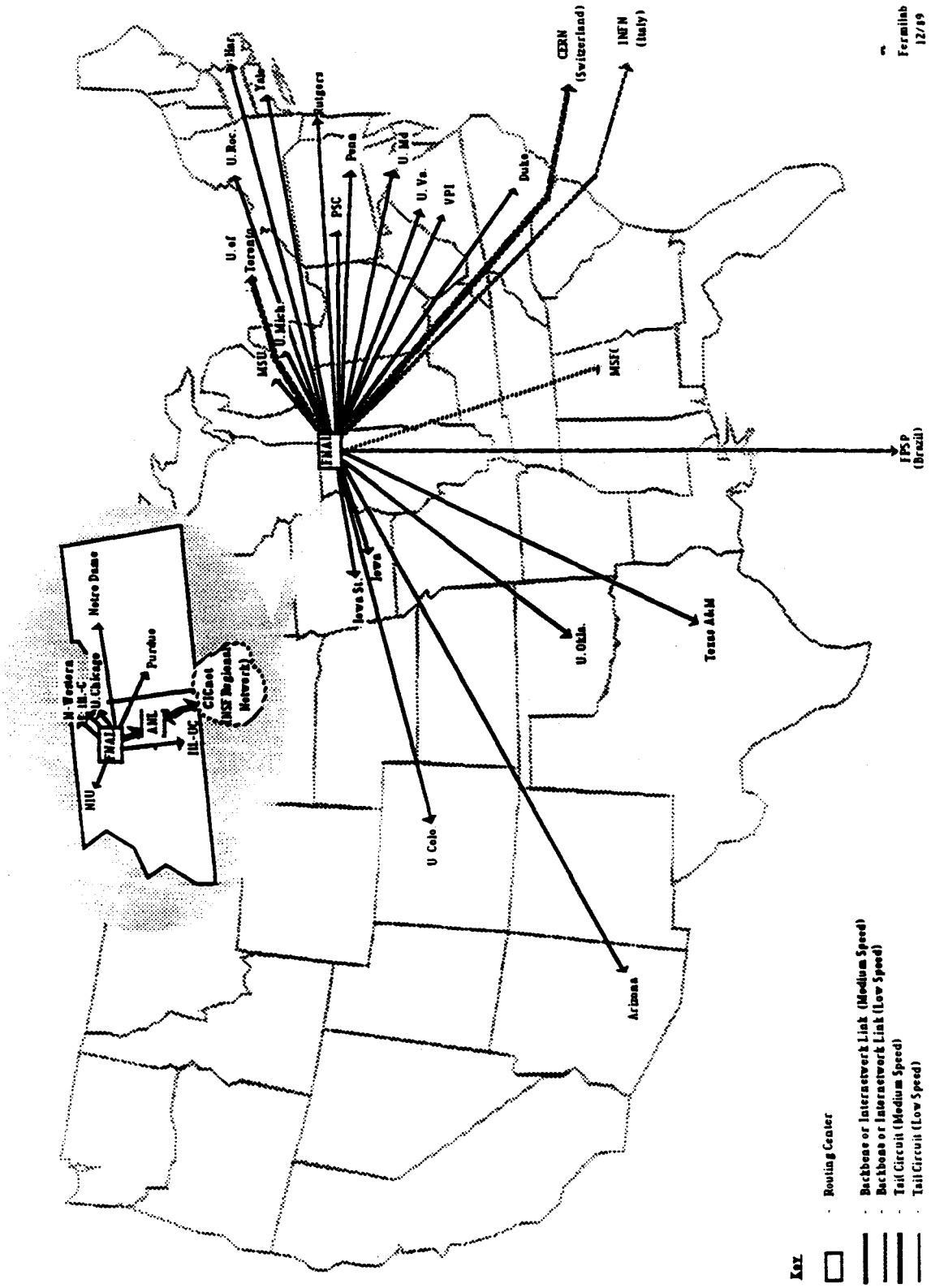


Figure 1-1: Off-site Networking Connections thru Fermilab

Chapter 2

The OSI Reference Model and Terminology

When communication is desired among heterogeneous (different vendors, different models of same vendor) machines, the software development can be a nightmare. Different vendors use different data formats and data exchange conventions. The only acceptable alternative is to implement a common set of conventions. For this to happen standards must be developed and adopted by appropriate organizations. Standards have the following effects:

- Vendors feel encouraged to implement the standards because of an expectation that, because of wide usage of the standards, their products would be less marketable without them.
- Customers are in a position to require that the standards be implemented by any vendor wishing to propose equipment to them.

Before standards can be developed, there should be a structure or architecture that defines the communications tasks. In 1977 the International Organization for Standardizations (ISO) established a subcommittee to develop such an architecture. The result was the Open Systems Interconnection (OSI) Reference Model, which is henceforth referred to as OSI/RM. It is a framework for defining standards for linking heterogeneous computers. The OSI model provides the basis for connecting open systems for distributed applications processing. Also, the OSI/RM is the basis for comparison of protocols and network architectures. The term open denotes the ability of any two systems conforming to the reference model and the associated standards to connect.

2.1. Concepts of a Layered Architecture

A widely accepted structuring technique, and the one chosen by ISO, is layering. The layers reduce the design complexity of a network. The communications functions are partitioned into a vertical set of layers. Each layer performs a related subset of the functions required to communicate with another system. It relies on the next lower layer to perform more primitive functions and to conceal the details of those functions. It provides services to the next higher layer. Changes

in one layer should not require changes in the other layers. The purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how the offered services are actually implemented.

The task of the ISO subcommittee was to define a set of layers and the services performed by each layer. The resulting OSI reference model has seven layers, which are listed with a brief definition in Table 2-1.

Layer	Definition
1. Physical	Concerned with transmission of unstructured bit stream over physical link; involves such parameters as signal voltage swing and bit duration; deals with the mechanical, electrical, and procedural characteristics to establish, maintain, and deactivate the physical link.
2. Data link	Provides for the reliable transfer of data across the physical link; sends blocks of data (frames) with the necessary synchronisation, error control, and flow control.
3. Network	Provides upper layers with independence from the data transmission and switching technologies used to connect systems; responsible for establishing, maintaining, and terminating connections.
4. Transport	Provides reliable, transparent transfer of data between end points; provides end-to-end error recovery and flow control.
5. Session	Provides the control structure for communication between applications; establishes, manages, and terminates connections (sessions) between cooperating applications.
6. Presentation	Performs generally useful transformations on data to provide a standardized application interface and to provide common communications services; examples: encryption, text compression, reformatting.
7. Application	Provides services to the users of the OSI environment; examples: transaction server, file transfer protocol, network management.

Table 2-1: The OSI Layers

Of course, it takes two to communicate, so the same set of layered functions must exist in two systems. Communications is achieved by having the corresponding (peer) layers in two systems communicate. Layer *n* on one machine carries on a conversation with layer *n* on another machine. The peer layers communicate by means of a set of rules or conventions known as a *protocol.* The entities comprising teh corresponding layers on different machine are called *peer processes.* The peer processes communicate using the protocol. The set of layers and protocols is called a *network architecture.*

Figure 2-1 illustrates the OSI model. Each system contains the seven layers. Communication is between applications in the systems, labeled AP X and AP Y in the figure. IF AP X wishes to send a message to AP Y, it invokes the application layer (layer 7). Layer 7 establishes a peer relationship with layer 7 of the target machine, using a layer 7 protocol. This protocol requires services from layer 6, so the two layer 6 entities use a protocol of their own, and so on down to the physical layer, which actually passes the bits through a transmission medium. There is no direct communication between peer layers except at the physical layer.

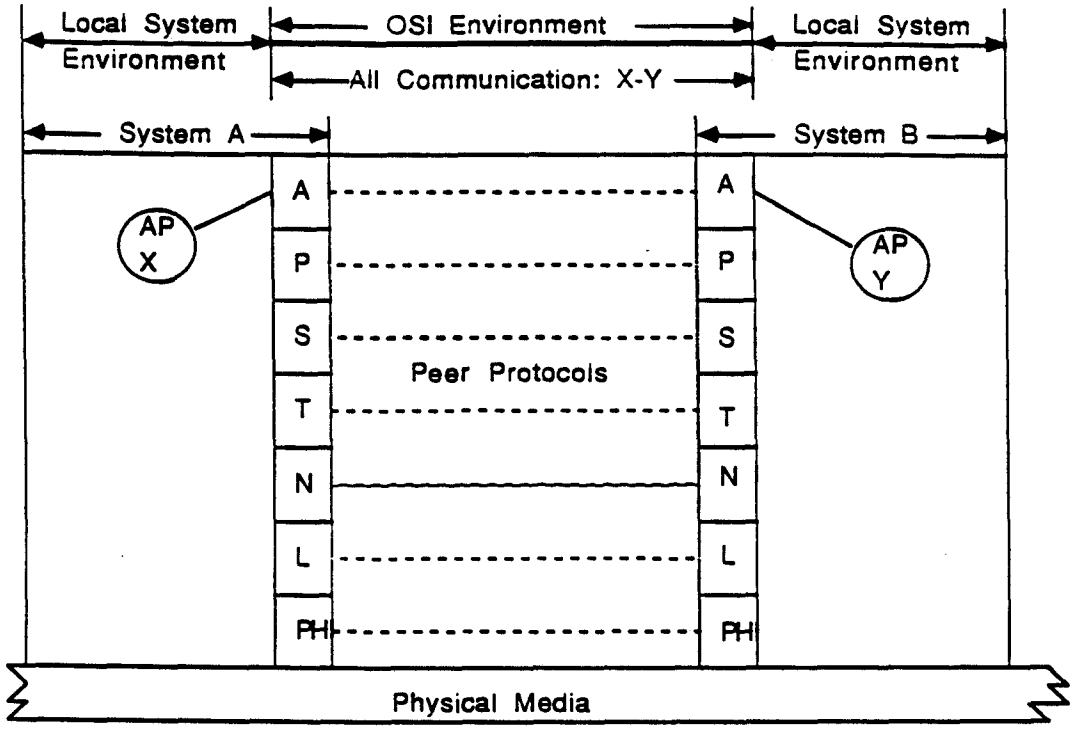


Figure 2-1: The OSI Environment

The term *n-layer* is used to refer to an unspecified layer of the architecture. See Figure 2-2. The names of constructs associated with that layer are also preceded by *n*. Within one system, the component responsible for the n-layer is called the *n-entity* or *n-process*. There are one or more active entities at each layer. The two n-entities communicate with each other using a protocol called the *n-protocol*. Between each pair of adjadent layers there is an interface. The interface defines which primitive operations and services the lower layer offers to the upper one. Protocols are conveyed using the services of the next lower layer. In general, the n-protocol is carried as data by the *(n-1)-service*. Each entity communicates with entities in the layers above and below it across an interface called Service Access Points (SAPs). The n-layer SAPs are the places where the (n+1)-layer can access the services offered. Each SAP has an address that uniquely identifies it. The lower layer has no knowledge of, or interest in, the content of the data it conveys. Each n-entity provides the *n-service* to a local client. The client may be either the (n+1)-layer, or an end-user.

The operation of the OSI/RM is straightforward. Some functions common to all layers are:

- Encapsulation
- Segmentation
- Connection establishment

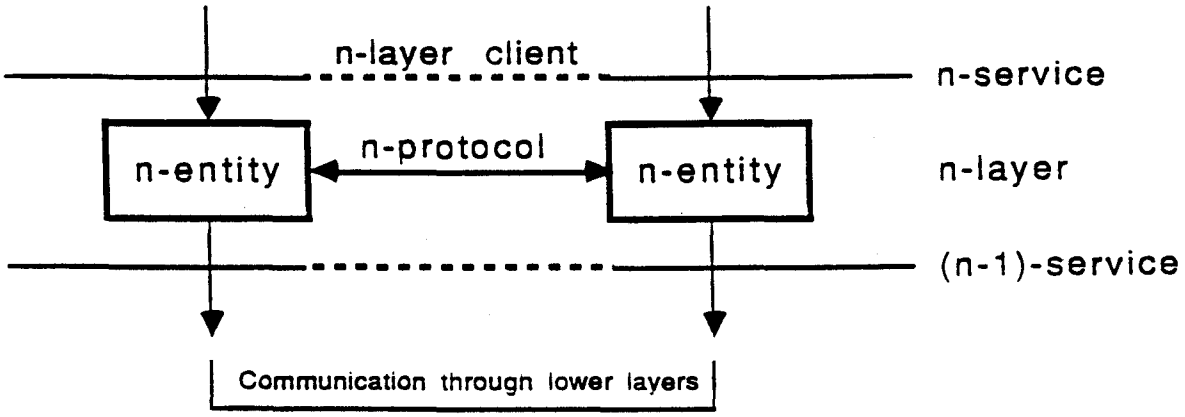


Figure 2-2: Principles of a Single Layer

- Flow control
- Error control
- Multiplexing

The most common way in which protocols are realized is by a process of encapsulation. When AP X has a message to send to AP Y, it transfers those data to a 7-entity in the application layer. A header is appended to the data that contains the required information for the peer layer 7 protocol; this is referred to as an encapsulation of the data. See Figure 2-3. The original data, plus the header, is now passed as a unit to layer 6. The 6-entity treats the whole unit as data, and appends its own header (a second encapsulation). This process continues down through layer 2, which generally adds both a header and a trailer, with the trailer containing a Frame Check Sequence (FCS) for error detection. This layer 2 unit, called a frame, is then transmitted by the physical layer onto the transmission medium. When the frame is received by the target system, the reverse process occurs. As the data ascend, each layer strips off the outermost header, acts on the protocol information contained therein, and passes the remainder up to the next layer.

A layer may segment the data unit it receives from the next higher layer into several parts to accommodate its own requirements. The corresponding peer layer must then reassemble the data units before passing them up to the next layer.

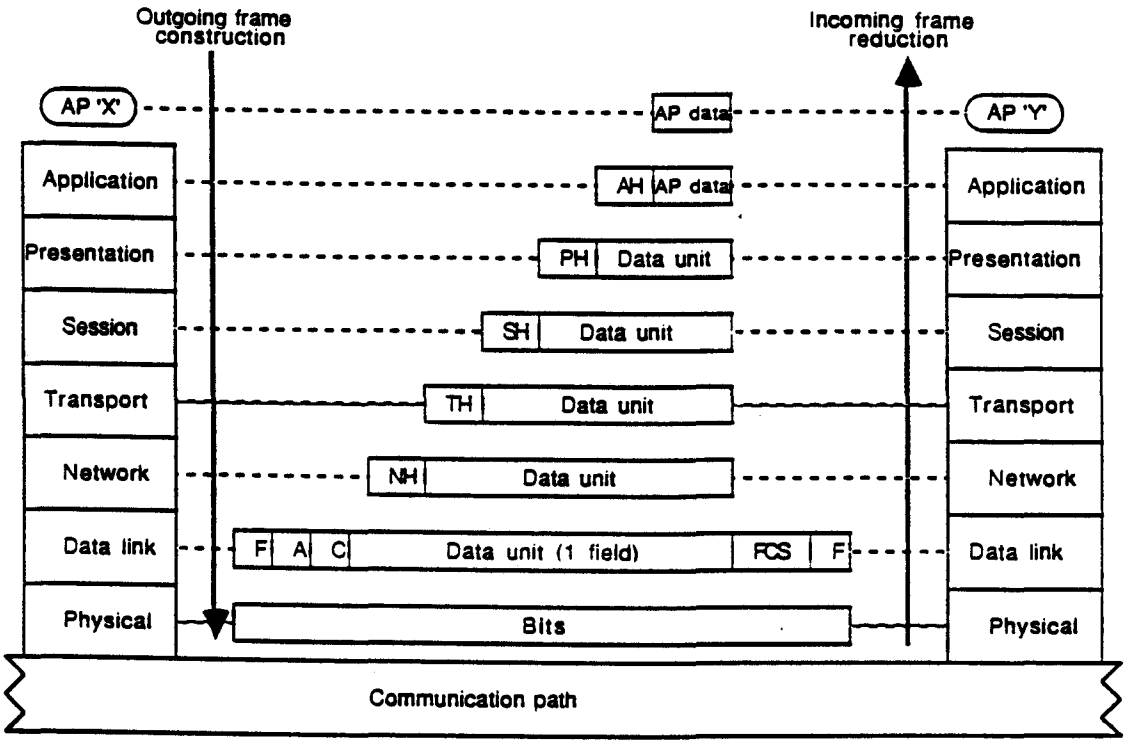


Figure 2-3: The OSI Operation

Flow control is a function performed by an n-entity to limit the amount or rate of data it receives from another n-entity. This prevents overflow. Error control refers to mechanisms to detect and correct errors that occur in the transmission of data units between peer entities.

Multiplexing can occur in two directions. Upward multiplexing means that multiple (N) connections are multiplexed on, or share, a single (N-1) connection. Downward multiplexing, or splitting, means that a single (N) connection is built on top of multiple (N-1) connections, the traffic on the (N) connection being divided among the various (N-1) connections.

2.2. Layers

Physical Layer

The physical layer covers the physical interface between devices and the rules by which bits are passed from one to another. The physical layer has four important characteristics:

- Mechanical
- Electrical

- Functional
- Procedural

Examples of standards at this layer are RS-232-C, RS-449/422/423, and portions of X.21.

Data Link Layer

Although the physical layer provides only a raw bit stream service, the data link layer attempts to make the physical link reliable and provides the means to activate, maintain, and deactivate the link. The principal service provided by the data link layer to the higher layers is that of error detection and control. Thus, with a fully functional data link layer protocol, the next higher layer may assume virtually error-free transmission over the link. However, if communication is between two systems that are not directly connected, the connection will compromise a number of data links in tandem, each functioning independently. Thus the higher layers are not relieved of an error control responsibility. Examples of standards at this layer are HDLC, LAP-B, LAP-D, and LLC.

Network Layer

The basic service of the network layer is to provide for the transparent transfer of data between transport entities. It relieves the transport layer of the need to know anything about the underlying data transmission and switching technologies used to connect systems. The network service is responsible for establishing, maintaining, and terminating connections across the intervening communications facility.

It is at this layer that the concept of a protocol becomes a little fuzzy. This is best illustrated with reference to Figure 2-4 which shows two stations communicating via a packet-switched network. The packet-switched network is known as a *subnetwork*. The stations have direct links to the network nodes. The layer 1 and 2 protocols are station-node protocols (local). Layers 4 through 7 are clearly protocols between n-entities in the two stations. Layer 3 is a little bit of both.

The principal dialogue is between the station and its node; the station sends addressed packets to the node for delivery across the network. It requests a virtual circuit connection, uses the connection to transmit data, and terminates the connection. All of this is done by means of a station-node protocol. However, because packets are exchanged and virtual circuits are set up between two stations, there are aspects of station-station protocol as well.

The most common use of layer 3 is to handle the details of using a communication network. In this case, the network entity in the station must provide the network with sufficient information to switch and route data to another station. The best known example of layer 3 is the X.25 layer 3 standard. The Internet Protocol (IP) is another example of a layer 3 standard.

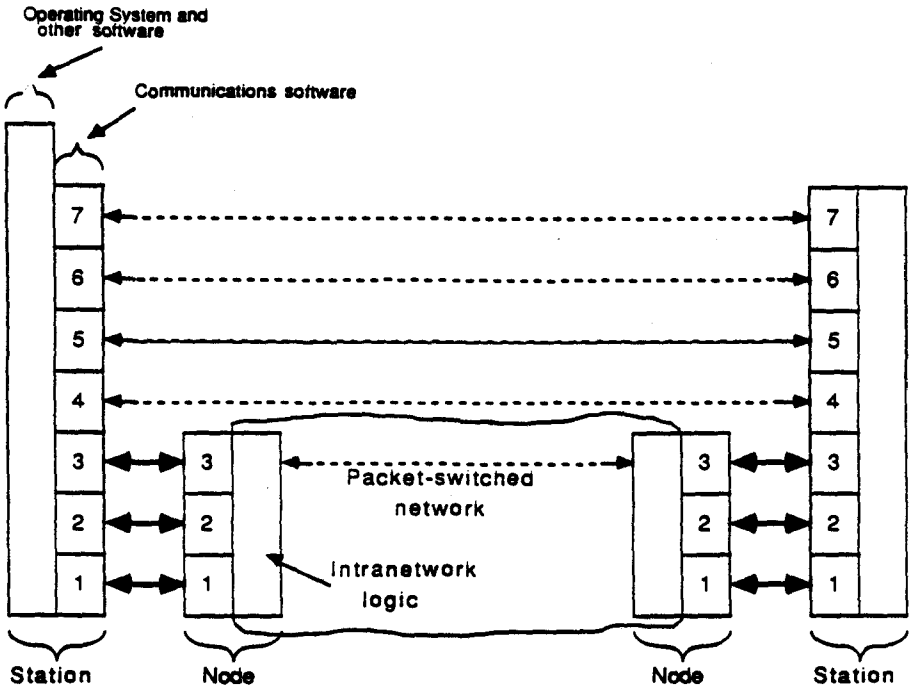


Figure 2-4: Communication Across a Network

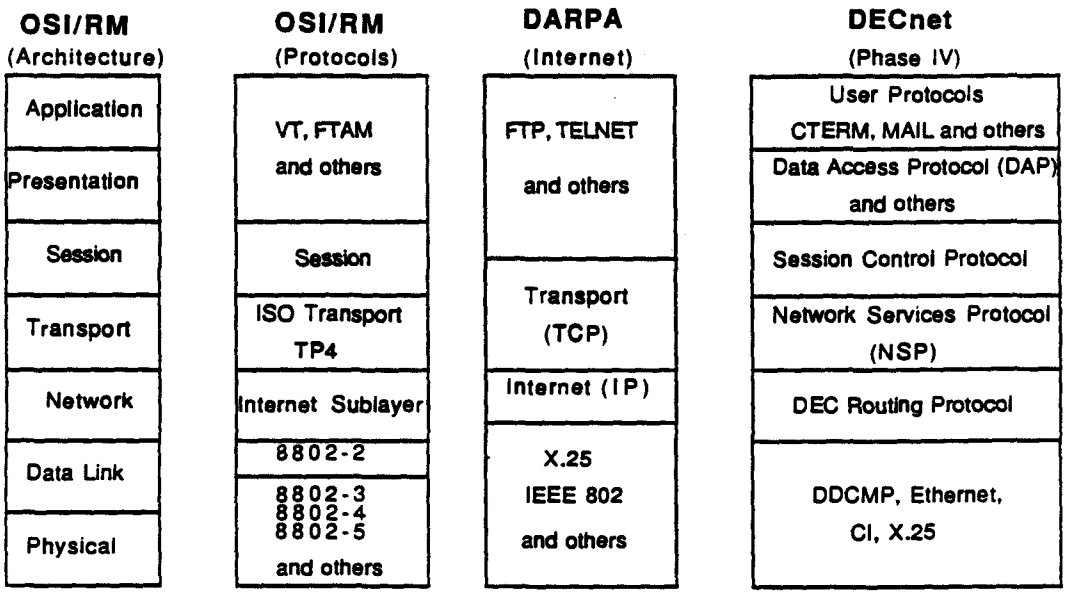


Figure 2-5: Some Well-Known Layers

Transport Layer

Layers 4 and above in the OSI model are generally referred to as the higher layers. Protocols at these levels are end-to-end and not concerned with the details of the underlying communications facility.

The purpose of the transport layer is to provide a reliable mechanism for the exchange of data between processes in different systems. The transport layer ensures that data units are delivered error-free, in sequence, with no losses or duplications. Typical features of the transport layer are:

Type of service: connection-oriented or connectionless.

Grade of service: allows the 5-entity to specify acceptable error and loss levels, desired delay, priority, and security.

Connection management:
sets up and manages connections between 5-entities.

Some well known transport layers are TCP and ISO's TP4.

Session Layer

The session layer provides the mechanism for controlling the dialogue between presentation entities. At a minimum, the session layer provides a means for two presentation entities to establish and use a connection, called a session. It may also provide:

Dialogue type: two-way simultaneous, two-way alternate, or one-way.

Recovery: The session layer can provide a checkpointing mechanism, so that if a failure of some sort occurs between checkpoints, the session entity can retransmit all data since the last checkpoint.

The ISO session protocol is a good example of an implementation of the session layer.

Presentation Layer

The presentation layer offers application programs and terminal handler programs a set of data transformation services.

Data translation:
Code and character set translation.

Formatting: Modification of data layout.

Syntax selection: Initial selection and subsequent modification of the transformations used.

Examples of presentation protocols are text compression, encryption, and virtual terminal protocol (ISO's VTM). Protocols implemented at this layer often also

encompass the application layer. eXternal Data Representation (XDR) is another presentation protocol. It is used to represent data on the network in a standard, consistent way.

Application Layer

The application layer provides a means for application processes to access the OSI environment. This layer contains management functions and generally useful mechanisms to support distributed applications. Examples of protocols at this level are virtual file protocol and job transfer and manipulation protocols. ISO has a procol called FTAM (File Transfer, Access, and Management).

2.3. Services and Functions of the OSI Layers

Physical Layer

Services	Functions
Physical connections	Physical-connection activation and deactivation
Physical SDUs	Physical SDU transmission
Physical connection endpoints	Physical layer management
Data-circuit identification	
Sequencing	
Fault condition notification	
Quality of service parameters	

Data Link Layer

Services	Functions
Data-link connection	Data-link connection establishment and release
Data-link SDUs	Data-link SDU mapping
Data-link connection endpoint identifiers	Data-link connection splitting
Sequencing	Delimiting and synchronization
Error notification	Sequence control
Flow control	Error recovery
Quality of service parameters	Error recovery

	Flow control
	Identification and parameter exchange
	Control of data-circuit interconnection
	Data link layer management

Network Layer

Services	Functions
Network addresses	Routing and relaying
Network connections	Network connections
Network connection endpoint identifiers	Network connection multiplexing
	Segmenting and blocking
Network SDU transfer	Error detection
Quality of service parameters	Error recovery
Error notification	Sequencing
Sequencing	Flow control
Flow control	Expedited data transfer
Expedited NSDU transfer	Reset
Reset	Service selection
Release	Network layer management

Transport Layer

Services	Functions
Transport connection establishment	Mapping transport addresses onto network addresses
Data transfer	Multiplexing transport connections onto network connections
Transport connection release	Establishment and release of transport connections
	End-to-end sequence control on individual connections
	End-to-end error detection and any necessary monitoring of the quality of service
	End-to-end error recovery

	End-to-end segmenting,blocking, and concatenation
	End-to-end flow control on individual connections
	Supervisory functions
	Expedited TSDU transfer

Session Layer

Services	Functions
Session connection establishment	Session connection to transport connection mapping
Session connection release	Session connection flow control
Normal data exchange	Expedited data transfer
Quarentine service	Session connection recovery
Expedited data exchange	Session connection release
Interaction management	Session layer management
Session connection synchronization	
Exception reporting	

Presentation Layer

Services	Functions
Transformation of syntax	Session establishment request
Selection of syntax	Data transfer
	Negotiation and renegotiation of syntax
	Transformation of syntax including data transformation, formatting, and special purpose transformations
	Session termination request

Application Layer

Services	Functions
Information transfer Identification of intended communications partners Determination of the current availability of the intended communication partners Establishment of authority to communicate Agreement of privacy mechanisms Authentication of intended communication partners Determination of cost allocation methodology Determination of the adequacy of resources Determination of the acceptable quality of service Synchronization of cooperating applications Selection of the dialogue discipline including initiation and release procedures	Functions that imply communication between open systems and are not already performed by the lower layers

2.4. Terminology

Repeater: Repeaters are low-level devices that amplify just electrical signals. They are needed to provide current to drive long cables. Repeaters copy the bits as they arrive.

Bridge: Bridges are store and forward devices. A bridge accepts an entire frame and passes it up to the data link layer where the checksum is verified. Then the frame is sent down to the physical layer for forwarding on a different subnet. Bridges can make minor changes to the frame before forwarding it, such as adding or deleting some fields from the frame header. They are

data link layer devices and therefore do not deal with headers at layer three and above.

Gateway: Gateways are conceptually similar to bridges, except that they are found in the network layer. Networks connected by a gateway can differ much more than those connected by a bridge. A major advantage of bridges is that they can connect networks with incompatible addressing formats.

Protocol converter:
The job of a protocol converter is much more complex than that of a gateway. The protocol converter must convert from one protocol to another without losing much meaning in the process.

2.5. References and Further Reading

1. Stallings, W.. *Handbook of Computer-Communications Standards.* Macmillan, New York, 1987.

2. Stallings, W.. *Local Networks An Introduction.* Macmillan, New York, 1987.

3. Stallings, W.. *Handbook of Computer-Communications Standards.* Macmillan Publishing Company, New York, 1988.

4. Tanenbaum, A.S.. *Computer Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1988.

This Page Intentionally Left Blank

Part II
NETWORK TECHNOLOGIES

This Page Intentionally Left Blank

Chapter 3

Understanding Ethernet

A local area network (LAN) is a communications network that provides interconnection of a variety of data communicating devices within a small area. Some of the typical characteristics of local networks are high data rates (0.1 to 100 Mbps), short distances (0.1 to 25 km), and a low error rate. The topology, transmission medium, and medium access control technique determine the nature of a local network.

3.1. What is Topology?

Topology refers to the layout of a network. It explains how stations of the network are interconnected. In the **star topology**, each station is connected by a point-to-point link to a common central switch. A LAN employing the **ring topology** consists of set of repeaters joined by point-to-point links in a closed loop. Each station attaches to the network at a repeater. No switches or repeaters are needed when using the **bus topology**. Any transmission propagates the length of the medium. In the **tree topology**, the transmission medium is a branching cable with no closed loops. Its behavior is similar to a bus. These four simple topologies are used in the construction of LANs.

3.2. What are Transmission Media?

The physical path between the transmitter and receiver in a communications network is known as the **transmission medium.** The three most common transmission media in LAN technology are twisted pair, coaxial cable, and optical fiber. Tables 3-1 and 3-2 give some characteristics of different transmission media for comparison. The values in Table 3-1 are for point-to-point connections whereas the values in Table 3-2 are for multipoint connections. Notice the difference in data rates between point-to-point and multipoint.

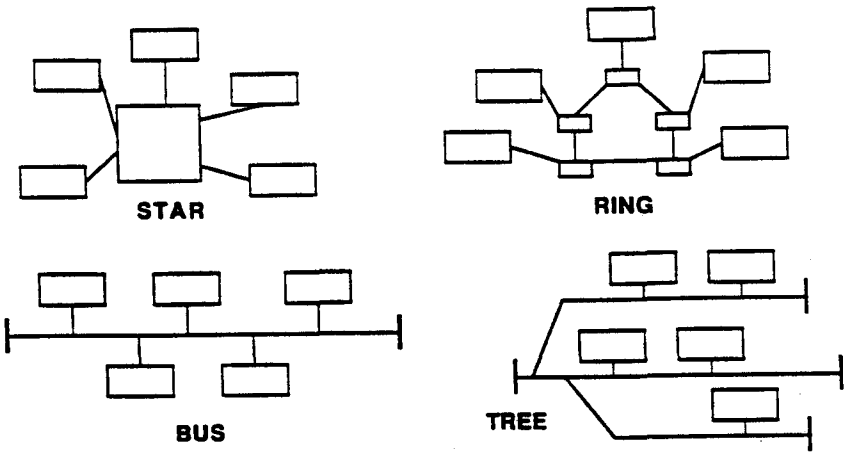


Figure 3-1: Local Network Topologies

Transmission Medium	Total Data Rate	Bandwidth	Repeater Spacing
Twisted Pair	4 Mbps	250 kHz	2-10 km
Coaxial Cable	500 Mbps	350 MHz	1-10 km
Optical Fiber	2 Gbps	2 GHz	10-100 km

Table 3-1: Transmission Media for Local Networks: Point-to-point

FDM in Table 3-2 is an abbreviation for Frequency Division Multiplexing. A number of signals can be carried simultaneously if each signal is modulated onto a different carrier frequency. The single channel analog signaling technique is one in which the entire spectrum of the cable is devoted to a single transmission path for analog signals.

3.3. What are Medium Access Protocols?

Control of access to the network is the job of the **medium access protocol**. There are three categories of access protocols:

Round robin: Everybody takes a turn when using the round robin technique. Token bus, used in bus/tree networks, and token ring, used in ring networks, are examples of protocols in the round robin category.

Medium	Signaling Technique	Maximum Data Rate (Mbps)	Maximum Range at Maximum Data Rate (km)	Practical Number of Devices
Twisted Pair	Digital	1-2	Few	10's
Coaxial cable (50 ohm)	Digital	10	Few	100's
Coaxial cable (75 ohm)	Digital	50	1	10's
	Analog with FDM	20	10's	1000's
	Single channel Analog	50	1	10's
Optical fiber	Analog	10	1	10's

Table 3-2: Transmission Media for Local Networks: Multipoint

Reservation: The reservation technique requires stations to reserve future time slots for transmission. Centralized reservation for bus networks is an example of a protocol using the reservation technique.

Contention: Contention techniques use no control to determine whose turn it is. Stations compete for access. Carrier Sense Multiple Access with Collision Detection (CSMA/CD) is an example of a contention protocol. It is one of the most common access protocols for bus networks. Register insertion and slotted ring protocols are examples of contention techniques used in ring networks.

Specifics for each protocol will not be given here. The interested reader should consult any local network text book.

3.4. Principles of Ethernet

Ethernet, a 10 Mbps CSMA/CD network, is one of the most successful LAN technologies. It is a broadcast system for local communication among computing stations. Ethernet is named for the historical luminiferous ether through which electromagnetic radiations were once alleged to propagate. Its shared communication facility, its Ether, is a passive broadcast medium with no central control. Coordination of network access is completely distributed among contending stations. Any station wishing to transmit must wait if another transmission is already in progress. If no other station is transmitting, the sender can begin immediately. Collisions occur when 2 or more stations sense the channel is idle and begin to transmit simultaneously. In the event of a collision, all transmission ceases while colliding stations are made aware. The colliding stations then wait a

random amount of time before retransmitting. Ethernet is implemented on networks exhibiting a bus topology, therefore transmissions can be heard by all stations. Taps are used to connect stations to the network. The complete Ethernet specification, developed by Digital Equipment Corporation, Intel Corporation, and Xerox Corporation, can be found in *The Ethernet: A Local Area Network: Data Link Layer and Physical Layer Specifications, Version 2.0.*

3.5. Ethernet and the OSI Reference Model

The minimum essential communications functions that must be performed by a LAN correspond to layers 1 and 2 of the Open Systems Interconnection Reference Model (OSI/RM) as shown in Figure 3-2. Two characteristics of LANS are important in this context. First, data are transmitted in addressed frames. Second, there is no intermediate switching, hence no routing required. It may seem that the network layer would also be needed. Ethernet provides only unacknowledged connectionless service at the data link layer, therefore the network layer function of providing the upper layers with independence from the data transmission as already been accomplished. The layer 1 or physical layer functions of Ethernet include:

1. encoding/decoding of signals
2. preamble generation/removal (for synchronization)
3. bit transmission/reception

Recall, the data link layer attempts to make the physical link reliable and provides the means to activate, maintain, and deactivate the link. It provides error control and flow control. The functions of the data link layer are:

1. to provide a logical interface between two adjacent layers (i.e. the physical layer and the network layer - unacknowledged connectionless service)
2. assemble the data into frames with address and Cyclical Redundancy Check (CRC) fields
3. disassemble received frames, perform address recognition and CRC validation
4. manage communication over the link.

The functions of these layers as found in Ethernet should directly correspond the the traditional layers 1 and 2 (physical and data link) in the OSI/RM.

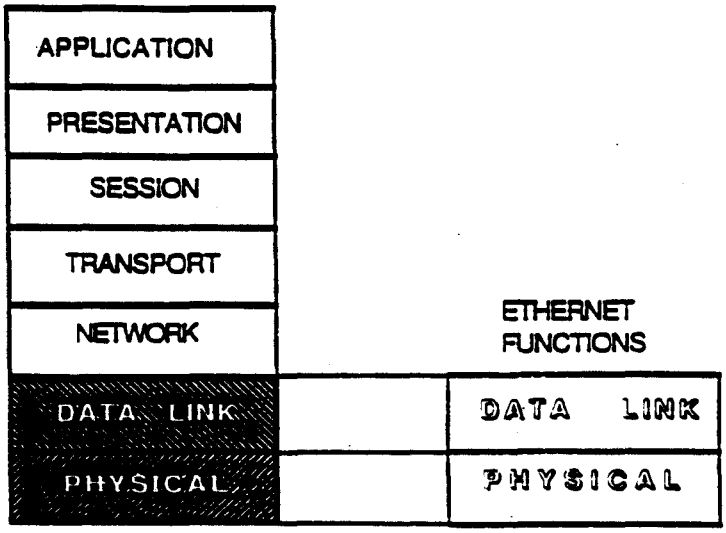


Figure 3-2: Ethernet Protocol Layers Compared to OSI/RM

3.6. Ethernet: Physical Specification

The physical specification for Ethernet describes the physical characteristics of the network. The specification provides the type of transmission technique, baseband or broadband. The components that physically make up Ethernet are described along with their functions and limitations. The limitations ensure that the network behaves properly. The components of the Ethernet architecture are described later in this section. The actual Ethernet physical specification is called 10BASE5.

3.6.1. Baseband and Broadband Transmission Techniques

Two transmission techniques are in use for LANs: baseband and broadband. Figure 3-3 describes the two techniques. Baseband, using digital signaling, can be employed on twisted-pair or coaxial cable. Broadband, using analog signaling in the radio-frequency range, employs coaxial cable.

In baseband systems, digital signals are inserted on the line as voltage pulses, usually using either Manchester or Differential Manchester encoding. The entire frequency spectrum of the medium is used to form the signal. Transmission is bidirectional. A signal inserted at any point on the medium propagates in both directions to the ends, where it is absorbed. The digital signaling requires a bus topology. Baseband has the advantage of simplicity and lower cost.

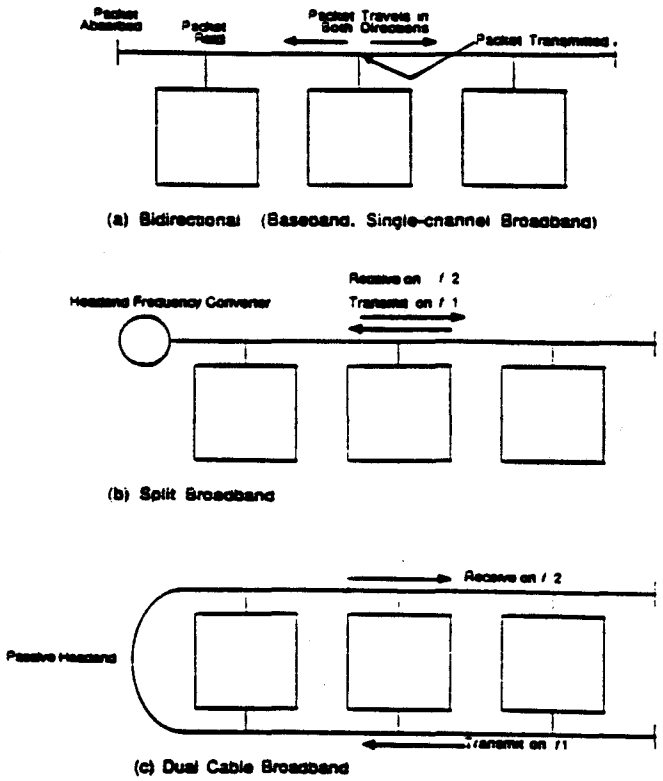


Figure 3-3: Baseband and Broadband Transmission Techniques

Broadband refers to any channel having a bandwidth greater than a voice-grade channel (4 kHz). The term is usually reserved for coaxial cable on which analog signaling is used. Broadband systems are capable of frequency-division multiplexing (FDM). Broadband is inherently a unidirectional medium. Signals inserted onto the medium can propagate in only one direction. Broadband's strength is its tremendous capacity. A wide variety of traffic can be carried on a number of channels and a very wide area of coverage can be achieved. These systems are more complex than baseband to install and maintain.

3.6.2. Ethernet : Physical Architecture

Figure 3-4 from the Ethernet specification, illustrates typical components and their functions. The main components are:

- Transceiver
- Transceiver cable
- Controller
- 50-ohm coaxial cable
- 50-ohm terminators
- Repeaters

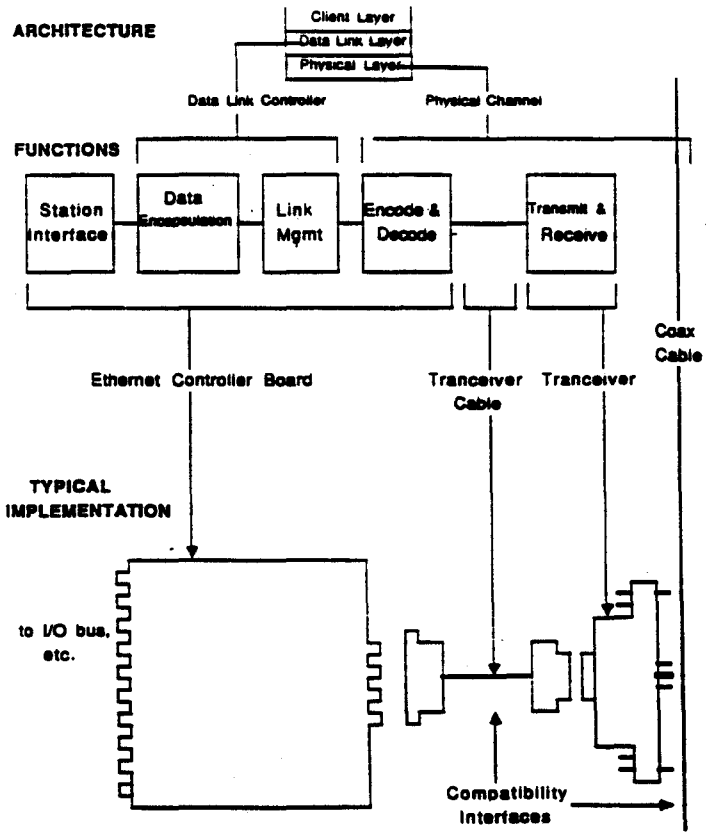


Figure 3-4: Ethernet Architecture

The transceiver taps into the coaxial cable. It transmits signals from the station to the cable, and vice versa. It also contains the electronics necessary to recognize the presence of a signal on the coaxial cable and to recognize a collision of two signals. Ground isolation between the signals from the station and the signals on the cable is provided by the transceiver.

The transceiver cable connects the transceiver to the controller, which contains most of the intelligence required to communicate over the LAN. The cable supplies power to the transceiver and passes data signals as well as control signals between the transceiver and the controller. Included in the control signals is a collision presence signal.

The controller is an implementation of all the functions (other than those performed by the transceiver) needed to manage access to the coaxial cable. It controls the exchange of data between the coaxial cable and the attached station.

Finally, the transmission system consists of 50-ohm coaxial cable and terminators. The terminators absorb signals, preventing reflection from the ends of the bus.

50-ohm repeaters can be used to extend the length of the network. A repeater consists of two transceivers joined together and connected to two different segments of coaxial cable. The repeater passes digital signals in both directions between the two segments, amplifying and regenerating the signals as they pass through.

3.6.3. The 10BASE5 Specification

The 10BASE5 medium specification is the original Ethernet specification. The first two digits give the data rate in megabits per second; the four letters are an abbreviation for the medium (baseband); and the final digit is the maximum cable length in hundreds of meters.

The medium employed is a 50-ohm coaxial cable. This is a special purpose coaxial cable that is usually used for baseband LANs in preference to the standard CATV 75-ohm cable. These values refer to the characteristic impedance of the cable. It defines the ratio of electromagnetic wave voltages to currents at every point along the conductor. Roughly speaking, impedance is a measure of how much voltage must be applied to the cable to achieve a given signal strength. The simplest baseband coaxial bus LAN consists of an unbranched length of coaxial cable with a terminating resistance at each end. The value of the resistance is set equal to the impedance of the cable, thus preventing reflection by absorbing any signal on the cable. For digital signals, the 50-ohm cable suffers less intense reflections from the insertion capacitance of the taps, and provides better immunity against low-frequency electromagnetic noise than 75-ohm cable.

Parameter	10BASE5
Transmission medium	Coaxial cable (50-ohm)
Signaling technique	Baseband (Manchester)
Data rate (Mbps)	10
Maximum segment length (m)	500
Network span (m)	2500
Nodes per segment	100
Node spacing (m)	2.5
Cable diameter (mm)	10
slotTime (bit times)	512
interFrameGap (μs)	9.6
attemptLimit	16
backoffLimit	10
jamSize (bits)	32
maxFrameSize (octets)	1518
minFrameSize (octets)	64

Table 3-3: Ethernet Physical Specification Parameter Values

10BASE5 specifies the 50-ohm coaxial cable and a data rate of 10 Mbps using digital signaling with Manchester encoding. The maximum length of the cable is set at 500 m. Stations attach to the cable by means of a tap, with the distance between any two taps being a multiple of 2.5 m. This spacing ensures that reflections from adjacent taps do not add in phase. A maximum of 100 taps is allowed per segment.

The length of the network can be extended by the use of repeaters. A repeater is transparent to the rest of the system. Only one path of segments and repeaters is allowed between any two stations. The standard allows a maximum of four repeaters in the path between any two stations, extending the effective length of the cable to 2.5 km.

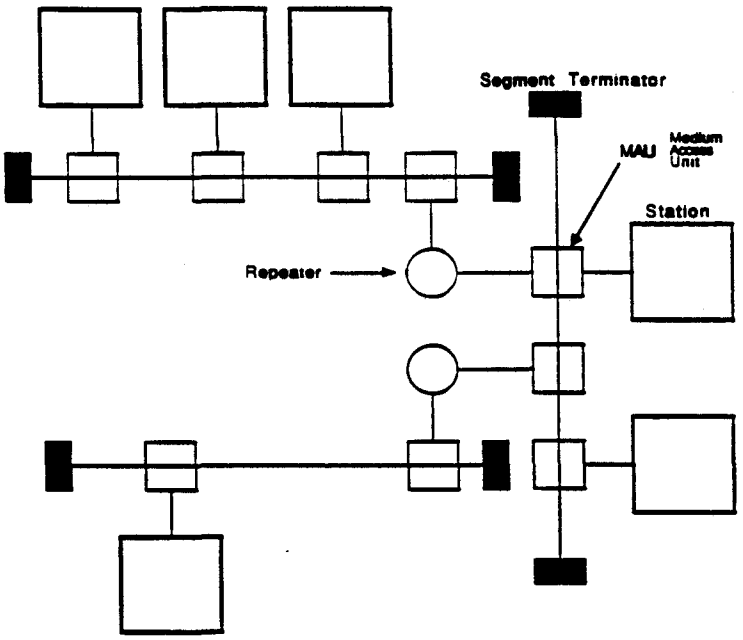


Figure 3-5: 10BASE5 Configuration with Three Segments

The standard specifies that a transmitting station will detect a collision if the signal on the cable at the station exceeds the maximum that could be produced by the station alone. Collisions produce substantially higher voltage swings than those produced by a single transmitter. Collision conditions must cross repeater boundaries as well. When a repeater detects a collision on either cable, it must transmit a jamming signal on the other side.

3.6.4. The Physical Topology at Fermilab

Figure 3-6 represents the physical topology of the network here at Fermilab at the present time. Keep in mind, that the network is constantly changing.

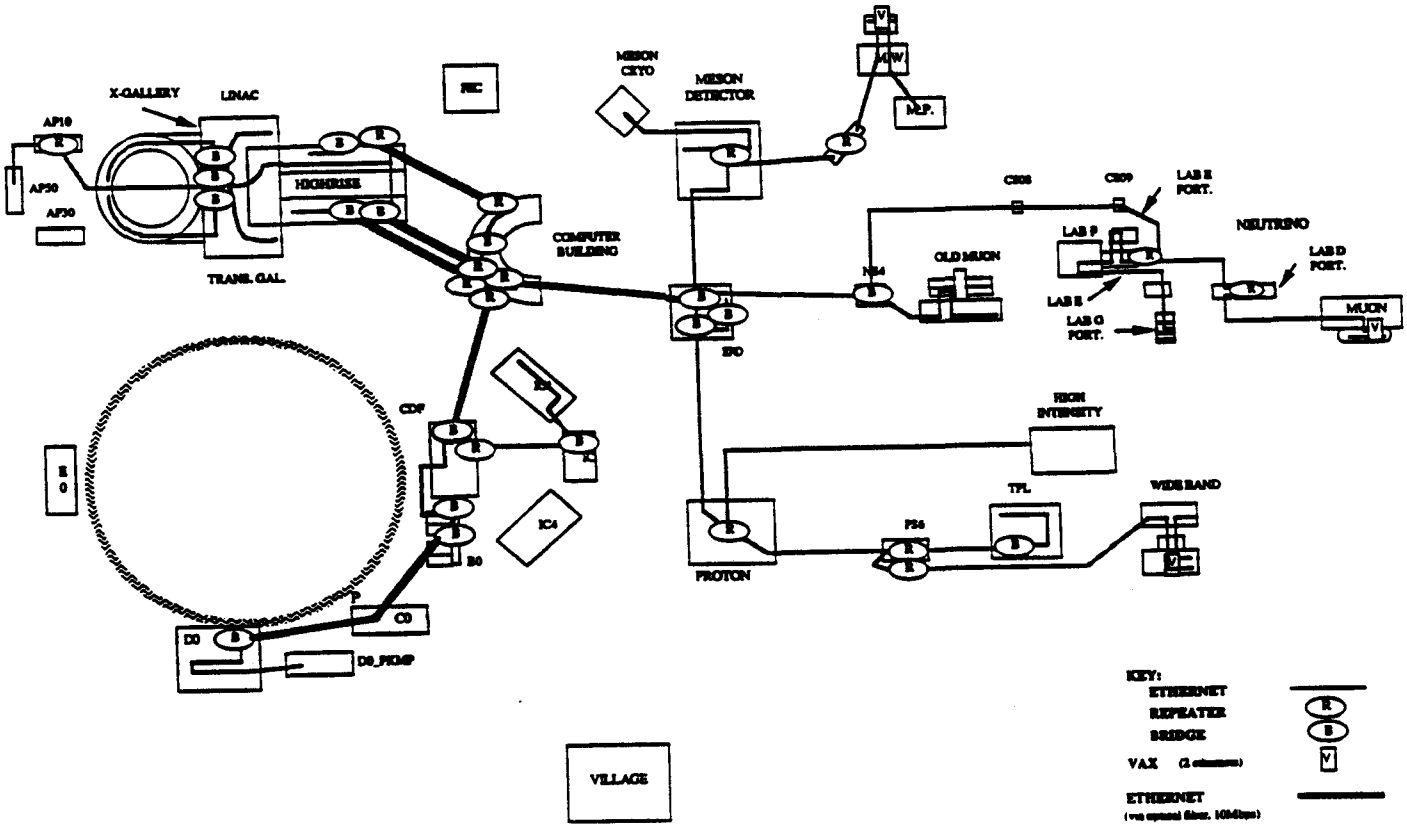


Figure 3-6: The Physical Topology at Fermilab

3.7. The CSMA/CD Protocol

CSMA/CD is is the medium access protocol used by Ethernet. It performs the OSI/RM layer 2 (data link layer) function of managing communication over the link.

3.7.1. Persistence

The persistence of the protocol determines when transmission takes place on an idle cable and how quickly another attempt will be made if the cable is busy. In CSMA/CD there are three levels of persistence : 1-persistent, p-persistent and non-persistent.

Non-persistent : If the medium is idle transmission occurs immediately. If the channel is busy, the station waits a random amount of time drawn from a probability distribution and then tests the channel again repeating the previous steps.

P-persistent : If the medium is idle, transmission occurs immediately with probability p, and is delayed by the end-to-end propagation delay with probability $(1 - p)$. If the medium is busy, the station defers until the channel is idle and then, with probability p, transmits, and with probability $(1 - p)$, delays transmission.

1-persistent : If the medium is idle, transmission occurs immediately, otherwise the station defers until the medium is idle and then transmits immediately.

Non-persistent and p-persistent systems have performance problems. Non-persistent stations always defer before trying to transmit. Capacity is wasted because the medium will generally remain idle following the end of a transmission even if there are stations waiting to send. The p-persistent protocol attempts to reduce idle time and collisions, yet p must be set low enough to avoid instability, which can result in atrocious delays under light load. The 1-persistent stations are selfish. A collision is guaranteed if two or more stations are waiting to transmit. The 1-persistent algorithm would seem to be even more unstable than p-persistent due to the greed of the stations. But the wasted time due to collisions is mercifully short. Two stations involved in a collision are unlikely to collide on their next tries because of the random backoff mechanism used by CSMA/CD. The 1-persistent protocol is the most common and is used in the CSMA/CD algorithm employed by Ethernet.

3.7.2. CSMA/CD Algorithm

The basic operation of the CSMA/CD protocol can be explained with the aid of the flow diagram shown in Figure 3-7. After the frame is transmitted, CSMA/CD employs hardware to listen while the transmitted frame is in transit over the cable. If no collisions are detected, a frame is successfully transmitted, and the algorithm is exited. On the other hand, if a collision is detected, the frame is promptly aborted, a jamming signal is transmitted and the collided frame is readied for retransmission after the appropriate back off time.

Stated concisely, in the CSMA/CD algorithm, a ready station takes the following action.

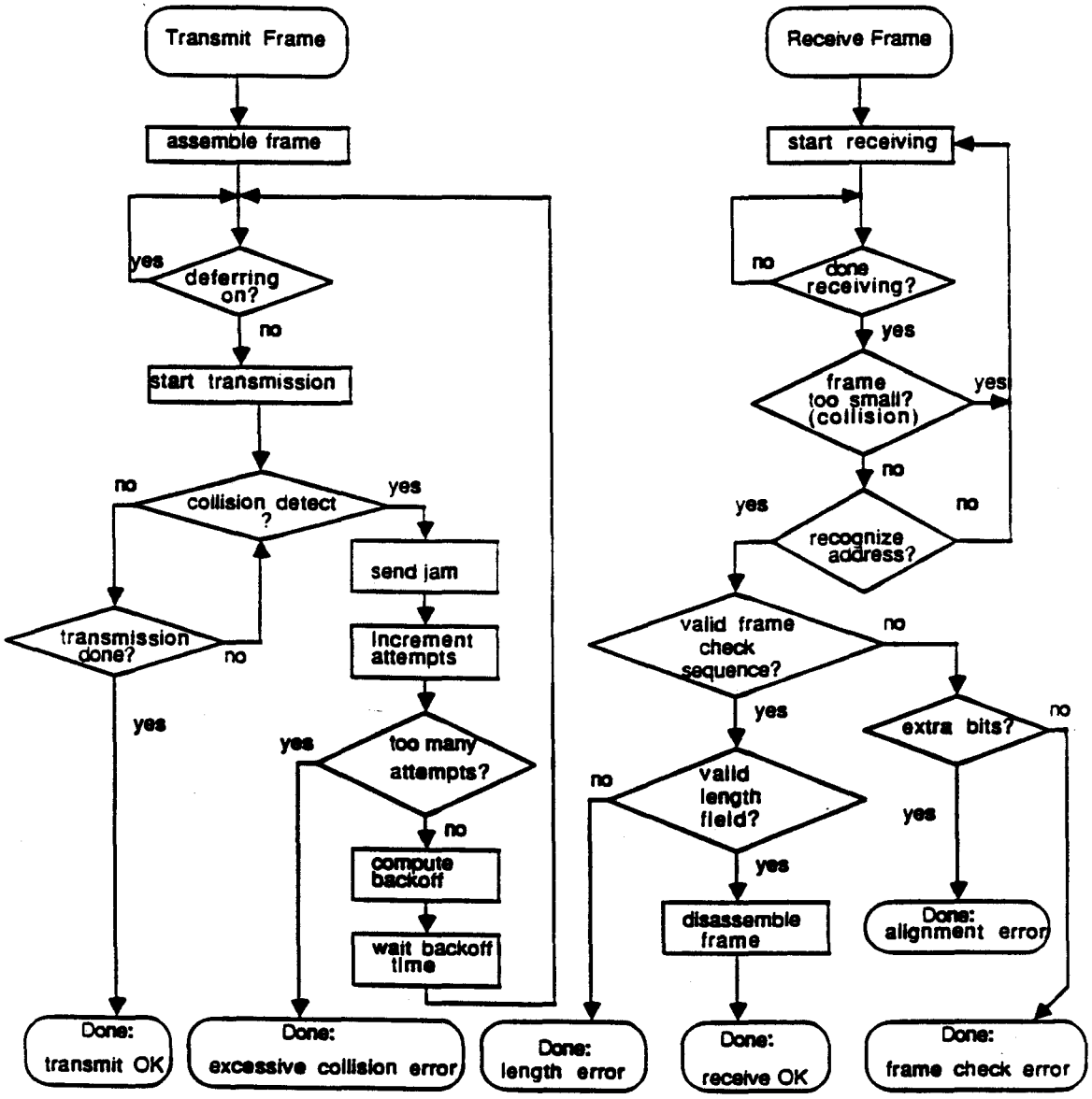


Figure 3-7: Control Flow Summary

1. If the channel is sensed idle, the frame is transmitted.
2. If the channel is sensed busy, then the station defers until the channel is sensed idle and then transmits a frame.
3. If a collision is sensed, the station aborts the frame being transmitted and transmits a jamming signal to assure that all stations know that there has been a collision and then cease transmission. After waiting a random amount of time, repeat step 1.

3.7.3. Slot Time

Slot time is a very important parameter in the CSMA/CD algorithm.

- It is an upper bound on the time it takes to detect a collision and the amount of wasted bandwidth.
- It is an upper bound on the acquisition time of the medium.
- It is an upper bound on the length of a frame fragment generated by a collision.
- It is the scheduling quantum for retransmission.

Therefore, slot time is defined to be larger than the sum of the physical layer round-trip propagation time. To understand slot time, consider Figure 3-8. Two stations are as far apart as possible. The amount of time it takes to detect a collision is twice the propagation delay. Ethernet requires that frames be long enough to detect a collision before the end of transmission. This is the minimum frame length. Slot time also comes into play in the retransmission algorithm.

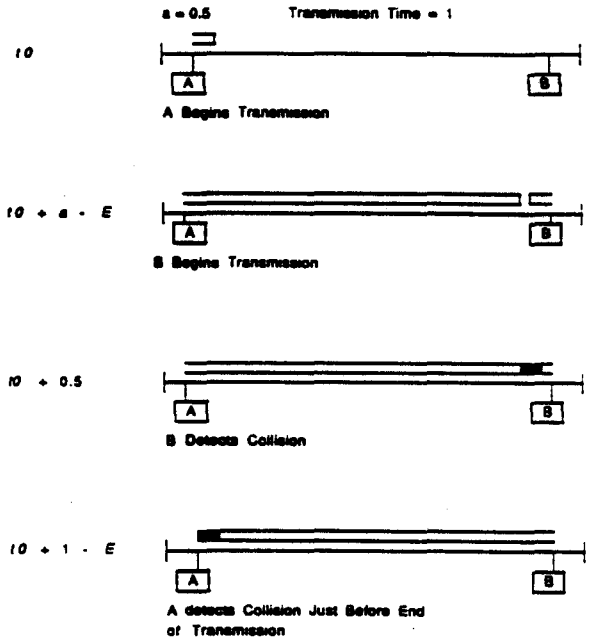


Figure 3-8: Collision Detection Timing

3.7.4. Backoff Algorithm

To ensure that backoff maintains stability, Ethernet uses a technique known as **truncated binary exponential backoff**. A station will attempt to transmit a frame repeatedly when collisions occur, but after each collision, the mean value of the random delay is doubled. After 16 unsuccessful attempts, the station gives up and reports an error. The backoff delay is an integral number of slot times. The number of slot times to delay before the nth retransmission attempt is

chosen as a uniformly distributed random integer r in the range $0 < r < 2^k$ where $k = \min(n,10)$.

```
while attempts < 16
    k := Min(attempts,10)
    r := Random(0,2^k)
    delay := r * slotTime
```

The truncated binary exponential backoff algorithm approximates the ideal algorithm where the probability of transmission of a frame is $1/Q$, with Q representing the number of stations attempting to transmit and with truncation occurring when Q equals the number of stations. The base backoff time is a suitably chosen time increment, often twice the end-to-end propagation delay.

The beauty of the 1-persistent algorithm with binary exponential backoff is that it is efficient over a wide range of loads. At low loads, 1-persistence guarantees that a station can seize the channel as soon as it goes idle, in contrast to the non-persistent and p-persistent schemes. At high loads, it is at least as stable as the other techniques. However, one unfortunate effect of the backoff algorithm is that it has a last-in, first-out effect; stations with no or few collisions will have a chance to transmit before stations that have waited longer.

3.7.5. Collision Detection and Resolution

Collisions produce higher voltage swings than those produced by a single transmitter. Accordingly, Ethernet specifies that a transmitting transceiver will detect a collision if the signal on the cable at the transceiver exceeds the maximum that could be produced by the transceiver alone. A transmitted signal attenuates as it propagates. This is the reason the maximum length of the cable is restricted to 500 m, thus preventing the signal strength from being so small that when it is added to the transmitted signal at the transceiver, the CD threshold is not exceeded. For a non-transmitting transceiver, a collision is detected if the signal strength exceeds that which could be produced by two transceiver outputs in the worst case.

A description of what happens in the channel during a collision interval (an unsuccessful busy period) is important in understanding the operation of CSMA/CD as employed in Ethernet. CSMA/CD has a vulnerable period τ seconds. If a station begins transmitting, another station will not detect the transmission until after the propagation delay of τ seconds. After a colliding signal arrives at a station that is already transmitting, the collision detection circuitry requires a time, denoted ϵ, to detect the presence of a collision.

Consider the timing diagram in Figure 3-9. Distance along the bus is on the horizontal axis with the spacing between two stations A and B indicated. The increasing time axis is drawn from top to bottom. Station A begins a transmission at time t_0 to start the cycle. The signal propagates to station B, but, before it arrives, station B initiates a transmission at the time $(t_0 + Y)$, where Y

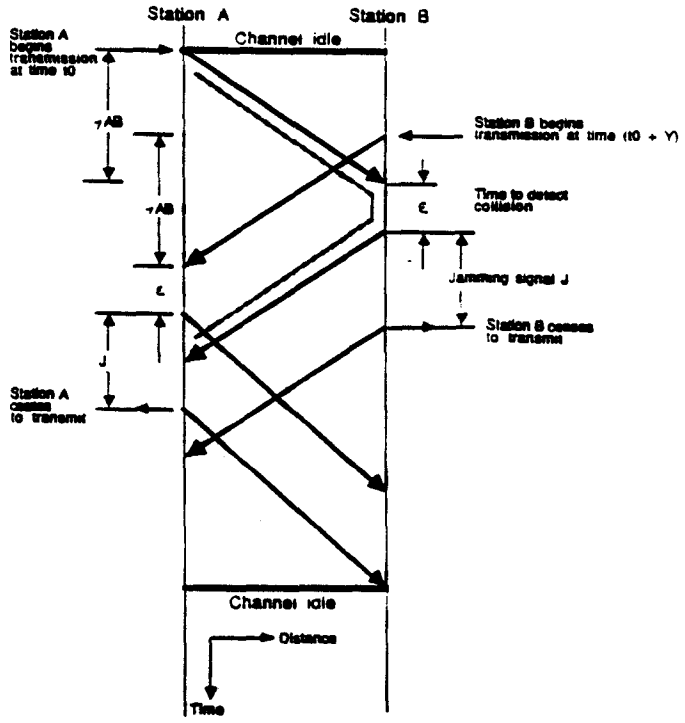


Figure 3-9: Timing Diagram for CSMA/CD, Showing Contention

is a random variable. Shortly after station B begins to transmit, the signal from A arrives and ϵ seconds later, B detects the collision. Station B then sends out a jamming signal for J seconds. The signal from B reaches A after τ_{AB} seconds, A detects a collision (ϵ seconds after receiving the signal from B), and then A initiates its own jamming signal for J seconds.

A is the first to transmit, transmits for the greater part of the collision interval, and transmits longer than station B. However, note that both stations view the channel as active for exactly the same length of time $(J + 2\tau_{AB} + \epsilon)$ seconds, which is independent of Y.

An idle period exists when no station perceives the channel to be busy. If the channel is not idle, it is busy. A busy period is a collision interval, if more than one station is attempting to transmit in the period. Otherwise, the busy period is a successful busy period.

3.7.6. Reliability

Ethernet is probabilistic. Frames may be lost due to interference with other frames, impulse noise on the channel, an inactive receiver at the frame's intended destination, or purposeful discard. Ethernet gives its best efforts to transmit frames successfully, but it is the responsibility of the processes in the source and destination stations to take precautions necessary to assure reliable communication.

It is very costly and dangerous to promise "error-free" communication. Ethernet attains both economy of transmission and high reliability averaged over many frames. Removing the responsibility for reliable communication from the transport mechanism allows the tailoring of reliability to the application and placement of error recovery where it will do the most good. This policy becomes more important as Ethernet networks are interconnected in a hierarchy of networks through which frames must travel further and suffer greater risks.

Ethernet does provide five mechanisms for reducing the probability and cost of losing a frame. These are

Carrier detection

A frame's bits are phase encoded as they are placed on the channel, which guarantees that there is at least one transition on the channel during each bit time. The passing of a frame on the channel can therefore be detected by listening for its transitions. To use a radio analogy, we speak of the presence of carrier as a frame passes a transceiver. Because a station can sense the carrier of a passing frame, it can delay sending one of its own.

With carrier detection, deference can be implemented: no station will start transmitting while hearing carrier. With deference comes acquisition: once a frame transmission has been in progress for the end-to-end propagation time, all stations are hearing carrier and are deferring; the channel as been acquired and the transmission will complete without an interfering collision.

Interference detection

Interference is detected, as stated before, by the transceiver. Interference detection has three advantages. Stations can know that a frame has been damaged and schedule it for retransmission immediately. Second, interference intervals on the channel are limited to a maximum of the roundtrip propagation delay. Third, the frequency of detected interference is used to adjust retransmission intervals and to optimize the channel efficiency.

Frame error detection

As a frame is placed on the channel, a checksum is computed and appended. As the frame is read from the channel, the checksum is recomputed. Frames which do not carry a consistent checksum are discarded. In this way transmission errors, impulse noise errors, and errors due to undetected interference are caught at the frame's destination.

Truncated frame filtering

Truncated frames are the result of collisions. To reduce the processing load that the rejection of such obviously damaged frames would place on the listening station software, truncated frames are filtered out in the hardware.

Collision consensus enforcement

When a station determines that its transmission is experiencing

interference, it momentarily jams the channel to insure that all other participants in the collision will detect interference and be forced to abort.

3.8. Frame Structure

Figure 3-10 depicts the format of the frame generated by the Ethernet protocol. It consists of the following fields:

- *Destination address (DA):* Specifies the station(s) for which the frame is intended. It may be a unique physical address, a multicast-group address, or a global address. The choice of a 16 or 48 bit address is an implementation decision, but must be the same for all stations on a particular LAN.
- *Source address (SA):* Specifies the station that sent the frame. The SA size must equal the DA size.
- *Type:* Determines which client protocol the frame is for.
- *Data:* Actual information being transferred.
- *Frame check sequence (FCS):* A 32 bit cyclic redundancy check based on all fields starting with the destination address.

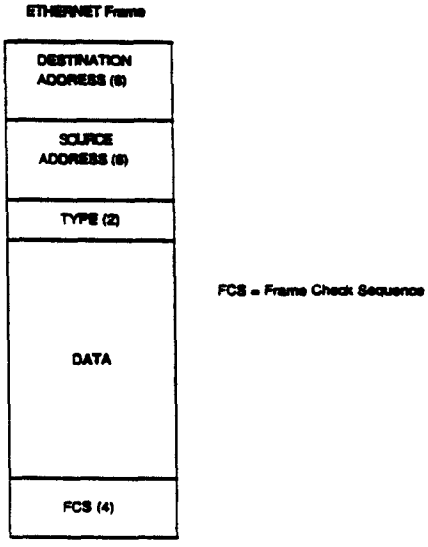


Figure 3-10: Ethernet Frame Structure

Ethernet does not include a length field, and therefore, expects some higher layer to ensure proper collision detection operation. Ethernet provides unacknowledged connectionless service. This service is a datagram type of service that simply allows for sending and receiving frames, with no form of acknowledgement to assure delivery.

The type field is significant. It is the 13th and 14th bytes in an Ethernet frame. Table 3-4 displays protocol types defined in the Ethernet V2.0 protocol specification. This information was taken from a file used in conjunction with the LAN Traffic Monitor (described in the last section of this chapter) here at Fermilab.

PROTOCOLS SEEN AT FERMILAB

Value	Name	Description
90-00	LoopBack	DEC Ethernet Cross-Company Loopback protocol
60-00	DEC_C_UA1	DEC unassigned
60-01	DEC_MOPDL	DEC Maintenance Operations Protocol DownLoad
60-02	DEC_MOPRC	DEC Maintenance Operations Protocol Remote Console
60-03	DEC_NET	DEC network or routing layer
60-04	DEC_LAT	DEC Local Area Transport for terminal servers
60-05	DEC_DIAG	DEC Diagnostics protocol tpe
60-06	DEC_C_USE	DEC protocol type reserved for customers
60-07	DEC_LAVC	DEC System Communications Architecture
60-08	DEC_UA2	DEC unassigned
60-09	DEC_UA3	DEC unassigned
80-3D	DEC_ENCRYP	DEC Ethernet Encryption Protocol
80-3F	DEC_LTM	DEC LAN Traffic Monitor
80-38	DEC_BRIDGE	DEC Bridge Management Protocol
80-39	DEC_C_UA4	DEC unassigned
80-3A	DEC_C_UA5	DEC unassigned
80-3B	DEC_C_UA6	DEC unassigned
80-3C	DEC_DNS	DEC Distributed Name Service
80-3E	DEC_C_UA9	DEC unassigned
80-40	DEC_C_UA11	DEC unassigned
80-41	DEC_MS/DOS	DEC VMS Services for MS-DOS
80-42	DEC_C_UA13	DEC unassigned
02-00	XRX_PUP	Xerox Pup Protocol is now revoked!
02-01	XRX_PUPAT	Xerox PUP Address Translation
06-00	XNS_IDP	Xerox NS IDP
08-00	DOD_TCPIP	Dept. of Defense Transport/internet Protocol
08-01	X.75_IP	Internet X.75 Protocol, CCITT
08-02	NBS_IP	NBS Internet Protocol, X.75 Transport
08-03	ECMA_IP	ECMA version of ISO transport
08-04	CHAOSNET	
08-05	X.25_LEV3	X.25 Level 3, CCITT
08-06	TCIP_ARP	TCP/IP Address Resolution Protocol
08-07	XRX_XNS	XEROX NS Compatability Protocol
08-1C	SYMBOLICS	SYMBOLICS Private Protocol
10-00	BERK_IP_0	Berkley Trailer IP Encapsulation
10-01	BERK_IP_1	Berkley Trailer IP Encapsulation
10-02	BERK_IP_2	Berkley Trailer IP Encapsulation
10-03	BERK_IP_3	Berkley Trailer IP Encapsulation
10-04	BERK_IP_4	Berkley Trailer IP Encapsulation
10-05	BERK_IP_5	Berkley Trailer IP Encapsulation
10-06	BERK_IP_6	Berkley Trailer IP Encapsulation
10-07	BERK_IP_7	Berkley Trailer IP Encapsulation
10-08	BERK_IP_8	Berkley Trailer IP Encapsulation
10-09	BERK_IP_9	Berkley Trailer IP Encapsulation
10-0A	BERK_IP_10	Berkley Trailer IP Encapsulation

PROTOCOLS SEEN AT FERMILAB, CONTINUED		
Value	**Name**	**Description**
10-0B	BERK_IP_11	Berkley Trailer IP Encapsulation
10-0C	BERK_IP_12	Berkley Trailer IP Encapsulation
10-0D	BERK_IP_13	Berkley Trailer IP Encapsulation
10-0E	BERK_IP_14	Berkley Trailer IP Encapsulation
10-0F	BERK_IP_15	Berkley Trailer IP Encapsulation
12-34	DCA_MULTI	DCA Multicasts
16-00	VALID_MH	VALID machine protocol
43-21	THD_DIDDLE	Tom Dunigan "test Protocol type
52-08	BBN_SIMNET	BBN SIMNET Private
70-00	UNG_BASS0	Ungermann-Bass Server ?
70-01	UNG_BASS1	
70-02	UNG_BASS2	Ungermann-Bass Hello?
70-07	OS9NET_1	0S-9 Network Protocol
70-09	OS9NET_2	OS-9 Network Protocol
70-30	PROTEON	
80-03	CRON_VLN	CRONUS VLN
80-04	CRON_DIR	CRONUS Direct
80-05	HP_PROBE	Hewlett Packard Probe Protocol
80-06	NESTAR	
80-10	EXCELAN	
80-35	REV_ARP	TCP/IP Reverse Address Resolution Protocol
80-5B	STAN_EXP	Standford V Kernel, Experimental
80-5C	STAN_PRO	Standford V Kernel, Production
80-7C	MRT_I_NODAL	Merit Internodal Protocol
80-80	VIT_B_MAN	VitaLink XNS Bridg Mgmt Protocol
80-9B	APPLETLK	Apple-Talk
80-C0	DCA_HELLO	CDA Broadcast
80-C1	DCA_TALK1	DCA Protocol for DCA Ether Boxes
80-C2	DCA_TALK2	DCA Protocol for new etherboards
80-C6	PNP_PACER	IBM Pacerlink Pacer Network Protocol
80-F3	APPLE_TSS	TSS AppleTalk
90-01	BRG_BRG1	BRIDGE Bridge Multicasts and HELLOs
90-02	BRG_SERV	BRIDGE Terminal Servers
90-03	BRG_BRG3	BRIDGE Bridge Inter-bridge communications
FF-00	BBN_VITA	BBN VITAL-LanBridge cache wakeup

Table 3-4: Protocols at Fermilab

3.9. IEEE 802.3 Compared to Ethernet

An experimental version of Ethernet was developed in the mid-1970's by Xerox Corporation. The network used the CSMA/CD protocol on a 3-Mbps baseband coaxial cable. In 1980, Version 1.0 of the Ethernet specification was published jointly by Xerox, DEC and Intel. The principal difference from the experimental system was the use of a 10-Mbps data transfer rate. In 1982 Version 2.0 was released. This later version incorporates changes and enhancements at the physical layer introduced during elaboration of the IEEE 802.3 standard.

IEEE 802.3 is a member of the IEEE 802 suite of protocols. It is a MAC standard that is used with the IEEE 802.2 Logical Link Control (LLC) standard to provide more elaborate functions at the data link layer. Ethernet and IEEE 802.3 both use the CSMA/CD, 1-persistent MAC protocol. Often Ethernet and IEEE 802.3 are used interchangeably although, there are a few subtle differences in electrical functions, link layer control, and frame format.

3.9.1. Electrical Functions

Ethernet 2.0 and 802.3 include a "heartbeat" function. This is a signal sent from the Ethernet transceiver to the station that confirms that the collision signaling is working and connected to the station (signal quality error signal). Without this signal, the station is unsure whether the frame was actually sent without a collision, or whether a defective transceiver failed to report properly a collision. Ethernet 1.0 does not have this function. Both 802.3 and Ethernet 2.0 have a *jabber* function. This is a self-interrupt capability allowing a transceiver to inhibit transmitted data from reaching the medium if the transmission occurs for longer than the maximum frame size.

3.9.2. IEEE 802 and the OSI/RM

The IEEE 802 standard relates to the OSI/RM somewhat differently than the Ethernet standard does. The IEEE 802 standard breaks up the data link layer into 2 separate layers : the Logical Link Control (LLC - 802.2) layer and the Medium Access Control (MAC - 802.3) layer as in Figure 3-11. One reason for this division is that the logic required to manage access to a multiple-source, multiple-destination link is not found in traditional data link layer control. Also, for the same LLC, several MAC options may be provided. The 802.3 MAC layer employs CSMA/CD as does Ethernet. The 802.2 LLC layer performs the traditional data link layer functions, error control and flow control, but also provides some network layer functions. This allows 802.2\3 to provide 3 levels of service:

- *Unacknowledged connectionless service:* This is a datagram service that simply allows for sending and receiving frames.

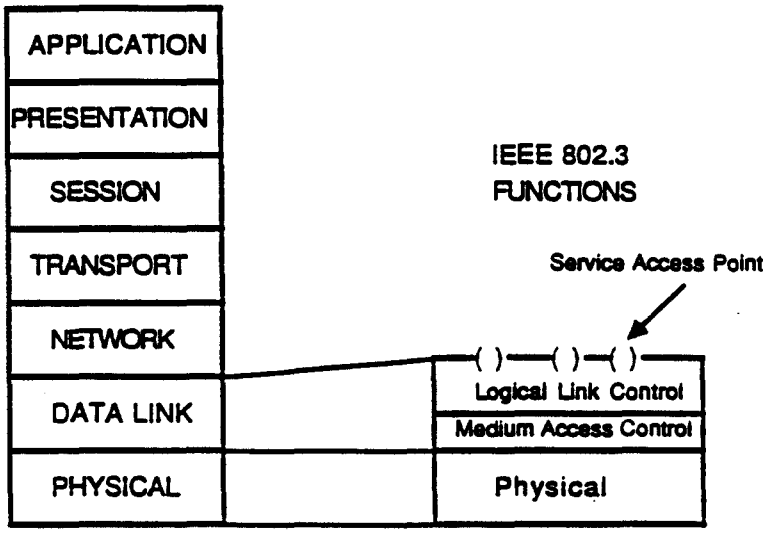


Figure 3-11: IEEE 802 Protocol Layers Compared to OSI/RM

- ***Connection-oriented service:*** This provides a logical connection between service access points. It provides flow control, sequencing, and error recovery.
- ***Acknowledged connectionless service:*** This is also a connectionless service, but provides for acknowledgment, relieving higher layers of this burden.

Recall that Ethernet provides only unacknowledged connectionless service.

3.9.3. Frame Formats

The IEEE 802.2\3 format differs significantly from the Ethernet format. IEEE 802.2\3 frame includes a length field which specifies the number of LLC bytes that follow. See Figure 3-12. In order to provide the different levels of service, the LLC layer has a frame format of its own which is encapsulated in the MAC frame format. The MAC frame format is known as the IEEE 802.3 frame format. The 802.2 LLC data consists of the source and destination service access points, a control field, and the data being transmitted. A service access point is a logical interface between two adjacent layers and is a similar concept to the Ethernet type field. The control field is the mechanism by which the levels of service are provided. Ethernet has no control field. The *pad* field contains a sequence of bytes to assure that the frame is long enough for proper collision-detection operation. The frame must be at least long enough to require one slot time to transmit. Ethernet expects some higher layer to perform this padding function to ensure a minimum length frame.

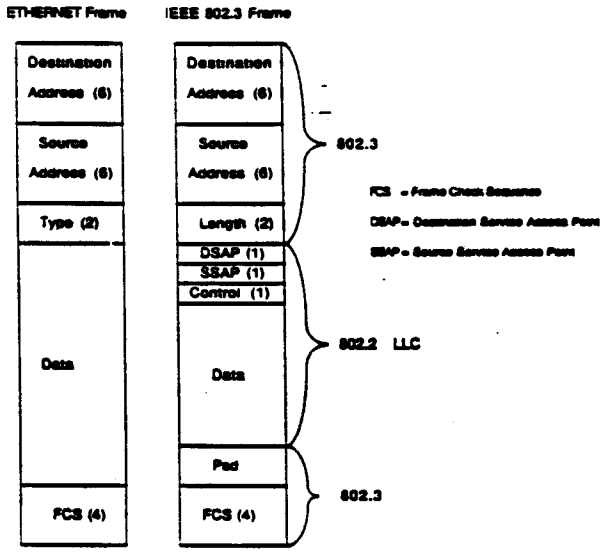


Figure 3-12: Comparison of IEEE 802.3 and Ethernet Frame Formats

3.10. Performance

The level of performance provided by Ethernet is of much interest. Quantifying network performance can be difficult. Numerous measures of performance exist and there are many factors that affect performance. LAN traffic itself is highly variable and unpredictable. Many studies of Ethernet have been done and the results are nearly always identical. In theory and practice, Ethernet provides a very high level of performance for most networking applications.

This section has 3 objectives:

- To give the reader some insight into the different measures of and the factors that affect Ethernet's performance.
- To present a summary of past Ethernet performance analysis results and studies..
- To explain how Fermi's Ethernet is analyzed and how it performs.

3.10.1. Measures of Performance

An obvious and important performance measure from the user point of view is response time. The performance of a network, as with most interesting computer systems, cannot by quantified with a single dimension. Performance measures, which also affect response time, include :

- *Average delay:* the average time it takes to send a frame, measured from the time the host first wishes to acquire the channel.
- *Throughput:* the fraction of the nominal network bandwidth that is actually used for carrying data. Frame headers are considered useful data in calculating this value.
- *Channel Capacity:* the maximum achievable throughput for a given set of parameters. Capacity is a function of such parameters as frame length and network length.
- *Fairness:* in a fair network, each host with pending traffic should have an equal probability of acquiring the channel (this is not an equal share of the bandwidth, since hosts use differing frame sizes).
- *Stability:* if the throughput actually drops at high loads then the network is said to be unstable in that region.

3.10.2. Factors Affecting Performance

The performance of an Ethernet depends on a variety of parameters, in addition to those fixed by the specification. The fixed parameters of the standard Ethernet are described in section 6 of this chapter. In addition to the fixed parameters, performance depends on several other factors determined by the users of the network.

- *frame length distribution:* Frame length can have a profound affect on network performance. If all frames are small, performance will not be optimum. If a few larger frames are mixed in, performance will improve dramatically.
- *Actual number of hosts:* The number of hosts on an Ethernet can vary tremendously. With more hosts, it is possible to offer a greater load to the network.
- *Arrival rate of frames:* Most hosts are unable to transmit or receive more than a few hundred packets per second. This places a lower bound on the value of average channel access time that actually affects performance.
- *Actual length of cable:* The actual length of an Ethernet may be far shorter than the specification allows. This means collisions are detected far sooner that the worst-case propagation delay would imply.

3.10.3. Offered Load

Performance measures are usually described as a function offered load. Offered load can be defined as the average number of attempted frame transmissions per frame transmission time. It is the actual load or traffic demand presented to the local network. Retransmissions caused by collisions are also included in this number. Most normally loaded Ethernets have an offered load of much less than 1. If one defines the offered load at each host as the fraction of the network bandwidth that the host would use if it had complete access to the network, then the offered load on the network as a whole is simply the sum of the offered loads at each host. Each host's offered load is less than or equal to 1. The network offered load can therefore be greater than 1, although throughput cannot.

The nature of the traffic offered by the user devices is a major factor in determining the performance of the network. Unfortunately, the offered traffic load is a highly variable quantity affected by a large number of factors. If consideration is given to a local network with, for example, 100 nodes connecting typical user devices, it is easy to imagine the extreme variability of the traffic offered to the network. Any one device may be inactive for long periods and then require a large data rate for a short time. Variations between different hours of day and night are certain to occur. Averages taken over different intervals such as weeks, days, hours, or seconds can show large differences. Traffic fluctuates relatively close to, but below average for long periods of time and then jumps to values much greater than the average for short periods of time. Users tend to use the network intermittently.

3.10.4. Past Studies of Ethernet

Shoch and Hupp did one of the first performance studies of an Ethernet network. The network was called an experimental Ethernet because the data rate was only 3 Mbps and the traffic was artificially generated. Yet, the results closely reflect the performance of a running 10 Mbps running network. Shoch and Hupp found that as total offered load increases from 0-90% channel utilization matches it perfectly: all traffic gets out correctly. As the offered load moves above 90%, the channel utilization flattens out at a level above 96%. Figure 3-13 displays these findings. An Ethernet system under high load shows no instability: the throughput curve does not decline as the total offered load increases. The Ethernet control discipline is also very fair in its allocation of channel capacity.

The ethernet system remains stable even under extreme overload conditions. Figure 3-14 shows the results of experiments with as many as 90 hosts sending medium to large size frames (each offering up to a 95% network load). Ethernet utilization remains high and shows no signs of suddenly decreasing or becoming unstable. With more than 1 continuously queued source, some of the traffic cannot be accommodated, and each station can get only some fraction of its nominally desired bandwidth (100% per host). With 90 hosts sending, for example, the average utilization per host is 1.1% ranging from about 0.9-1.3%. The

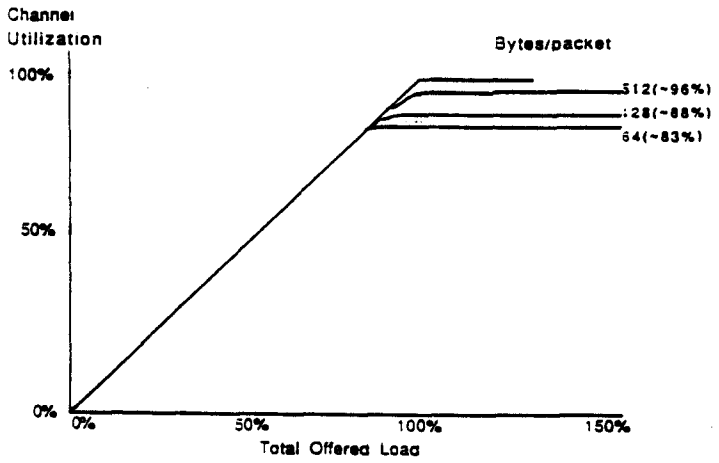


Figure 3-13: Measured Utilization of the Ethernet Network under High Load

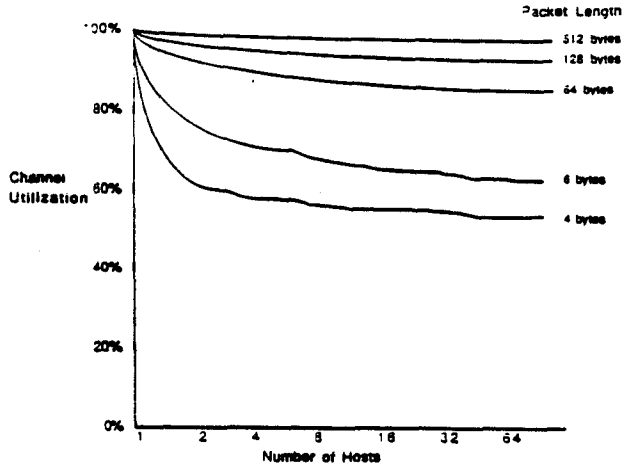


Figure 3-14: Measured Utilization with Continuously Queued Sources

total utilization starts out very high and decreases somewhat as additional hosts are added.

The following figures show the results of experiments on a real 10 Mbps Ethernet. Figure 3-15 conforms to Figure 3-13 and shows that bit rate increases with increasing packet size. Figure 3-16 illustrates that fairness increases as the number of hosts increases. Figure 3-17 shows that the average transmission delay increases linearly with increasing number of hosts (i.e. offered load). Figure 3-18 shows excess delay, a direct measure of inefficiency. It is derived from the delays plotted in Figure 3-17. The ideal time to send one packet and wait for each other host to send one packet is subtracted from the measured time. The time that remains was lost participating in collisions. Notice that it increases linearly with increasing number of hosts.

Table 3-5 presents repesentative performance figures (efficiency) for an Ethernet with the indicated packet sizes and number of continuously queued stations.

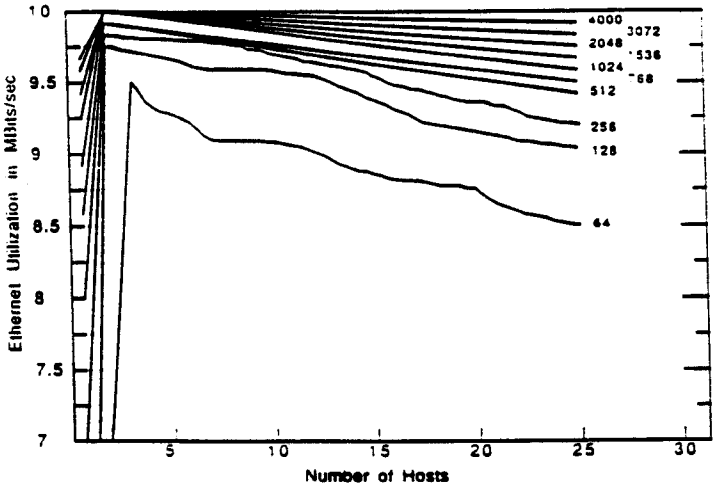


Figure 3-15: Total Bit Rate

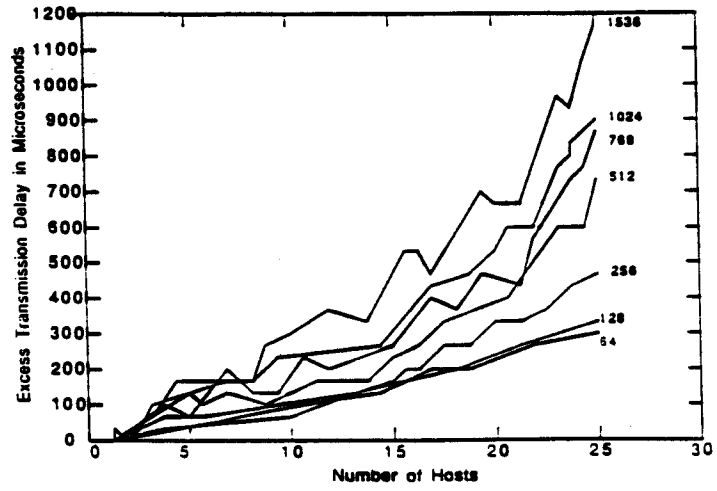


Figure 3-16: Std. Dev. of Bit Rate

3.10.5. Ethernet at Fermilab

The Data Communications Group has a number of tools which monitor the performance of the on site network and give various statistics. The most interesting of these tools is the LAN Traffic Monitor (LTM). The LTM is menu driven, allowing the user to display almost any statistics. The information in the following tables was taken from the LTM and resembles the computer screen. The statistics from the *current* line of Table 3-6 were taken curing the latest 2.95 second interval. On the *peak* line are statistics from the peak 2.95 second interval oc-

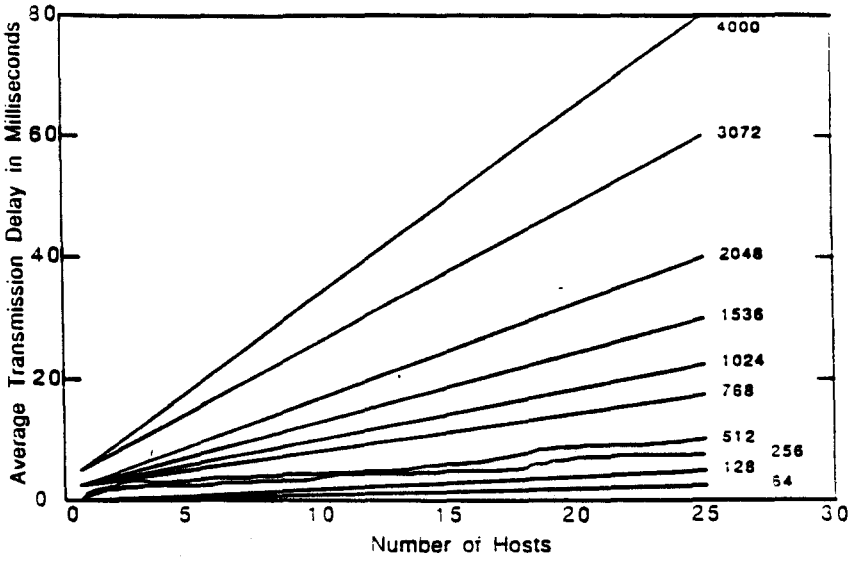


Figure 3-17: Average Transmission Delay

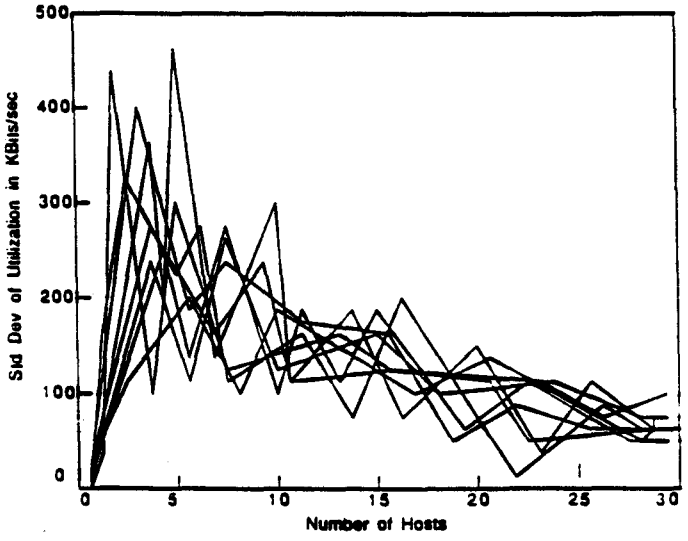


Figure 3-18: Excess Transmission Delay

curring since the listener counters were zeroed (over 20 days in this case). The *avg* line gives statistics from the latest 29 second interval. Also, the error statistics near the bottom of the table, show how few errors occurred. Even these few errors were detected.

Tables 3-7 - 3-9 show the three different types of network utilization and throughput statistics : current, peak and long term. The current statistics in Table 3-7 represent the most recent 2.95 second interval. Multicast frames are frames targeted for more than one destination. The network utilization from the most recent 2.95 second interval was only 4.14%. The throughput was 100%.

Q	P = 4096	P = 1024	P = 512	P = 48
1	1.0000	1.0000	1.0000	1.0000
2	0.9884	0.9552	0.9143	0.5000
3	0.9857	0.9447	0.8951	0.4444
4	0.9842	0.9396	0.8862	0.4219
5	0.9834	0.9367	0.8810	0.4096
10	0.9818	0.9310	0.8709	0.3874
32	0.9807	0.9272	0.8642	0.3737
64	0.9805	0.9263	0.8627	0.3708
128	0.9804	0.9259	0.8620	0.3693
256	0.9803	0.9257	0.8616	0.3686

Table 3-5: Ethernet Efficiency

LAN Traffic Monitor 17-JAN-1990 14:51:22
Listener Uptime 20 days 05:37:59

	Utilization	Frames/Sec	Multicast
Current	7.7 %	649	6 %
Peak	34.4 %	1296	62 %
Avg (29 sec)	6.9 %	734	5 %

frames seen by listener	420286689	100%
802.3 frames	120139	.0286%
CRC check errors	1081	.0003%
Oversize frame errors	0	0%
MC source address errors	6	0%

Table 3-6: Network Traffic Summary Display

Table 3-8 gives peak interval statistics. In other words, it shows the highest values gathered for each report and averaging interval since the listener started. The peak utilization over a 2.95 second interval was 34.35%. The peak utilization over a 29 second interval was 30.13%.

Table 3-9 shows the utilization and throughput since the listener has been active. The total network utilization has been 2.92% and throughput has been 100%.

Figures 3-19 and 3-20 show the graphical displays of network utilization available from the LTM. Figure 3-19 is a graph of average network utilization. Figure 3-20 is a graph showing peak network utilization. Time is represented on the horizontal axis and utilization is represented on the vertical axis.

Figure 3-21 displays a graph which tracks usage of 3 different protocols. Table 3-10 lists the top ten protocol types seen on the network by the listener. This information was gathered from the type field of each frame itself. This table

LAN Traffic Monitor 30-JAN-1990 13:30:29
Listener Uptime 00 days 01:06:06

UTILIZATION	Byte Count	Bytes/sec	% Util
multicast	10416	4105	0.39
single dest	128178	45556	3.75
total <bytes>	138594	49661	4.14
THROUGHPUT	**Frame Count**	**Frames/sec**	**% Total**
multicast	116	39	8.08
single dest	1436	204	91.2
total <frames>	1552	526	100

Table 3-7: Current LAN Utilization and Throughput

LAN Traffic Monitor 17-JAN-1990 14:51:22
Listener Uptime 20 days 05:37:59
Report Interval : 2.95 sec Averaging Interval : 29 sec

LAN Utilization Statistics	Peak Value
for all report intervals	34.35%
for all averaging intervals	30.13%
Frames/second	
for all report intervals	1296
for all averaging intervals	1082
Multicast : total ratio	
for all report intervals	62%
for all averaging intervals	57%

Table 3-8: Peak LAN Utilization and Throughput

LAN Traffic Monitor 17-JAN-1990 14:51:22
Listener Uptime 20 days 05:37:59

UTILIZATION	Byte Count	Bytes/sec	% Util
multicast	7519994368	4301	0.34
single dest	56205447744	32149	2.57
total <bytes>	63725442112	36451	2.92
THROUGHPUT	**Frame Count**	**Frames/sec**	**% Total**
multicast	63811557	36	15.19
single dest	356149597	204	84.81
total <frames>	419961154	240	100

Table 3-9: Longterm Utilization and Throughput

gives a general view of what exactly the network is being used for and can prove very helpful.

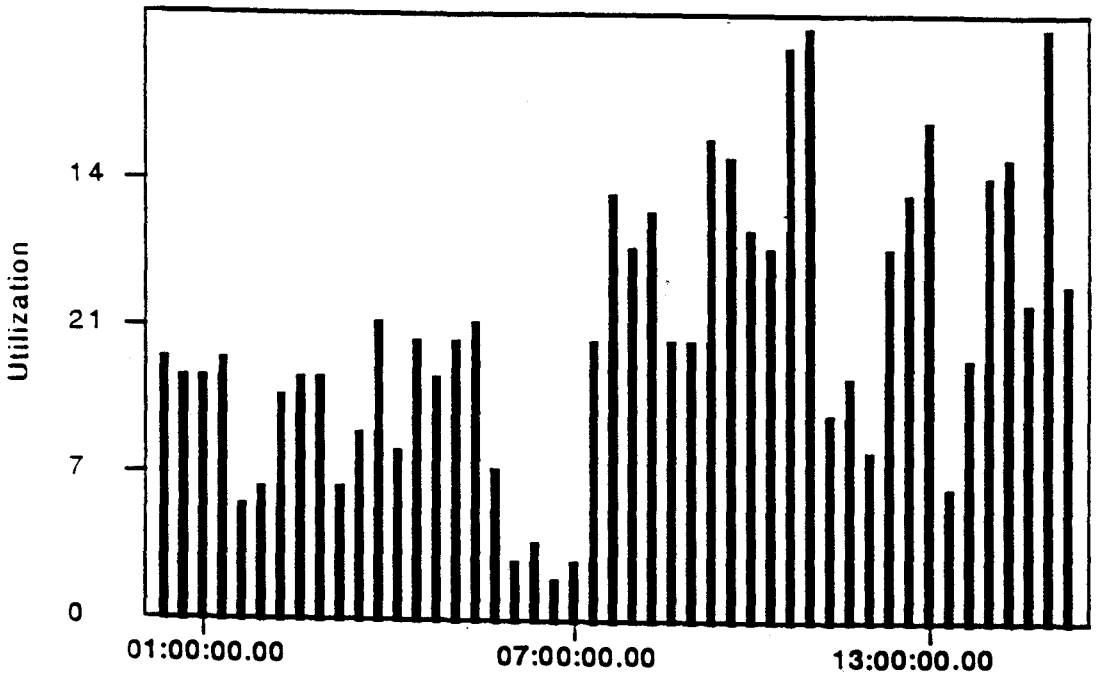


Figure 3-19: Average Utilization

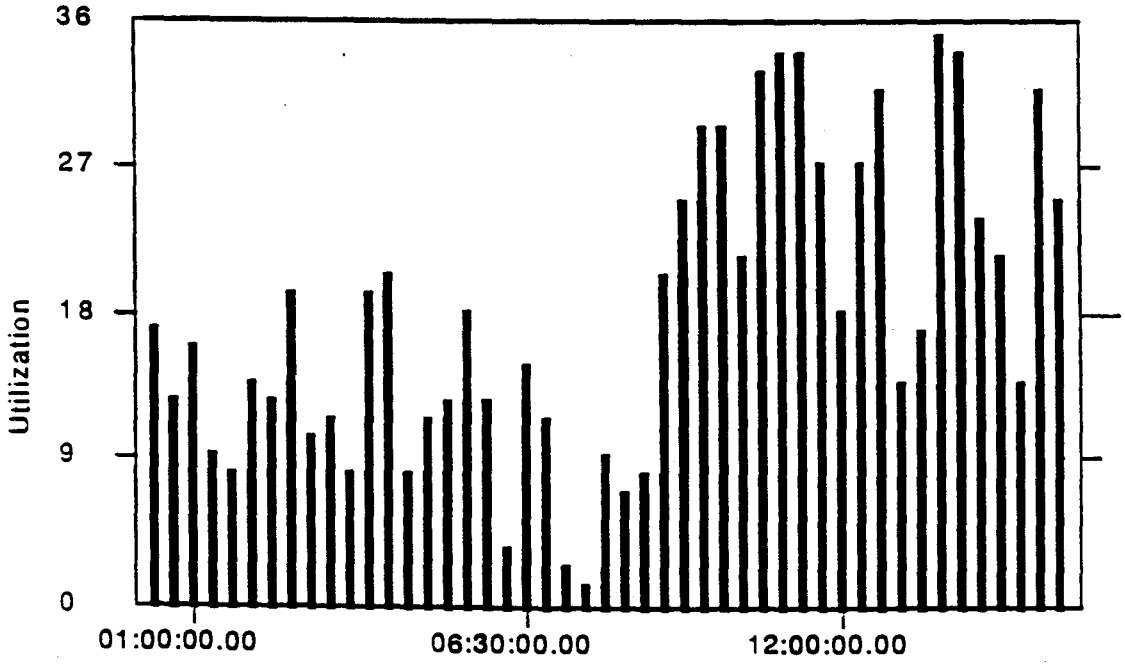


Figure 3-20: Peak Utilization

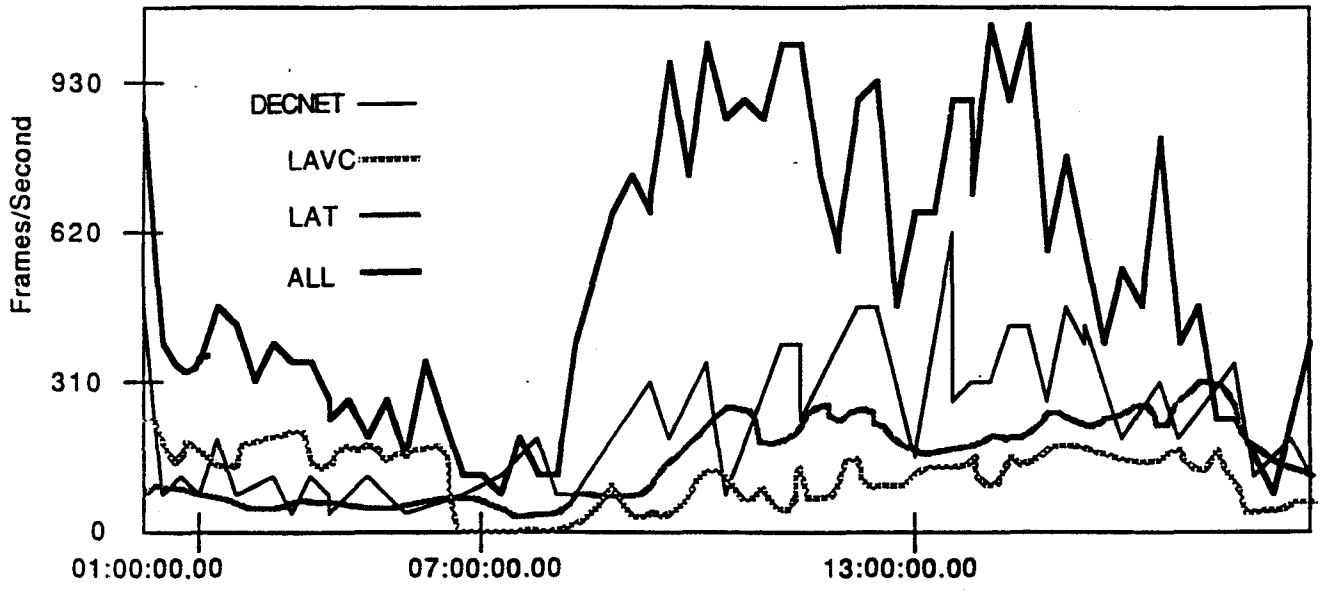


Figure 3-21: Protocol Tracking

LAN Traffic Monitor 30-JAN-1990 13:30:29 Listener Uptime 00 days 01:06:06		
Type Name	**Frame Count**	**Percentage**
DEC_LAT	1136734	47.3 %
DEC_NET	934020	39.3 %
DEC_LAVC	269734	11.1 %
DOD_TCPIP	30277	1.3 %
APPLETLK	6041	.3 %
DEC_BRIDGE	4491	.2 %
DEC_LTM	4338	.2 %
DEC_MOPRC	3597	.2 %
	3167	.1 %
TCPIP_ARP	739	0 %
other	675	0 %

Table 3-10: Top 10 Protocol Types

The packet size distribution is displayed in Table 3-11. The performance of a network is greatly improved when there are a few large frames mixed in with shorter ones.

LAN Traffic Monitor 17-JAN-1990 14:51:22 Listener Uptime 20 days 05:37:59				
64	100	300	800	1518
15%	60%	13%	11%	1%

Table 3-11: Frame Size Distribution

3.10.6. Performance Summary

Ethernet is capable of good performance even at high offered load. The performance of CSMA/CD is sensitive to frame size and the time it takes to detect a collision. Users must keep these parameters in mind. As the number of hosts increases, the fairness increases. Bit rate increases with increasing frame size. Bit rate decreases with increasing number of hosts. The average transmission delay increases linearly with increasing number of hosts (i.e. offered load). On shorter networks, efficiency increases as collision resolution time decreases.

Some hypotheses about the performance of an ethernet system:

- The error rates are very low and very few frames are lost.
- The loss of bandwidth due to collisions or electrical noise on a lightly loaded Ethernet is, for all intents and purposes, zero.
- Under normal load, transmitting stations rarely have to defer and there are very few collisions. Thus, the access time for any station attempting to transmit is virtually zero.
- Frame arrivals averaged over longer periods of time (i.e. minutes) indicate that between 5 and 10 times more frames are transmitted on the LAN during peak periods as compared to average periods.
- Under heavy load there are more collisions, but the collision detection and resolution mechanisms work well, and channel utilization remains very high - approaching 98%. In addition, the utilization remains very stable, and channel sharing is quite fair.
- Even under extreme overload, the Ethernet channel does not become unstable.

3.11. Managing Ethernet

Managing any LAN is a complex and challenging process. A LAN is constantly changing and growing. Many different tools have been developed to help network managers cope.

3.11.1. Tools At Fermilab

The following network management tools are in use here at Fermilab. A brief description follows the name of the tool. The description is only meant to give an overview of the functions these tools perform and not an in depth analysis.

- **LAN Traffic Monitor** lets network managers look at the overall network traffic level and then zoom in on areas of interest: Who are the top ten talkers, currently and long term? What are the throughput and utilization rates? What are the multicast addresses?

  The LTM consists of unique monitoring software that can capture and present traffic data from an extended Ethernet. The LAN Traffic Monitor VMS contains the LTM Listener software, which counts and classifies Ethernet traffic using a LAN_Bridge as a monitoring device; the LTM User Interface, which collects and displays data received from the LTM Listener. It is designed to work at full Ethernet speed, thus providing realtime data on Ethernet LAN throughput and utilization, allowing managers to actively monitor the Ethernet usage. Data can be collected on any protocol type.

  Managers interact with the application software through a menu structure. LAN utilization can also be displayed graphically based on the most recent traffic data received from the LTM Listener.

- **NMCC/VAX ETHERnim** (Ethernet network integrity monitor) is a network maintenance application used to test segments at the user level. The software has the capability to isolate and identify a problem. Permanent databases containing basic information about each node connected to the Ethernet can be constructed. It is also possible to maintain a topological view of the network and monitor the network to gather statistics.

- **NMCC/DECnet Monitor** is a comprehensive data collection and monitoring system for network performance data. It is used here at Fermilab to keep track of the off-site DECnet links. The kernel collects data from the network using polling (gives status, characteristics and traffic/error counters) or event-logging (allows remote nodes to log events automatically to the kernel offering details on status changes and traffic/error counters as they occur) procedures. The User Interface subsystem displays selected statistics and graphs.

- **LAN Analyzer** is a tool used to examine Ethernet frames individually. It taps directly onto the Ethernet allowing network managers to grab individual frames and look inside of them. When an

unidentified problem exists on the network, managers can examine frames to check for addressing errors or truncated packets or anything unusual that may help identify and thus ultimately solve the problem. The LAN Analyzer includes a STATS package which allows managers to monitor trends on the Ethernet such as utilization, errors, traffic between nodes, etc. It is similar to the LTM, but a bit more sophisticated.

- **NETSTAT** is a network management tool used to monitor the status of DECnet nodes and LAN Bridges. These are two modes in use right now at Fermilab. The display is a color grid with one bridge or node per block on the grid. The blocks on the grid are green when the bridge or node status is up and running correctly. If NETSTAT detects a problem, the block representing the affected node or bridge becomes red, thus signaling the operator of the problem.

3.12. References and Further Reading

1. Boggs, D.R., Mogul, J.C., and Kent, C.A. SIGCOM 1979. Measured Capacity of an Ethernet: Myths and Reality, SIGCOM, 1979, pp. 222-234.

2. Hammond, J.L. and O'Reilly, P.J.P.. *Performance Analysis of Local Computer Networks.* Addison-Wesley, Reading, MA, 1986.

3. Metcalfe, R.M. and Boggs, D.R. "Ethernet: Distributed Packet Switching for Local Computer Networks". *Communications of the ACM 19*, 7 (July 1976), 395-404.

4. Shoch, J.F., and Hupp, J.A. "Measured Performance of an Ethernet Local Network". *Comunications of the ACM 23*, 12 (December 1980), 711-721.

5. Stallings, W.. *Handbook of Computer-Communications Standards.* Macmillan, New York, 1987.

6. Stallings, W.. *Local Networks An Introduction.* Macmillan, New York, 1987.

Chapter 4

Understanding FDDI

4.1. Need for High-Speed Networks

First-generation Local Area Network (LAN) technology such as ANSI/IEEE 802.3 (ISO 8802.3) and ANSI/IEEE 802.5 (ISO 8802.5), commonly referred to as Ethernet and Token-Ring respectively, are characterized by the following:

- Signaling rates in the range of 1 to 20 Mbps,
- Copper-based media,
- VLSI implementations of physical and data link layers,
- Underdeveloped network management and security standards.

Initially, such technology was used to provide basic services such as file transfer and electronic mail. As services and applications evolve the network bandwidth is proving to be inadequate in meeting the requirements of high-performance workstations, graphics and CAD/CAM modeling systems and the need for high-speed access of distributed computing resources. In general, one can note that physical signaling rates seem to be doubling every 14 months and computing power is doubling every 12 months.

4.2. FDDI: Standard and Definition

The Fiber Distributed Data Interface (FDDI) is a standard for high-speed ring local network developed by the X3T9.5 Accredited Standards Committee of the American National Standards Institute (ANSI). Like the IEEE 802.5 Token-Ring standard, FDDI employs the token ring algorithm. Figure 4-1 depicts the architecture of the FDDI standard.

This standard specifies the Medium Access Control (MAC) and the physical layers for a 100-Mbps, counter-rotating, token ring, optical fiber LAN. FDDI supports both synchronous and asynchronous traffic. The use of IEEE 802.2 Logical Link Control (LLC) standard is assumed. The FDDI standard is in four parts:

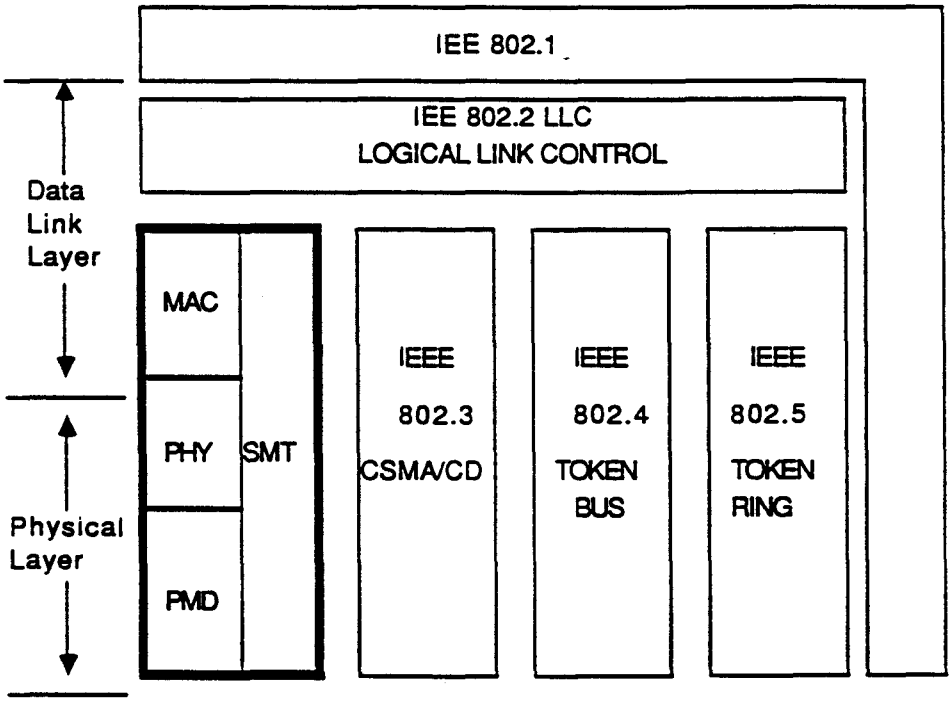


Figure 4-1: FDDI Architecture

- Medium access control (MAC)
- Physical protocol (PHY)
- Physical medium dependent (PMD)
- Station management (SMT)

4.2.1. Specification

FDDI is the first standard LAN that specifies the use of optical fiber as its primary media. FDDI's data-transfer rate is 100 Mbps over distances up to 200 kilometers (km) with up to 1000 stations connected. The design specification requires a Bit Error Rate (BER) of no more than 1 error per 2.5 X 10^{10} bits. FDDI uses both single and multimode fibers. The maximum distance between two nodes when multimode fibers are used, is 2 km. The motivation for providing support for single mode fibers was to provide support for a maximum station separation distance of 40 km.

4.2.2. Counterrotating Rings

The FDDI cabling consists of two fiber rings one transmitting clockwise and the other transmitting counterclockwise. If either ring breaks the other is used as a backup. If both rings break at the same point, for example, due to a fire or other accident in the cable duct, the two rings can be joined into a single ring approximately twice as long. Each station contains relays that can be used to join the two rings or bypass the station in the event of station problems.

4.2.3. Default Parameters

Table 4-1 summarizes FDDI default parameters.

Parameter	Value
Maximum data rate	100 Mbps
Signal rate	125 Mbaud
Physical connections	1000
Total path length	200 km. With dual-ring, each ring is limited to 100 km.
Maximum station separation	With standard 62.5/125 um) limited to 2 km; with Single Mode Fiber (SMF) category 1 laser limited to 10 km, category 2 laser limited to 40 km.
Station types	Dual Attachment Station (DAS); Single Attachment Station (SAS); Concentrators.
Maximum Ring Latency (D_Max)	1.773 ms
Total cable latency	1.017 ms (200 kms x 5.085 ns/meter)
Maximum station delay	1.164 us (GOSIP requirement)
Target Token Rotation Time (TTRT)	Default should be 8 ms.

Table 4-1: FDDI Default Parameters

4.3. Origin of FDDI

In 1983 work was completed on the Locally Distributed Data Interface (LDDI), a 50 Mbps "channel-like" interface based on Control Data Corp.'s Loosely Coupled Network (LCN). LDDI was a broadband network that supported seven nodes and physically connected systems over a total distance of 1 km. It was obvious even then that a higher bandwidth was necessary and that the connectable number of nodes should be on the order of hundreds or even thousands. The goal was to extend the communications on the computer floor, to allow disk drives to be located at a significant distance from the mainframe, and to support distributed multiprocessor computing environments.

Thus, the ANSI X3T9.5 membership originally consisted of representatives of manufacturers interested in developing a high-bandwidth serial "channel" interface for mainframes. Sperry was responsible for submitting the first proposals in mid-1983.

4.4. FDDI Topology: An Example

Figure 4-2 illustrates how an FDDI ring is used as a backbone network to connect LANs and computers.

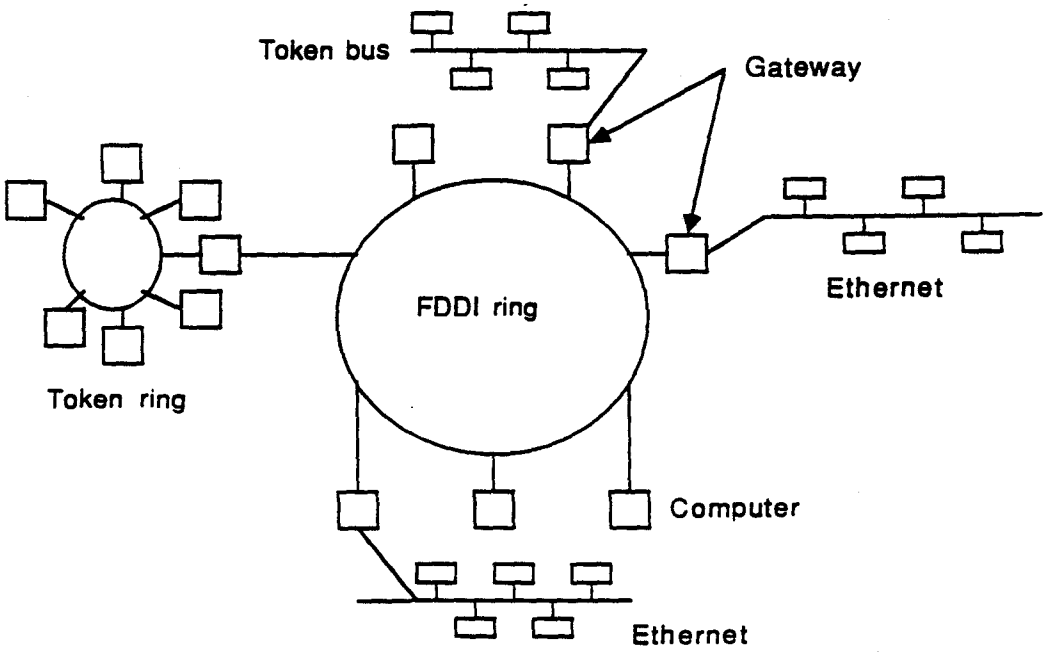


Figure 4-2: FDDI ring used as a backbone network

4.4.1. Current Fermilab FDDI Topology

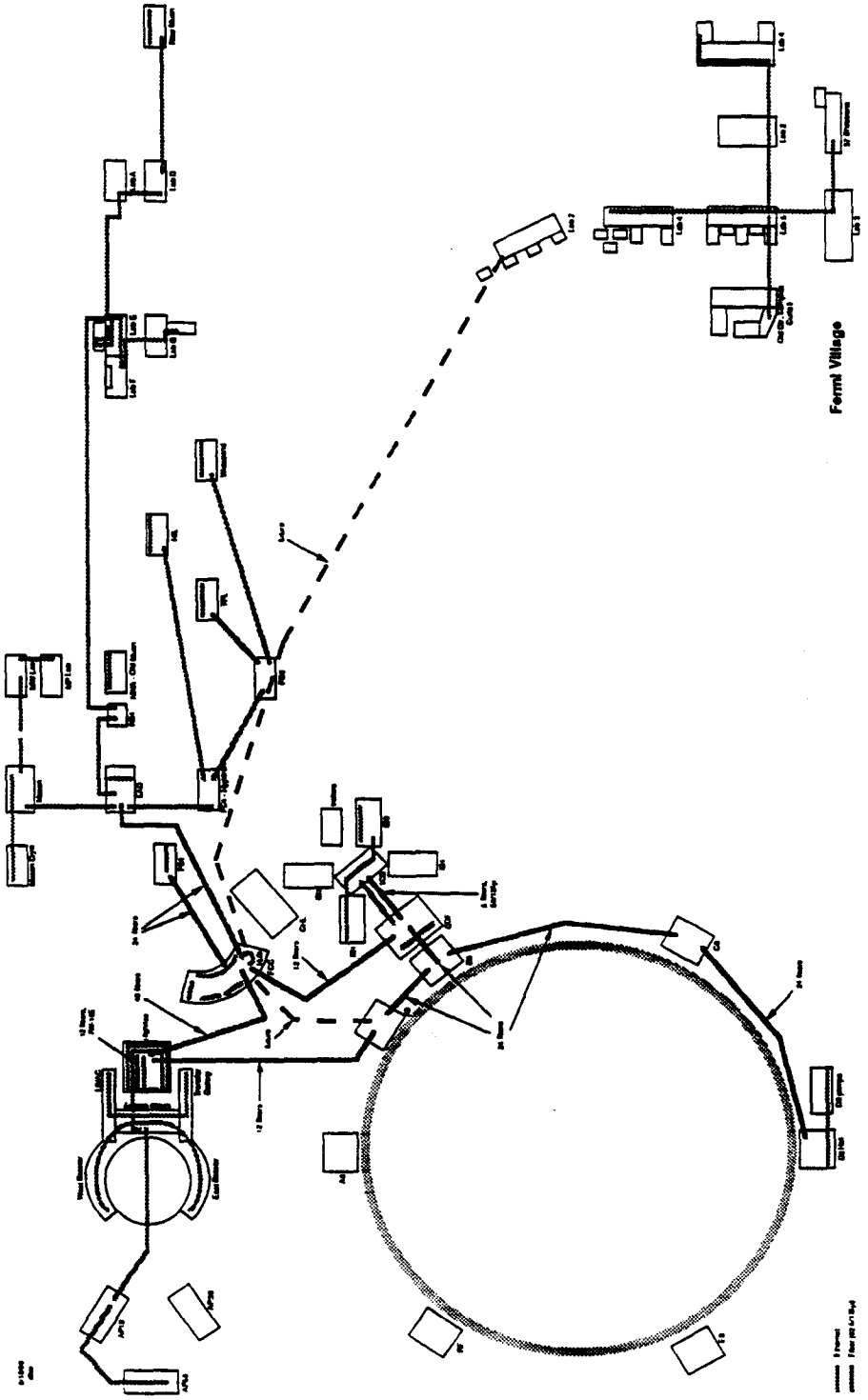


Figure 4-3: FDDI Topology at Fermilab

4.5. FDDI and IEEE 802.5

The FDDI standard is based on the IEEE 802.5 Token Ring standard. The X3T9.5 committee decided to adopt as much of 802.5 as possible, making changes only where necessary to exploit the high speeds of a fiber ring and to provide the services expected on a high-speed local network. This strategy has several advantages:

1. The token protocol is known to work effectively, particularly at high loads; thus there is no need to take the risk of adopting a different approach.
2. The use of similar frame formats facilitates the internetworking of high- and low-speed rings.
3. Understanding of FDDI is facilitated for those already familiar with 802.5.
4. Implementation experience, particularly at the chip level, may be of benefit to vendors of FDDI systems and components.

Table 4-2 summarizes the differences between IEEE 802.5 and FDDI.

FDDI	IEEE 802.5
Optical fiber	Shielded twisted pair
100 Mbps	1 and 4 Mbps
NRZI-4B/5B code	Differential manchester
Explicit reliability specification	No explicit reliability specification
Distributed clocking	Centralized clocking
Timed token rotation	Priority and reservation bits
New token after transmit	New token after receive
Seize token by absorption	Seizes token by flipping bit
FDDI frame structure	802.5 frame structure
4500-octet maximum frame size	No maximum frame size
16- and/or 48-bit addresses	16- or 48-bit addresses
Distributed recovery	Active monitor

Table 4-2: Differences between FDDI and IEEE 802.5

4.6. Application of FDDI Technology

The applications for which FDDI may be used include the following:

- Backbone Networks
- High Speed LANs
- Mainframe I/O Connectivity
- Specialized Military Applications

4.6.1. Backbone Networks

FDDI is expected to become a popular choice for providing the capability to link together various subnetworks within an organization. The section on *FDDI Topology* provides an example of how a FDDI backbone may be configured. Thus, low capacity, low cost local networks may be employed within buildings or departments and may be linked to a higher-capacity, higher-cost local network. This high-capacity network is referred to as a backbone network.

4.6.2. High Speed LANs

The need for a high speed LAN has increased significantly over the last few years. Services and applications have evolved significantly and the network bandwidth is proving inadequate for high performance workstations, graphics and CAD/CAM modeling systems. In addition optical disks are beginning to reach technical maturity and are being developed toward realistic desktop capacities exceeding 1 Gbyte. Also, fax machines, document image processors and graphics applications typically have a resolution of 400 x 400 per page. Even with compression techniques, this will generate a tremendous load. These new demands will require local networks with high speed that can support the larger numbers and greater geographic extent of office systems as compared to computer room systems.

4.6.3. Mainframe I/O Connectivity

FDDI was originally developed for mainframe-to-mainframe and mainframe-to-peripheral connectivity. The backend use of FDDI can be used with FDDI LANs and/or a backbone FDDI network. Backend local networks are used in a computer-center environment to interconnect mainframe computers and mass storage devices. The FDDI network in a backend application is comprised of a preponderance of dual stations with relatively few single-ring stations.

4.8. FDDI Concentrators

Figure 4-7 illustrates the schematic for a FDDI concentrator.

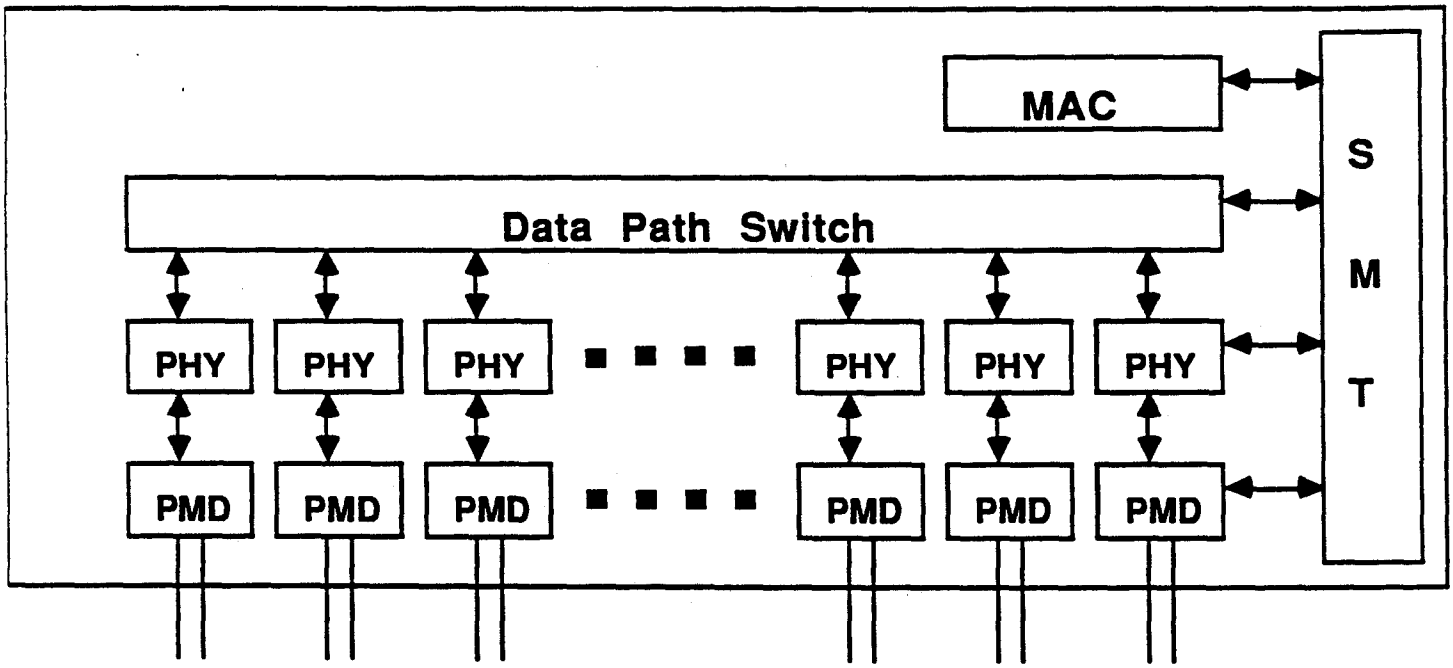


Figure 4-7: FDDI Concentrator

The Dual Attachment Concentrator (DAC) is the "root" concentrator of a "tree" that connects to other "root" concentrators on the dual-ring to form what is known as "Dual Ring of Trees" topology. The Single Attachment Concentrator (SAC) forms the branches within the "tree." Figure 4-8 shows how a FDDI topology may be designed.

4.9. FDDI Frame Format

The standard describes the frame structure in terms of symbols exchanged between Media Access Control (MAC) entities. Each symbol corresponds to four bits. This assignment was chosen because at the physical layer data are transmitted in four-bit groups. MAC entities must deal with individual bits. Figure 4-9 illustrates the frame formats generated by the FDDI protocol.

The overall frame format consists of the following fields:

Preamble — Synchronizes the frame with each station's clock. The originator of the frame uses a field of 16 idle symbols (64 bits); subsequent

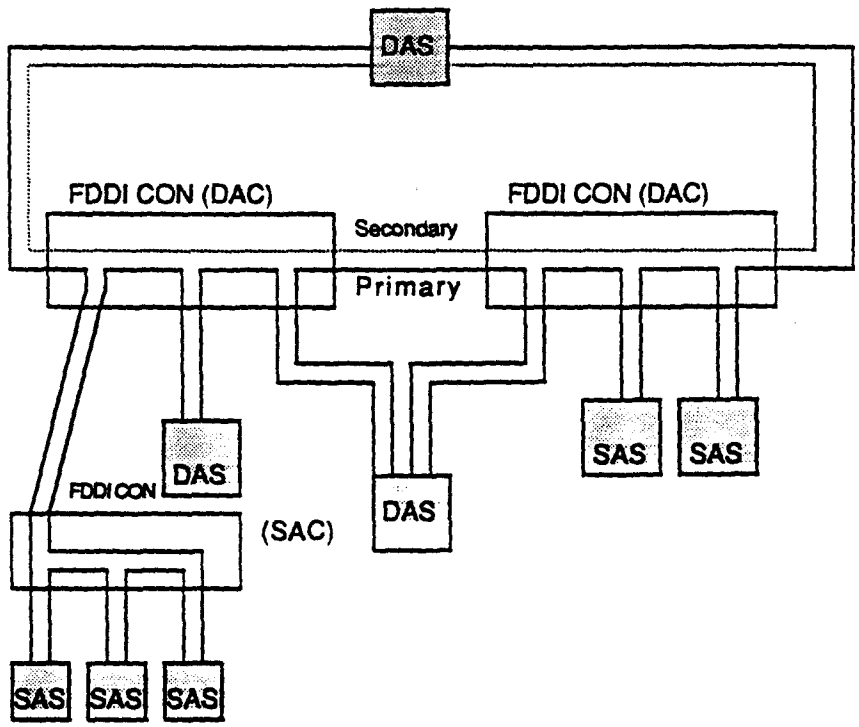


Figure 4-8: FDDI Topology

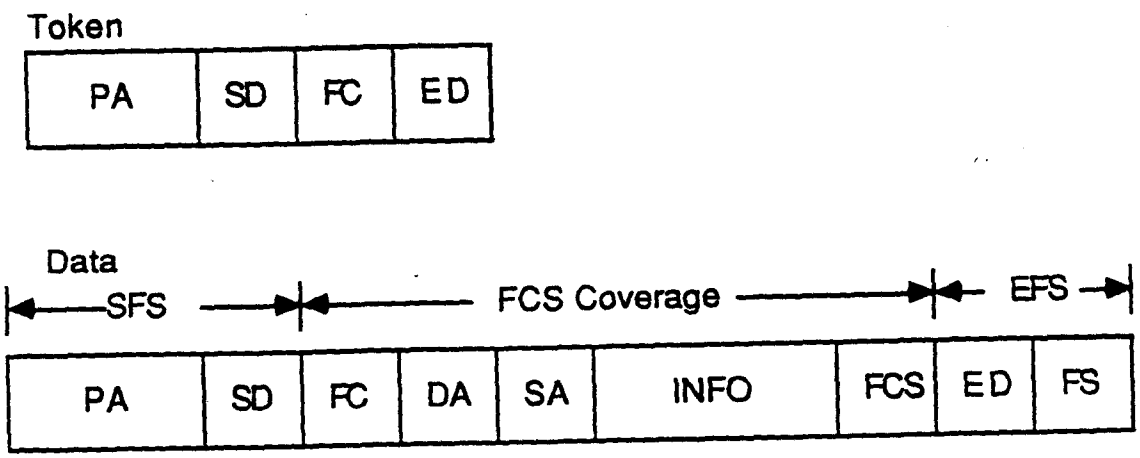


Figure 4-9: FDDI Frame Format

repeating stations may change the length of the field consistent with clocking requirements. The idle symbol is a nondata fill pattern. The actual form of a nondata symbol depends on the signal encoding on the medium.

SD — The Starting Delimiter (SD) indicates the start of the frame. The SD consists of signaling patterns that are always distinguishable from data. It is coded as follows: JK, where J and K are nondata symbols.

FC — The Frame Control (FC) field has the bit format CLFFZZZ, where C indicates whether this is a synchronous or an asynchronous frame; L indicates the use of 16- or 48-bit addresses; FF indicates whether this is a LLC frame or a MAC control frame. In the latter case, the remaining bits indicate the type of MAC frame.

DA — The Destination Address (DA) specifies the station(s) for which the frame is intended. It may be a unique physical address (one station), a multicast-group address (a group of stations), or a broadcast address (all stations on the local network). The ring may contain a mixture of 16-bit and 48-bit addresses.

SA — The Source Address (SA) field specifies the station that sent the frame.

Information — Conatins LLC data or information related to a control operation.

FCS — The Frame Check Sequence (FCS) field is a 32-bit cyclic redundancy check based on FC, DA, SA and Information fields.

ED — The Ending Delimiter (ED) field contains nondata symbols to indicate the end of the frame (except for the FS field). The delimiter is eight bits long for a token (two nondata T symbols) and four bits long (one T symbol) for all other frames. This varies so that frames occupy an integral number of octets.

FS — The Frame Status (FS) field contains the error detected (E), address recognized (A) and frame copied (C) indicators. Each indicator is represented by a symbol, where R represents "off" or "false" and S represents "on" or "true." The FS field may contain additional trailing control indicators whose use is implementor defined. If there is an odd number of additional symbols, the FS field ends with a T symbol.

4.9.1. Address Field Format

Figure 4-10 shows the address field format. The first bit is always zero in the source address field. In the destination address field, it is set to zero to indicate an individual address and to one to indicate a group address. A group address of all 1s is a broadcast address for active stations on the LAN. All other group addresses designate a logical user group defined at configuration time or by a higher-layer convention.

4.6. Application of FDDI Technology

The applications for which FDDI may be used include the following:

- Backbone Networks
- High Speed LANs
- Mainframe I/O Connectivity
- Specialized Military Applications

4.6.1. Backbone Networks

FDDI is expected to become a popular choice for providing the capability to link together various subnetworks within an organization. The section on *FDDI Topology* provides an example of how a FDDI backbone may be configured. Thus, low capacity, low cost local networks may be employed within buildings or departments and may be linked to a higher-capacity, higher-cost local network. This high-capacity network is referred to as a backbone network.

4.6.2. High Speed LANs

The need for a high speed LAN has increased significantly over the last few years. Services and applications have evolved significantly and the network bandwidth is proving inadequate for high performance workstations, graphics and CAD/CAM modeling systems. In addition optical disks are beginning to reach technical maturity and are being developed toward realistic desktop capacities exceeding 1 Gbyte. Also, fax machines, document image processors and graphics applications typically have a resolution of 400 x 400 per page. Even with compression techniques, this will generate a tremendous load. These new demands will require local networks with high speed that can support the larger numbers and greater geographic extent of office systems as compared to computer room systems.

4.6.3. Mainframe I/O Connectivity

FDDI was originally developed for mainframe-to-mainframe and mainframe-to-peripheral connectivity. The backend use of FDDI can be used with FDDI LANs and/or a backbone FDDI network. Backend local networks are used in a computer-center environment to interconnect mainframe computers and mass storage devices. The FDDI network in a backend application is comprised of a preponderance of dual stations with relatively few single-ring stations.

4.6.4. Specialized Military Applications

Characteristics of fiber such as small size, light weight, and EMI immunity make FDDI an attractive choice for military applications. Areas of interest include vehicular and on-board applications for aircrafts, submarines, and ships. FDDI is also being considered for NASA space station applications.

4.7. FDDI Stations

The FDDI network can be constructed with either Single Attachment Stations (SAS) or with Dual Attachment Stations (DAS). Figure 4-4 is a schematic for a SAS station. SAS stations are also referred to as (Class B) stations. SAS stations can be connected only to the primary ring. A failure can isolate a SAS station.

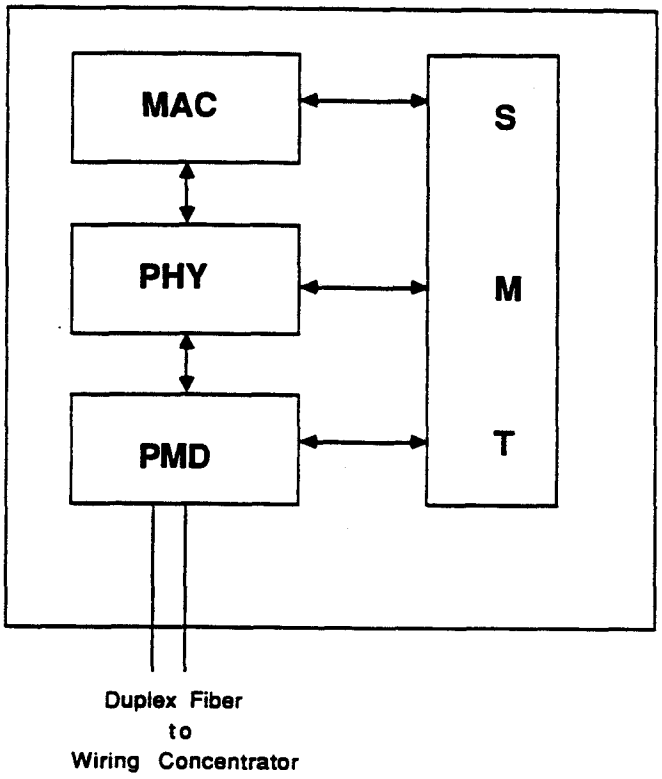


Figure 4-4: Schematic for a Single Attachment Station

DAS stations also referred to as *Class A* can connect to both the primary and secondary rings. In the event of a failure, provisions are made within a DAS station to reconfigure the network using a combination of the operational links of the primary and secondary rings. Figure 4-5 is a schematic for a DAS station.

Figure 4-6 illustrates an FDDI ring architecture with SAS and DAS stations.

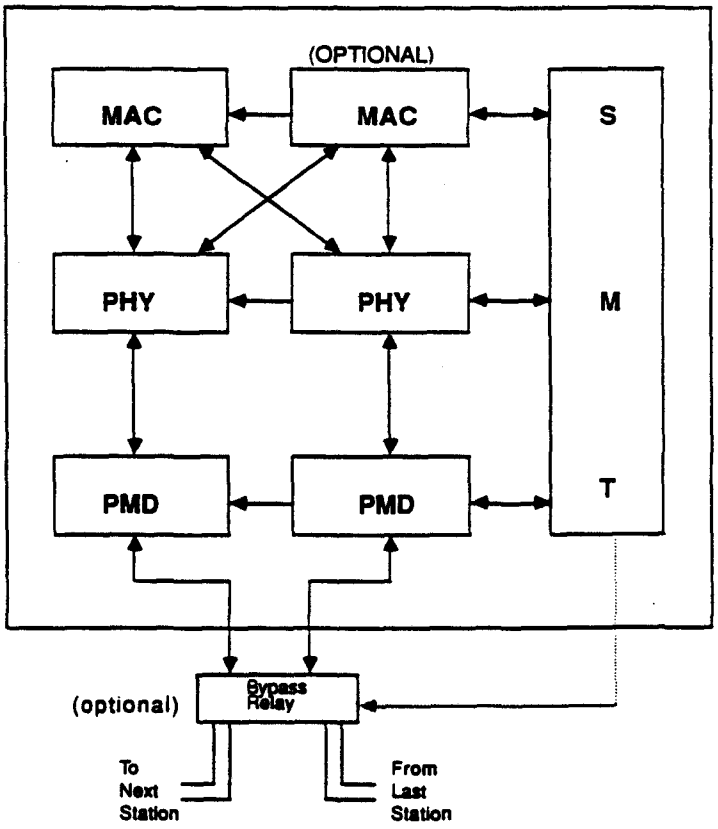


Figure 4-5: Schematic for a Dual Attachment Station

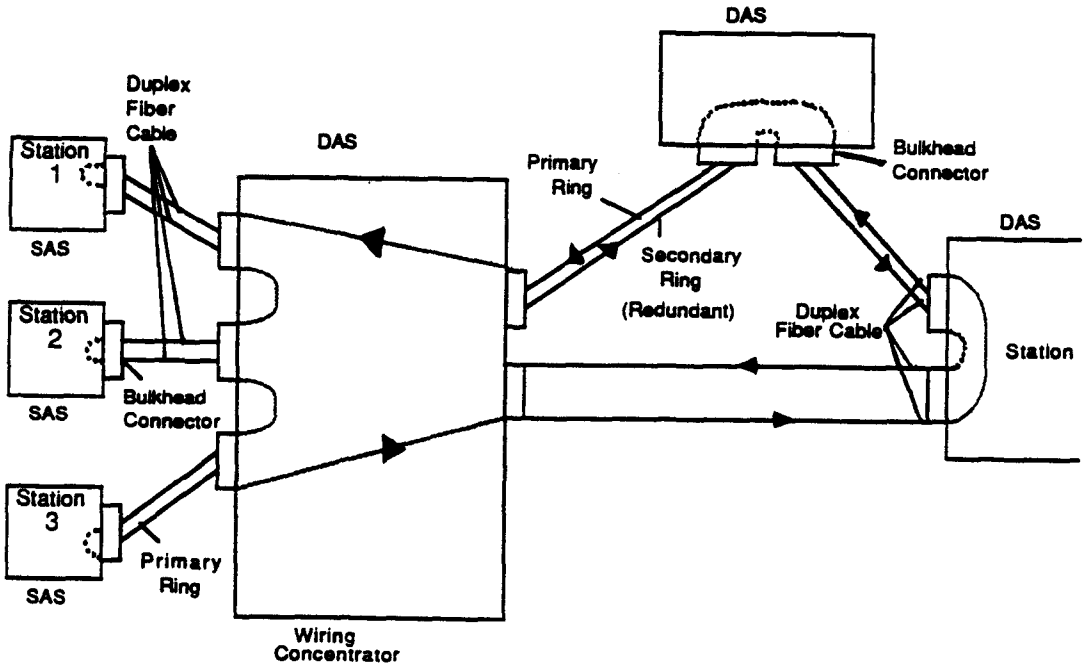


Figure 4-6: FDDI Ring Architecture including SAS and DAS

4.8. FDDI Concentrators

Figure 4-7 illustrates the schematic for a FDDI concentrator.

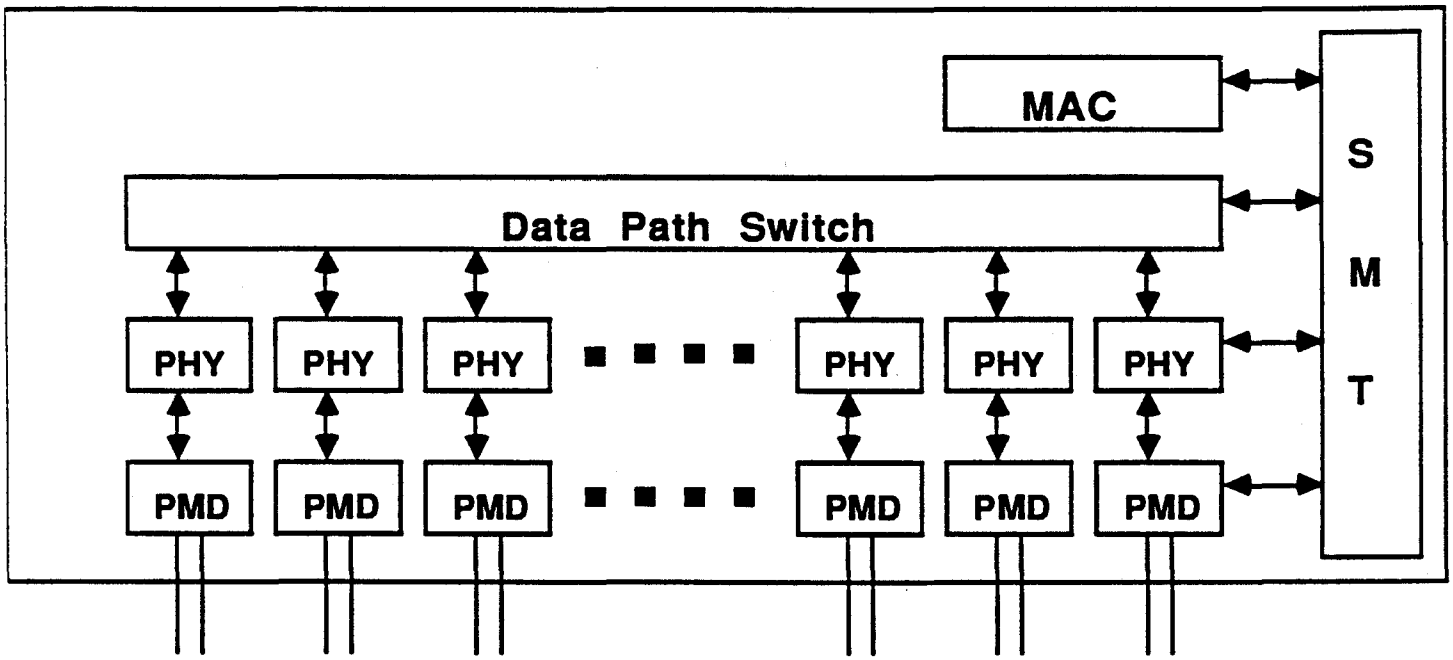


Figure 4-7: FDDI Concentrator

The Dual Attachment Concentrator (DAC) is the "root" concentrator of a "tree" that connects to other "root" concentrators on the dual-ring to form what is known as "Dual Ring of Trees" topology. The Single Attachment Concentrator (SAC) forms the branches within the "tree." Figure 4-8 shows how a FDDI topology may be designed.

4.9. FDDI Frame Format

The standard describes the frame structure in terms of symbols exchanged between Media Access Control (MAC) entities. Each symbol corresponds to four bits. This assignment was chosen because at the physical layer data are transmitted in four-bit groups. MAC entities must deal with individual bits. Figure 4-9 illustrates the frame formats generated by the FDDI protocol.

The overall frame format consists of the following fields:

Preamble — Synchronizes the frame with each station's clock. The originator of the frame uses a field of 16 idle symbols (64 bits); subsequent

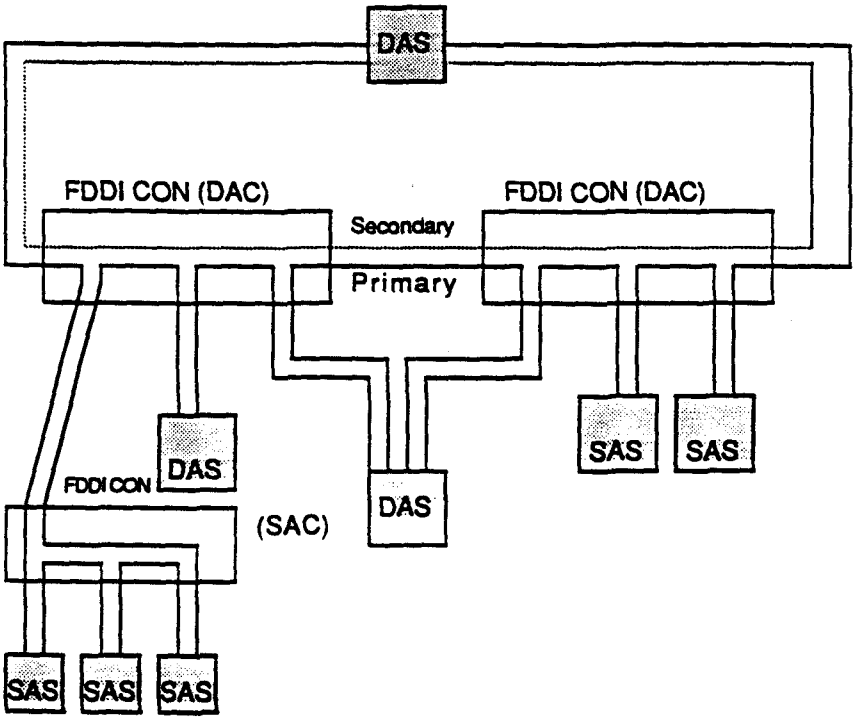


Figure 4-8: FDDI Topology

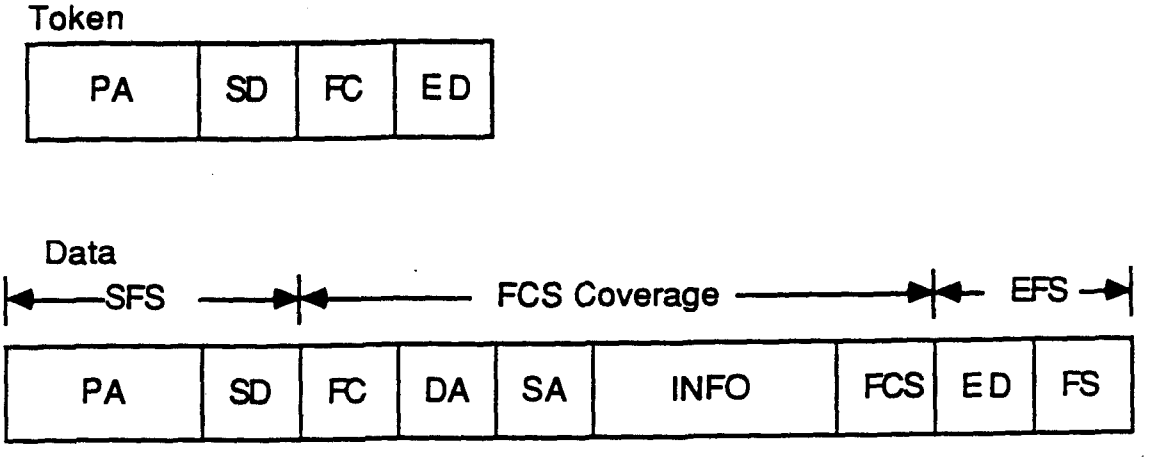


Figure 4-9: FDDI Frame Format

repeating stations may change the length of the field consistent with clocking requirements. The idle symbol is a nondata fill pattern. The actual form of a nondata symbol depends on the signal encoding on the medium.

SD — The Starting Delimiter (SD) indicates the start of the frame. The SD consists of signaling patterns that are always distinguishable from data. It is coded as follows: JK, where J and K are nondata symbols.

FC — The Frame Control (FC) field has the bit format CLFFZZZ, where C indicates whether this is a synchronous or an asynchronous frame; L indicates the use of 16- or 48-bit addresses; FF indicates whether this is a LLC frame or a MAC control frame. In the latter case, the remaining bits indicate the type of MAC frame.

DA — The Destination Address (DA) specifies the station(s) for which the frame is intended. It may be a unique physical address (one station), a multicast-group address (a group of stations), or a broadcast address (all stations on the local network). The ring may contain a mixture of 16-bit and 48-bit addresses.

SA — The Source Address (SA) field specifies the station that sent the frame.

Information — Conatins LLC data or information related to a control operation.

FCS — The Frame Check Sequence (FCS) field is a 32-bit cyclic redundancy check based on FC, DA, SA and Information fields.

ED — The Ending Delimiter (ED) field contains nondata symbols to indicate the end of the frame (except for the FS field). The delimiter is eight bits long for a token (two nondata T symbols) and four bits long (one T symbol) for all other frames. This varies so that frames occupy an integral number of octets.

FS — The Frame Status (FS) field contains the error detected (E), address recognized (A) and frame copied (C) indicators. Each indicator is represented by a symbol, where R represents "off" or "false" and S represents "on" or "true." The FS field may contain additional trailing control indicators whose use is implementor defined. If there is an odd number of additional symbols, the FS field ends with a T symbol.

4.9.1. Address Field Format

Figure 4-10 shows the address field format. The first bit is always zero in the source address field. In the destination address field, it is set to zero to indicate an individual address and to one to indicate a group address. A group address of all 1s is a broadcast address for active stations on the LAN. All other group addresses designate a logical user group defined at configuration time or by a higher-layer convention.

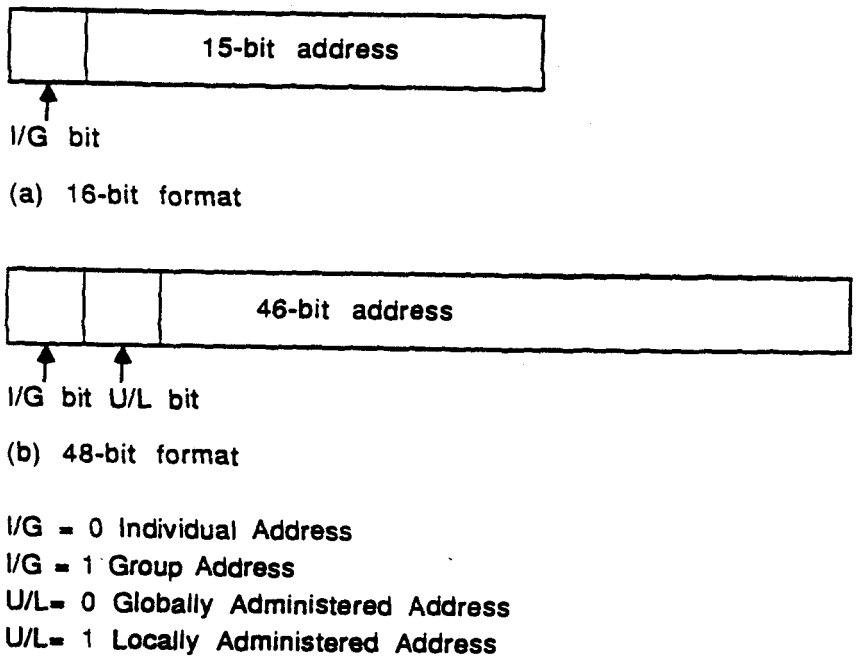


Figure 4-10: FDDI Address Field Format

For 48-bit address fields, the second bit in the source and destination fields is set to zero to indicate a globally administered address and to one to indicate a locally administered address.

The standard specifies that a mixture of 16-bit and 48-bit addresses may be employed on the ring. All stations have the capability to employ 16-bit addresses. A station with only 16-bit address capability can function in a ring with stations concurrently operating with 48-bit addresses Thus, the 16-bit station can:

- Repeat frames with 48-bit addresses
- Recognize the 48-bit broadcast address (all ones)
- React correctly to Claim frames and Beacon frames with 48-bit addresses

Any station detecting the need for initialization of the ring starts the claim token process by issuing *Claim frames*. The *Beacon frame* is used to isolate a serious ring failure such as a break in the ring.

A station using 48-bit addresses has a minimum 16-bit address capability such that the station can:

- Have a fully functional 16-bit individual address
- Recognize the 16-bit broadcast address (all ones)

4.9.2. Frame Control Field

The frame control field indicates which type of frame this is and also contains bits that specify details of operation. There are four types of frames, indicated by the FF bits. These are control, LLC, reserved for implementor and reserved for future standardization.

4.10. Token Ring Operation

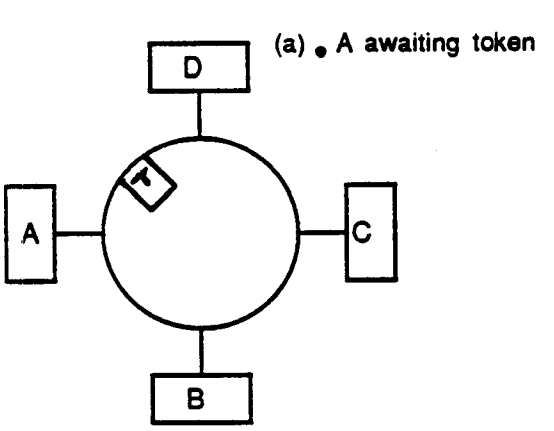
Figure 4-11 provides an example of ring operation. An idle token is initially circulating on the ring. Station A seizes the token and transmits frame F1. It then immediately transmits a new token. F1 is addressed to station C, which copies it as it circulates past. The frame eventually returns to A, which absorbs it. Meanwhile, B seizes the token issued by A and transmits F2 followed by a token. This action could be repeated any number of times, so that at any one time, there may be multiple frames circulating the ring. Each station is responsible for absorbing its own frames based on the source address field.

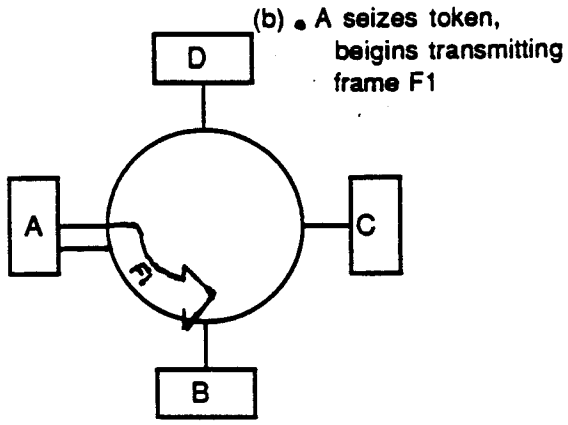
Note that a station releases a new token as soon as it completes frame transmission, even if it has not begun to receive its own transmission. A station that wishes to transmit waits until a token frame goes by, as indicated by a FC field with FF bits set to 00 and ZZZZ bits set to 0000. The station seizes the token by absorbing the remainder of the token from the ring before the entire FC field is repeated. After the captured token is completely received, the station may begin transmitting frames. The station may continue to transmit until it has no more data to transmit or until a token-holding timer (THT) expires.

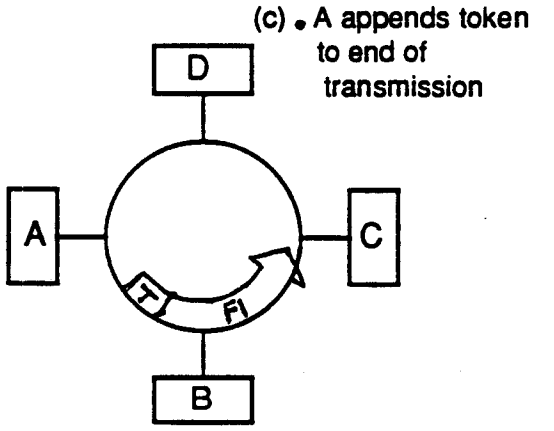
Other stations listen to the ring and repeat passing frames. Each station introduces into the ring approximately a one-bit delay due to the time required to examine, copy, or change a bit as necessary. Each station can check passing bits for errors and set the E indicator if an error is detected. If a station detects its own address, it sets the A indicator; it may also copy the frame, setting the C indicator. This allows the originating station to differentiate three conditions:

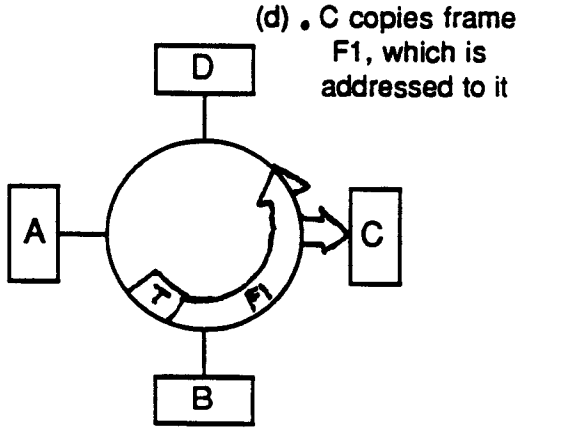
- Station nonexistent/nonactive
- Station exists but frame not copied
- Frame copied

The station that originates the frame is responsible for removing the frame from the ring. The status indicators (E, A, C) in the ending delimiter are examined to determine the result of the transmission. However, if an error is reported, the MAC protocol does not attempt to re-transmit the frame. This is the responsibility of LLC or some higher-layer protocol.


(a) • A awaiting token
D
T
A
C
B


(b) • A seizes token, beigins transmitting frame F1
D
A
F1
C
B


(c) • A appends token to end of transmission
D
A
T
F1
C
B


(d) • C copies frame F1, which is addressed to it
D
A
T
F1
C
B

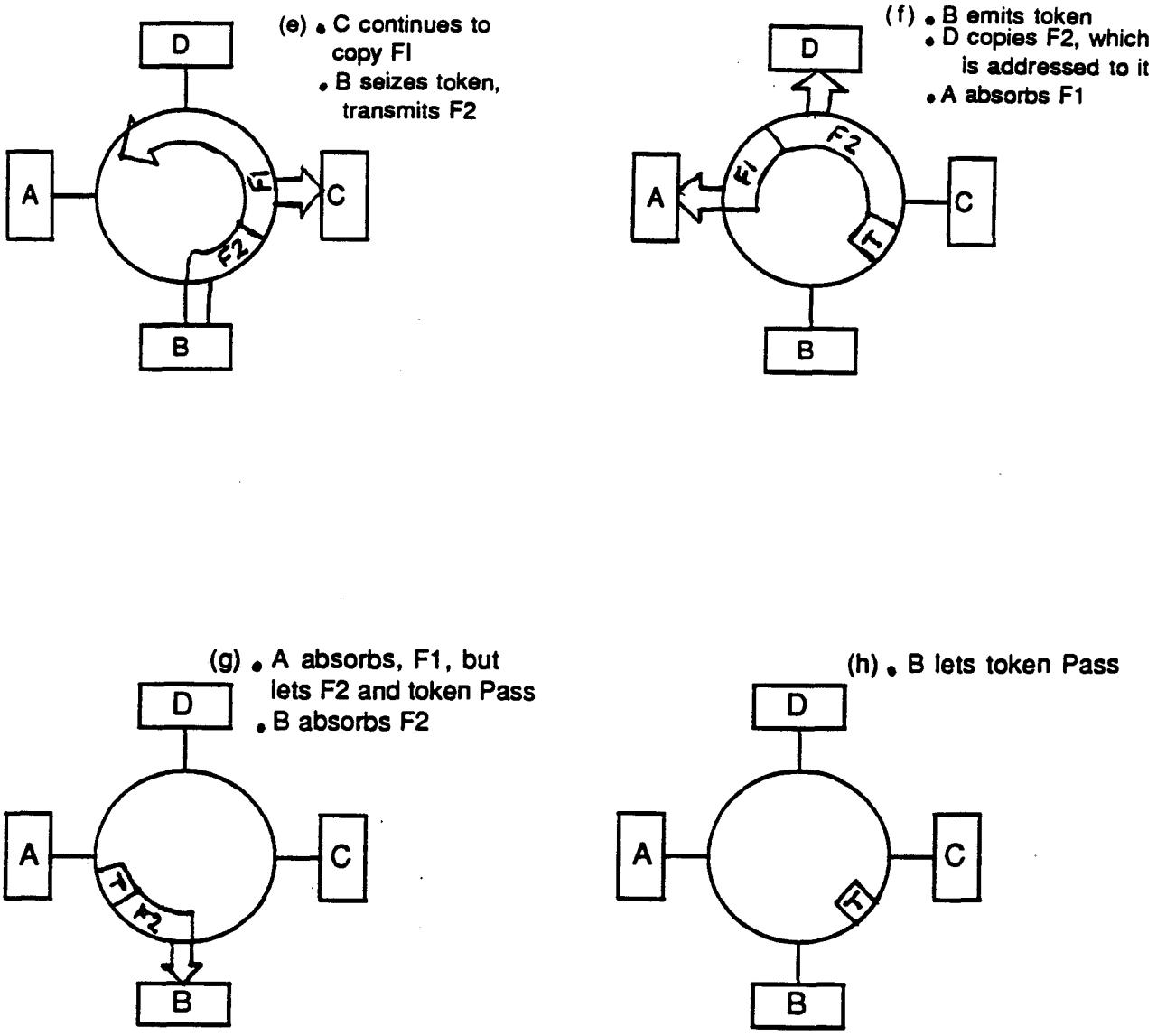


Figure 4-11: FDDI Token and Frame Operation

Figure 4-12 describes the FDDI initialization process.

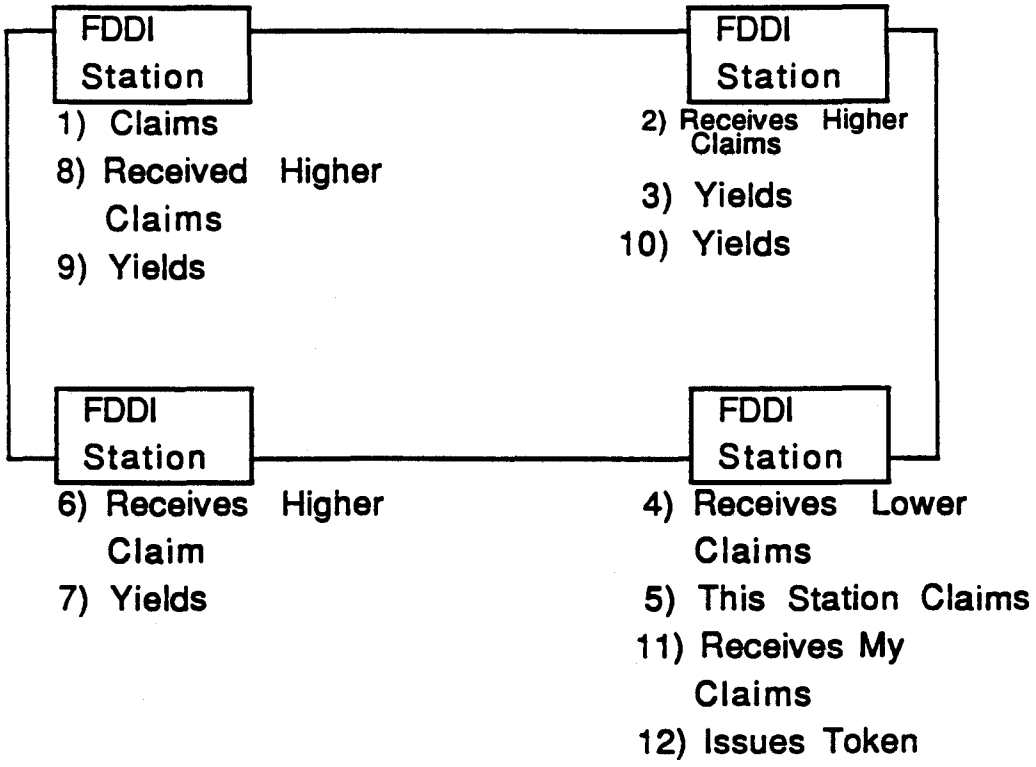


Figure 4-12: FDDI Initialization Process

4.11. FDDI Architecture

The FDDI architecture consists of three components: physical layer, data link layer (only the MAC portion) and station management. Figure 4-13 illustrates the components of the FDDI architecture and a brief summary of their functions.

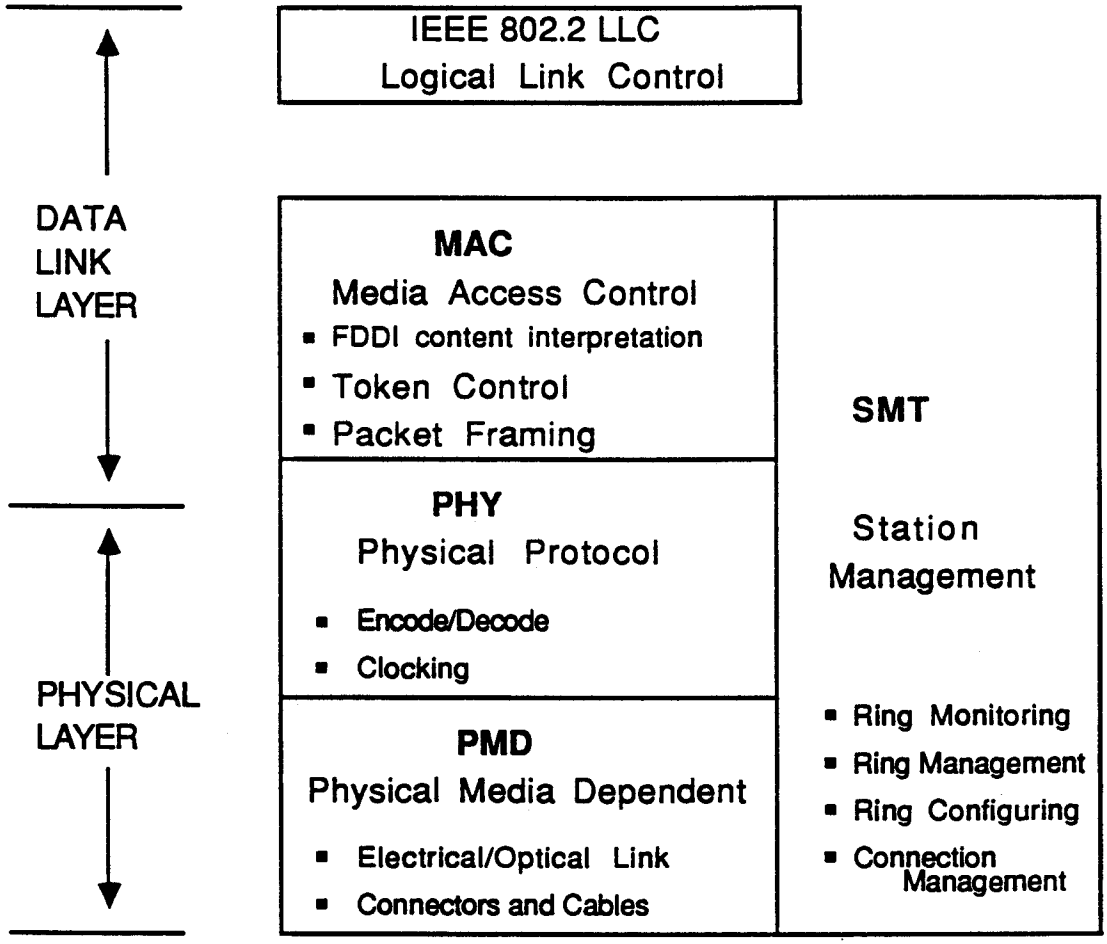


Figure 4-13: FDDI Architecture

4.11.1. Physical Layer

The physical layer defines the medium, connectors, optical bypassing and driver/receiver requirements. It also defines the encode/decode and clock requirements for data transmission to higher layers and the peer entity. The physical layer consists of two sub-layers: the physical layer medium dependent and the physical layer protocol.

Physical Medium Dependent (PMD)

The medium-dependent portion of the physical layer specification defines the physical medium and specifies some reliability features. The FDDI standard specifies an optical fiber ring with a data rate of 100 Mbps, using the Non-Return-to-Zero Inverted (NRZI)-4B/5B encoding scheme. The 4B/5B encoding scheme is described in the next section on *Physical Layer Protocol.* The wavelength specified for data transmission is 1300 nm. Virtually all fiber transmitters operate at 850, 1300, or 1550 nm. The cost and performance of the system increases as the wavelength increases.

Long-distance networks rely primarily on single-mode fiber. Single-mode fiber requires the use of lasers as a light source; light emitting diodes (LEDs) are used for multimode fibers. The dimensions of the multimode optical fiber cable are specified in terms of the diameter of the core of the fiber and the outer diameter of the cladding layer that surrounds the core. The combinations specified in the standard are 62.5/125 and 85/125 um. The standard lists as alternatives 50/125 and 100/140 um. In general, smaller diameters offer higher potential bandwidths but also higher connector loss.

The FDDI standard includes three techniques for increasing reliability.

- *Station bypass*: A bad or powered-off station may be bypassed using an automatic optical bypass switch.
- *Wiring Concentrator*: Wiring concentrators can be used in a star wiring strategy.
- *Dual rings*: Two rings are employed to interconnect the stations in such a way that a failure of any station or link results in the reconfiguration of the network to maintain connectivity.

Physical Layer Protocol

The medium-independent portion of the physical layer specification is referred to as the physical layer protocol (PHY). In addition to defining the physical layer services, the physical layer protocol addresses data encoding and *jitter.* Jitter is defined as the deviation of clock recovery that can occur when the receiver attempts to recover clocking as well as data from the received signal.

The FDDI PHY implements full-duplex physical connections. A physical connection consists of a pair of physical links each on a separate counter-rotating FDDI ring. Each connection exists independent of other connections in the network. This provides for automatic transmission system (PMD) fault isolation and permits concurrent initialization of all working connections.

Digital data needs to be encoded in some form for transmission as a signal. The type of encoding depends on the nature of the transmission medium, the data rate and other constraints such as cost. Optical fiber is inherently an analog medium; signals can only be transmitted in the optical frequency range. The FDDI standard specifies the use of a code referred to as 4B/5B. This code

represents each data quartet (4 bits) as a unique symbol. Each four bits of data are encoded into a symbol with five cells such that each cell contains a single signal element (presence or absence of light). In effect, each set of four bits is encoded as five bits. The efficiency is thus raised to 80 percent; 100 Mbps is achieved with 125 Mbaud.

Each element of the 4B/5B stream is treated as a binary value and encoded using a technique referred to as Nonreturn to Zero Inverted (NRZI). In this code a binary 1 is represented with a transition at the beginning of the bit interval and a binary 0 is represented with no transition at the beginning of the bit interval. A benefit of this scheme is that it is generally more reliable to detect a transition in the presence of noise and distortion than to compare a value to a threshold. This aids the ultimate decoding of the signal after it has been converted back from optical to the electrical realm.

The FDDI standard specifies the use of a distributed clocking scheme: each station uses its own autonomous clock source to transmit or repeat information onto the ring. Each station has its own elastic buffer of at least 10 bits. Data is clocked into the buffer at the clock rate recovered from the incoming stream, but is clocked out of the buffer at the station's autonomous clock rate. This distributed system is believed to be more robust, and to minimize jitter. As a consequence of reclocking at each station, jitter does not limit the number of repeaters in the ring. However, the size of the elastic buffer limits frame size to a maximum of 4500 octets.

4.11.2. Media Access Control Layer

The FDDI Medium Access Control (MAC) layer provides a data service similar to IEEE 802.5 token ring. The types of packet data services that are defined are: Station Management (SMT), Logical Link Control (LLC) and implementor. Each MAC provides SMT data service, and MACs in peer and slave stations normally provide all data services. Data packets are conveyed in frames, and each frame is explicitly labeled with its packet type.

The data link layer of the OSI/RM consists of two sublayers: MAC and LLC. The FDDI LLC frame format is the same as the IEEE 802.2 LLC.

The FDDI MAC uses a timed token protocol for ring scheduling. This protocol provides three classes of service: synchronous, restricted asynchronous and priority asynchronous. Synchronous service guarantees both throughput and response-time limits to one or more concurrent users. Synchronous bandwidth is allocated in advance using the SMT protocol and is dynamically assigned by the MAC protocol. Unused synchronous bandwidth is allocated to the asynchronous services on the ring. Restricted asynchronous service guarantees both throughput and response-time limits for one extended dialog (e.g. bursty-mode transmission) at a time. Restricted bandwidth is dynamically allocated and assigned by the MAC protocol and any unused restricted bandwidth is wasted.

Priority asynchronous service is FDDI's normal mode of access. It supports multiple, independently programmable priorities (with arbitrary granularity) and provides fair distribution of throughput and response time among users at each level. Priority bandwidth is dynamically allocated and assigned by the MAC protocol. Unused bandwidth at one priority level is available to lower-priority users. This service operates more efficiently as the offered load increases, although response time deteriorates.

The FDDI MAC is a published ANSI standard and is also an approved ISO standard.

4.11.3. Station Management

The Station Management (SMT) standard was still unfinished at the time this chapter was written. This standard provides services such as control of station initialization, configuration management, fault isolation and recovery, and scheduling procedures. The SMT standard is currently working on the following: duplicate-address detection after a logical ring has been established and a SET frame specification for a SMT packet protocol to allow control of any station. Another area of work is synchronous bandwidth allocation.

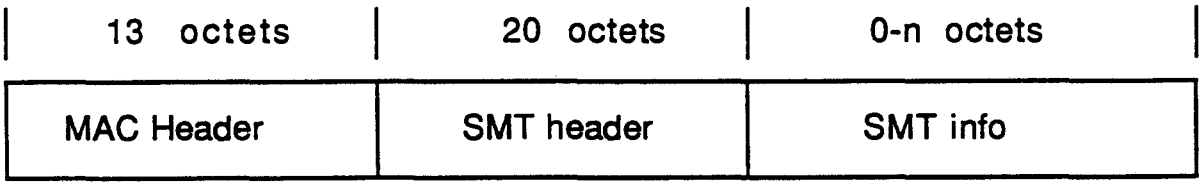
The SMT protocol has a packet dialog for uniquely determining the relative position of all nodes using Neighbor Information Frames, thus supporting the construction of ring maps. It also specifies Station Information Frames that can be used to acquire information about each node's internal configurations, packet throughput and important reports of media or hardware errors. Figure 4-14 describes the SMT frames:

NIF	The Neighbor Information Frame (NIF) is used by a station for periodic announcement of its address and basic station description. It is of three types: Announcement, Request and Response. NIF is transmitted every 30 seconds. NIF can also be used by the monitoring station to build a logical ring map.
SIF	The Status Information Frame (SIF) is used to request and provide (response) station configuration and operating information. There are two possible classes: SIF Configuration and SIF Operation.
ECF	The Echo Frame may be used to confirm that a station's PHY, MAC and SMT are at least partly operational.

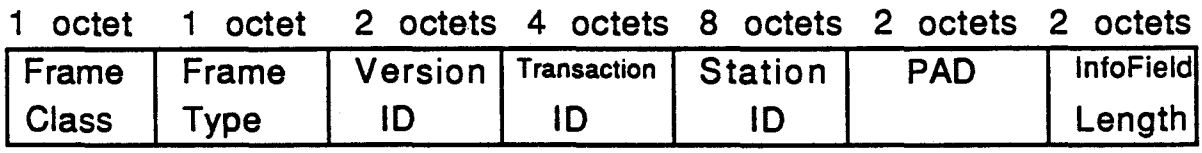
Section(Industry Trends) Computer systems available in the market today from vendors such as DIGITAL, Sun Microsystems, Data General, IBM, Silicon Graphics and Ardent have the processing power that was found only in computer rooms just a year or so ago. These systems need the increased performance possible with FDDI; in fact, for many of the applications being proposed for such systems, FDDI bandwidth is insufficient.

Frame Formats and Protocols

SMT Frame Format



SMT Header



SMT InfoField

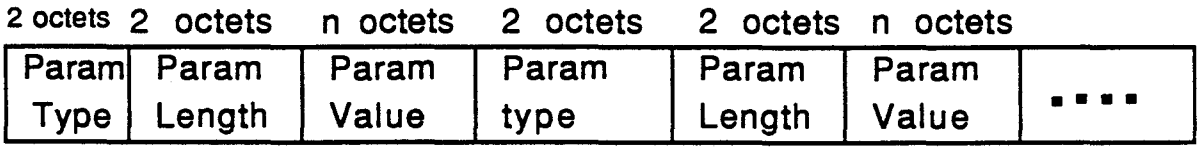


Figure 4-14: SMT Frame Formats

However, for the next few years only the top end of the market will be able to capitalize on the opportunities that result from advances in network technology. The primary reason for this is cost. Initial FDDI implementations for VME and Multibus systems will very likely cost between $7500 and $10,000, given the cost of VLSI, fiber-optic transceivers (also referred to as FOX) and other components.

The types of FDDI products that have been developed include:

- Bridges (usually to Ethernet)
- Wiring concentrators
- Routers to other LANs (including FDDI)
- Dedicated sensors/controllers connected with fiber (military/industrial applications)

- Connections to general-purpose computer/workstations

For example, Sun Microsystems' FDDI product is the FDDI/DX dual-attach VMEbus adapter. Current FDDI network adapters must use several chips that are serviced by a dedicated coresident microprocessor, commonly referred to as a *node processor.* An architecture having the VMEbus as its backplane interconnection and UNIX as the host operating system is a common server configuration providing more than 10 M bytes per second throughput for well-designed direct memory access (DMA) VMEbus interfaces. For example, the SPARCserver 490 can achieve DMA rates approaching 22 M bytes per second over the VMEbus. Sun's FDDI/DX hardware includes two FDDI media interface connectors. The node processor for the FDDI/DX adapter is Motorola's 68020 processor. The node processor accesses a set of SuperNet FDDI chips from Advanced Micro Devices.

The operating system, memory speed, bus architecture and a host of other system related design limitations will limit the applicability of FDDI. It is thus unlikely that FDDI will find acceptance in PC-class machines for some time. The raw bandwidth available with FDDI will have a significant effect on the design of the next generation of networked hardware.

4.12. References and Further Reading

1. Cooper, S. "Joining the Next LAN Generation". *UNIX Review 7*, 2 (February 1989), 48-59.

2. Jain, R. *Performance Analysis of FDDI Token Ring Networks: Effects of Parameters and Guidelines for Setting TTRT.* DIGITAL, 1989. DEC-TR-655.

3. Mezrich, B. "FDDI Hits the Road with Fast Throughput, High Availability". *Sun Technical Journal 3*, 2 (May/June 1990), 42-60.

4. Stallings, W.. *Handbook of Computer-Communications Standards.* Macmillan, New York, 1987.

5. Tanenbaum, A.S.. *Computer Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1988.

This Page Intentionally Left Blank

Part III

ARCHITECTURES, PROTOCOLS, AND GATEWAYS

This Page Intentionally Left Blank

Chapter 5

Understanding TCP/IP

ARPANET, MILNET, MINET, NSFNET, CSNET, SATNET, WIDEBAND and other linked networks that use Transmission Control Protocol/Internet Protocol (TCP/IP) form what is known as the ARPANET Internet or just the Internet network. SATNET and WIDEBAND are satellite networks that are connected to ARPANET. The National Science Foundation, the Department of Energy and the National Aeronautics and Space Administration all participate, using TCP/IP to connect many of their research sites with those of Defense Advanced Research Projects Agency (DARPA). TCP/IP is a non-proprietory standard and is supported by IBM, APPLE, DEC, HP, NCR, AT&T, Amdahl, Cray and others. The Internet network consists of thousands of hosts and over 100,000 users. It is interesting to note that the Internet has the potential to support over 4 billion nodes. The services offered include file transfer, electronic mail and remote login. The services are supported by protocols such as FTP, SMTP and TELNET. Since 1987, digital voice transmission and encryption are in use on the ARPANET.

Software packages such as SRI's MultiNet and DIGITAL's VMS/ULTRIX Connection enable VMS systems to support the Internet networking protocols, TCP and IP. Thus, the layered software products provide a bridge between VMS and UNIX systems.

5.1. Internet Communication Architecture

The TCP/IP Internet protocols are organized into four conceptual layers. Figure 5-1 provides comparison between the OSI/Reference Model and the Internet Communication Architecture.

Figure 5-2 details the layering model and the type of data passed between the layers.

The following is a brief description of each layer.

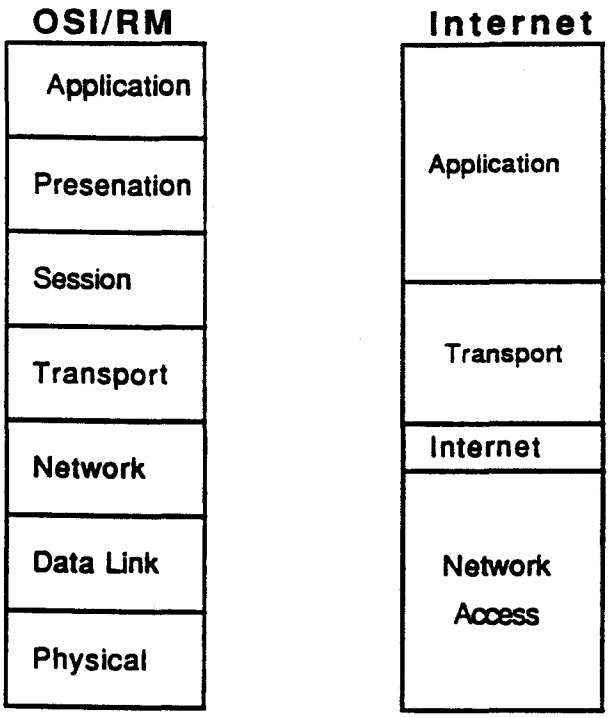


Figure 5-1: OSI/Reference Model and the Internet Architecture

1. **Network Access Layer:** This layer corresponds to the *Physical and Data Link Layer* in the OSI/Reference Model. The network access layer accepts IP datagrams and transmits them over the network to which the system is attached. For Ethernet based LANs, data sent over the media are Ethernet frames, each of which is 1518 bytes. At the network access layer, systems may be interfaced to packet switched networks such as X.25 and local networks such as IEEE 802.3.

2. **Internet Layer:** This layer handles machine to machine communication. The TCP packet received from the transport layer is encapsulated in an IP datagram. Based on the destination host information, the Internet layer uses a routing algorithm to determine whether to deliver the datagram directly, or send it to a gateway. The datagram is then passed to the network access layer for transmission. For incoming datagrams the Internet layer checks their validity, deletes

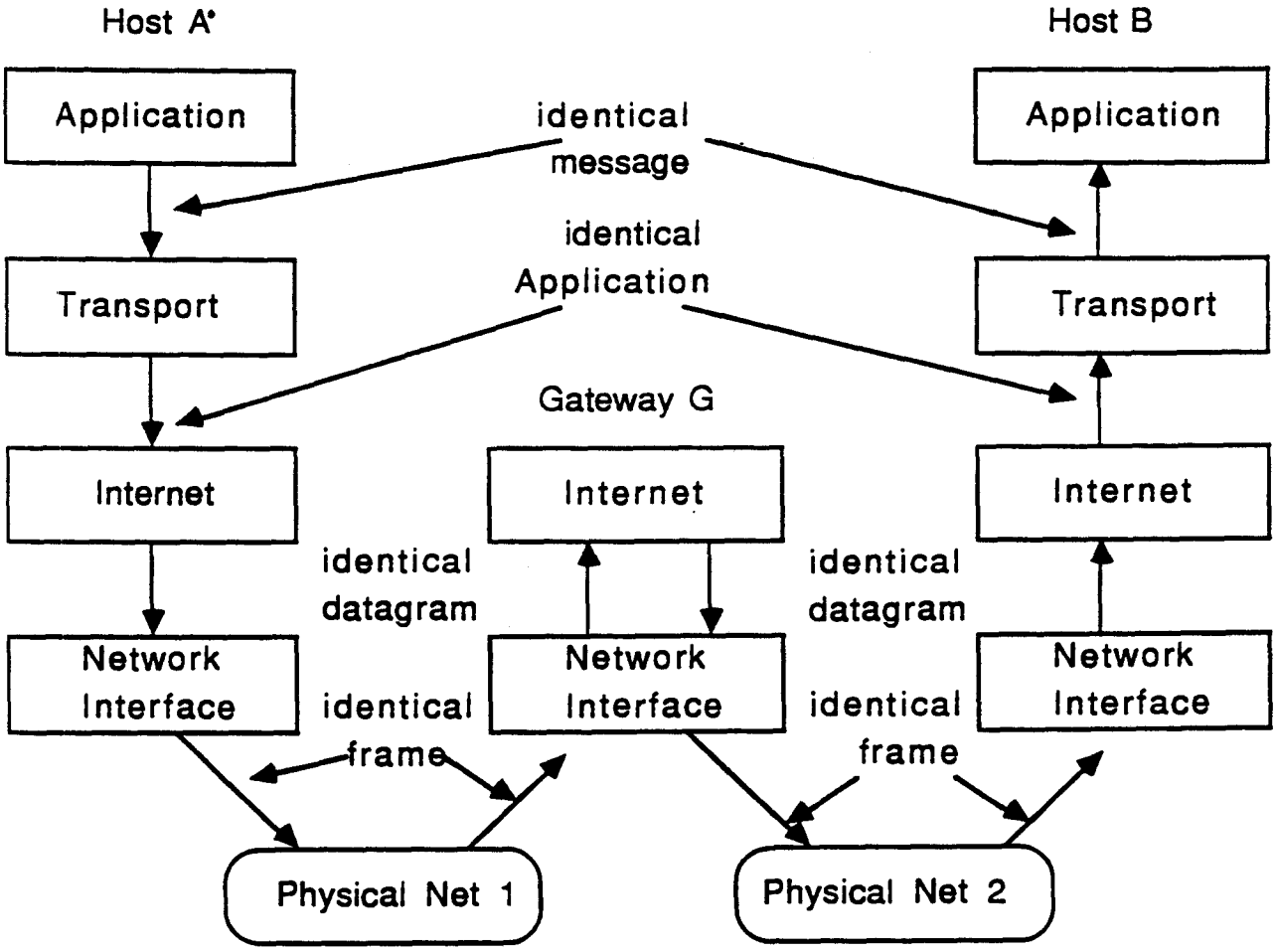


Figure 5-2: Internet Communication Architecture

header information and uses routing algorithm to determine if the datagram is to be processed on the local system or forwarded. If the datagram is for the local system, the Internet layer software chooses from among several transport protocols the one that will handle the packet. The Internet layer sends and receives all Internet Control Message Protocol (ICMP) messages. In general, ICMP provides feedback about problems in the communication environment.

3. **Transport Layer:** The transport software breaks the data received from the higher layers into small pieces, called packets. Each packet is passed to the Internet layer. The transport layer may regulate flow of information. It also provides a reliable transport mechanism, thereby, ensuring that data arrives without error and in sequence. This layer is also known as the *host-to-host layer.*

4. **Application Layer:** Users may invoke application programs such as TELNET, FTP or SMTP in order to access the Internet. The application interacts with the transport level protocol to send or receive data. The data, which may be a sequence of messages or a stream of bytes, is passed to the transport layer for delivery. This layer is also referred to as the *process layer.*

Figure 5-3 provides information on objects passed between Internet architecture layers.

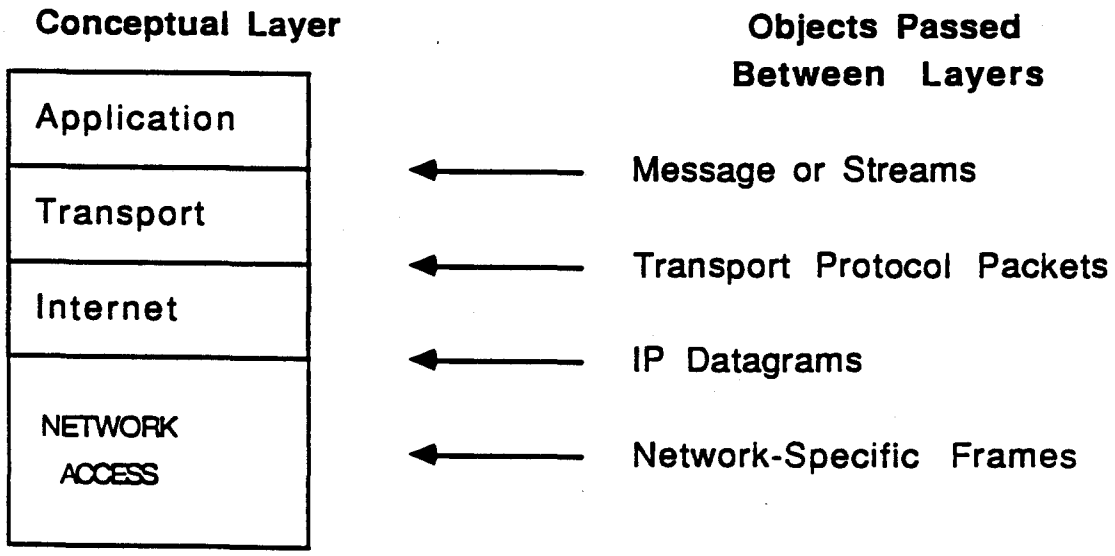


Figure 5-3: Data Passed between Internet Architecture layers

5.2. Internet Addressing Scheme

In order for a host to communicate with a remote host over Internet, it must know the remote host's Internet address. Each host has its own 32-bit Internet address that uniquely identifies it from any other host on the Internet. Conceptually, each address is a pair *(netid, hostid)*, where *netid* refers to network portion of the address, and *hostid* refers to the host portion of the address. Thus, the Internet address is divided into two parts, the network portion and the host portion. A common notation for specifying Internet addresses is four fields separated by periods. Each field ranges from 0 to 255 decimal.

`field1.field2.field3.field4`

5.2.1. Internet Address Classes

The network part of the Internet address specifies the network class and the network address. The characteristics of each class are detailed below:

1. Class A addresses use 7 bits for netid and 24 bits for hostid; thereby providing the potential for 128 networks with up to 16 million hosts each. The first field specifies the network number and class. The first field can be from 1 to 126. The first bit of the first field for a Class A network is always 0.

2. Class B addresses use 14 bits for netid and 16 bits for hostid; thereby providing the potential for 16,384 networks with up to 64K hosts each. The first two fields specify the network number and class. The first field can be from 128 to 191. The second field can be from 1 to 254. The first 2 bits of the first field for a Class B network are always 10.

3. Class C addresses use 22 bits for netid and 8 bits for hostid; thereby providing the potential for $2*10^6$ networks with up to 256 hosts each. The first three fields specify the network number and class. The first field can be from 192 to 223, the second field from 0 to 255, and the third field from 1 to 254. The first 3 bits of a Class C network are always 110.

4. Class D enables use of Multicast packets, wherein, a datagram is targeted to a group of hosts. The first 3 bits of a Class D network are always 111.

Figure 5-4 illustrates the different Internet address formats.

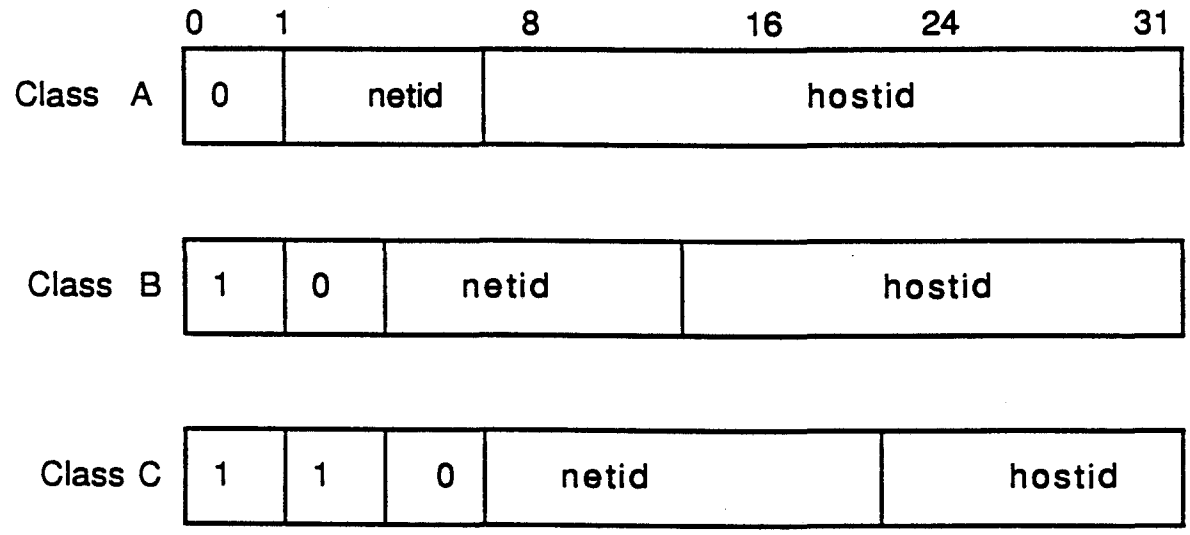


Figure 5-4: IP Source and Destination Address Formats

5.2.2. Loopback Address

Note that when the first field is 127, it is used as the loopback address for the local host. For UNIX systems this is defined in the **/etc/networks** and **/etc/hosts** files. Also, if you have an Internet address where the middle two fields are zero on a Class A network, you can obtain an alternate Internet address notation by dropping the two middle fields. Thus, Internet address 92.0.0.1 can be expressed as 92.1.

5.2.3. Internet Address Notation

The 32-bit Internet address is typically written in dotted decimal notation. The following is an example of the address notation.

```
32-bit address:  1000 0011  1110 0001  0000 1000  1100 1000
Dotted decimal:  131.225.8.200
Symbolic form :  CISC1.FNAL.GOV
```

131.225.8.200 or CISC.FNAL.GOV is the address assigned to the Cisco gateway at Fermilab. This gateway is Fermilab's interface to the rest of the Internet network. Thus, 131.225.8.200 is defined as Fermilab's default route. Note that this is a Class B address since the first field is between 128 and 192. Also, the ad-

dress contains four fields wherein fields one and two (131.225.) refer to the network portion of the address (netid) and fields three and four (8.200) refer to the host portion of the address (hostid).

5.2.4. Assignment of Internet Addresses

The network portion of all Internet addreses are assigned by *Network Information Center* (NIC) located at SRI International. Typically, large networks such as the ARPANET are assigned Class A addresses, intermediate networks are assigned Class B addresses, while sites with numerous networks and few hosts (< 254) for each network are assigned Class C addresses.

Fermilab is a Class B Internet network. The network portion of Fermilab's Internet address is 131.225. The Data Communications Group at Fermilab assigns local Internet addresses. The chapter *Using TCP/IP* provides information on how system managers at Fermilab may obtain Internet addresses for their local system.

5.2.5. Broadcast Addresses

An Internet broadcast address is the address used to send messages to all hosts on the network. The default format of the broadcast address consists of the network portion (netid) followed by all ones (1). Since Fermilab is a Class B network the network portion of the address is 131.255. The broadcast address for hosts at Fermilab is:

131.225.255.255

Note that the host portion in the broadcast address is 255.255 (all ones).

5.2.6. Subnetworks

Subnetworks are a useful way of organizing hosts within a network into logical groups. *Subnet routing* allows numerous subnetworks to exist within a given network. If subnet routing is utilized then the bits in the host field (lower order 16 bits for a Class B network) are divided into two groups: subnetwork and host. Thus, the Internet address for a Class B network consists of the following three fields: network, subnetwork and host.

Since the system does not know which bits in the host field are to be interpreted as the subnetwork part of the Internet address, a *subnet mask* is required. The subnet mask is what informs the system which bits of the Internet address are to be interpreted as the network, subnetwork and host address. A subnet mask is a 32-bit number with a one-to-one correspondence between each of the 32 bits in the subnet mask and each of the 32 bits in the Internet address.

Since each host must reside on a network, the first field of the subnet mask must be turned on. For each bit in the subnet mask that is a binary 1 the corresponding bit position of the Internet address is interpreted as part of the network and subnetwork address. It is a *netmask* entry that defines the subnet mask. Thus, a netmask entry of 255.255.255.255 is incorrect since it leaves no bits to be interpreted as the host address. On the other hand, a netmask entry of 0.0.0.0 is incorrect since it leaves no bits to be interpreted as the network address.

In general, the entire 8-bit field is turned on (255) or off (0). The first field of the subnet mask is always 255, so the system can interpret the network number. The fourth field is typically 0, so the system can interpret the host address. The most frequently used subnet masks are:

```
255.255.255.0
255.255.0.0
255.0.0.0
```

Since Fermilab is a Class B network, we have the option of either implementing subnetting or not doing so. If subnetting is not implemented then the significance of the address fields are as follows:

```
field1    network
field2    network
field3    host
field4    host
```

If subnetting were to be implemented at Fermilab then the address fields would imply:

```
field1    network
field2    network
field3    subnetwork
field4    host
```

At Fermilab subnetworks are not being implemented. Although, on a logical basis, assignment of Internet addresses is done as though subnets existed. For example, Department A with 200 hosts could be assigned an Internet address with the third field 25, while Department B could be assigned an Internet address with the third field set to 30. The motivation for assigning addresses as though subnets did exist are to ease the process of migrating towards subnet networks at Fermilab.

5.3. Network Byte Order

Some machines store 32-bit integers such that the lowest memory address holds the least significant byte of the integer. This is known as *Little Endian.* Other machines store 32-bit integers such that the lowest memory address holds the most significant byte of the integer. This is known as *Big Endian.*

The Internet has defined a *network standard byte order* that all machines must use for binary fields in Internet packets (obviously, the user data field is not subject to the standard). Each host converts binary fields from the local representation to network standard byte order before sending the packet. Conversely, when a packet is received, it is converted from network standard byte order to the host specific byte order.

The Internet standard specifies the Big Endian style of sending data; thus, the most significant byte is sent first.

5.4. Transmission Control Protocol (TCP)

Since 1983 computer systems connected to the ARPANET have been using the Transmission Control Protocol/Internet Protocol (TCP/IP). DARPA funded Bolt Beranek and Newman, Inc. (BBN) to implement the Internet protocols under UNIX. The University of California's Berkeley Software Distribution (BSD) integrated the network protocols with its distribution.

Transmission Control Protocol (TCP) is a transport layer protocol and is responsible for providing a reliable mechanism for the exchange of data between processes in different systems. The transport layer is the first *end-to-end* layer or in other words a source-to-destination layer. Thus, a program on the source system communicates with another program on the destination system. The lower layers, such as network and below, communicate between a system and its immediate neighbors and not directly between the source and destination systems.

TCP accepts arbitrarily long messages from higher layers and breaks them into pieces not exceeding 64K bytes. Each piece is sent as a separate datagram to the network layer. Since the network layer does not guarantee packet delivery, it is up to TCP to provide reliability. Packets may also be delivered by the network layer out of sequence; TCP is also responsible for packet sequencing. It is the IP module that encapsulates TCP segments inside IP packets and routes these packets ultimately to the destination host.

TCP calls the IP module which in turn calls the network device driver. TCP does utilize services of the operating system such as management of data structures.

In general, the categories of services provided by TCP include the following:

- Multiplexing
- Connection Management
- Data Transport
- Special Capabilities (such as data stream push and urgent data signaling)
- Error reporting

5.4.1. TCP Format

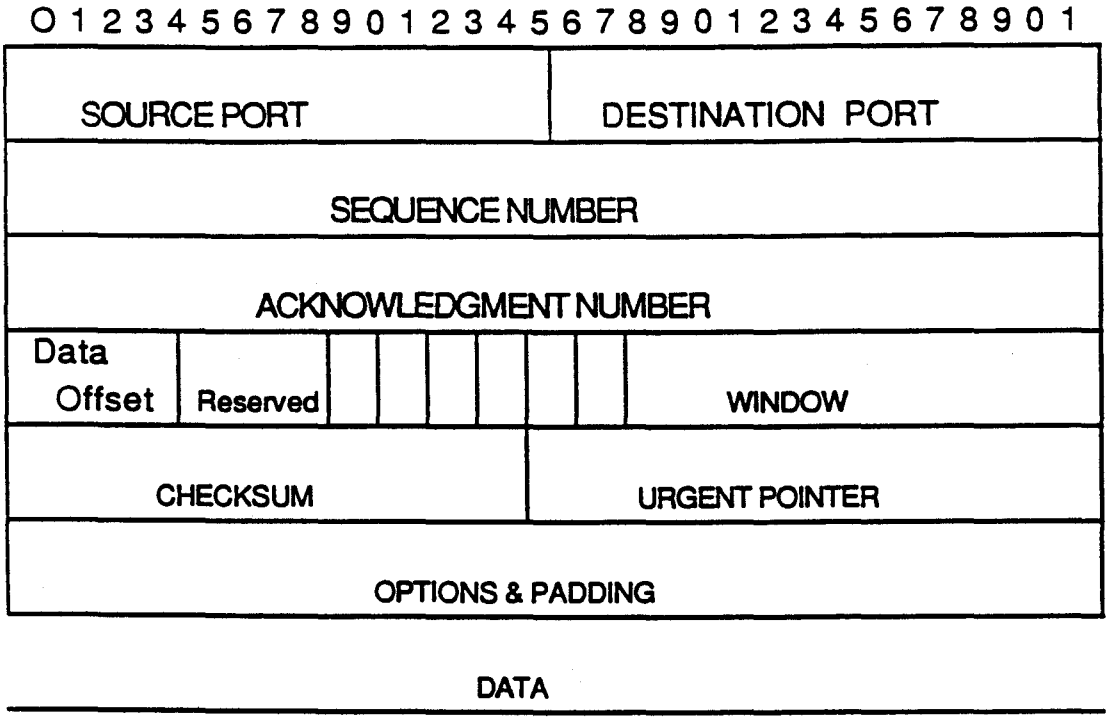


Figure 5-5: Transmission Control Protocol Packet Format

Figure 5-5 details the fields of the TCP datagram. The fields of the TCP datagram include:

- *Source Port (16 bits):* Identifies the source port.
- *Destination Port (16 bits):* Identifies the destination port.
- *Sequence Number (32 bits):* Sequence number of the first data octet in this segment, except when SYN (refers to synchronize sequence numbers) is present. If SYN is present the sequence number is the ISN (initial sequence number) and the first data octet is ISN+1.
- *Acknowledgment Number (32 bits):* A piggybacked acknowledgment that contains the sequence number of the next octet that the TCP entity expects to receive.

- *Data Offset (4 bits):* Number of 32-bit words in the header.
- *Reserved (6 bits):* Reserved for future use.
- *Flags (6 bits):*
    1. *URG:* Urgent pointer field significant
    2. *ACK:* Acknowledgment field significant
    3. *PHS:* Push function
    4. *RST:* Reset the connection
    5. *SYN:* Synchronize the sequence numbers
    6. *FIN:* No more data from sender
- *Window (16 bits):* Flow control credit allocation, in octets. Contains the number of data octets beginning with the one indicated in the acknowledgment field that the sender is willing to accept.
- *Checksum (16 bits):* Used for error detection.
- *Urgent Pointer (16 bits):* Points to the octet following the urgent data. Thus, the receiver can determine how much urgent data is coming.
- *Options (Variable):* Currently, only one option is defined, which specifies the maximum segment size that will be accepted.
- *Data (Variable):* Data field is a maximum of 65,535 octets.

5.5. User Datagram Protocol (UDP)

The User Datagram Protocol (UDP) provides a datagram form of communication at the transport layer. It is a protocol that fits in at layer 4 of the OSI/RM. In contrast to TCP, UDP provides a reliable stream oriented service. UDP does not provide congestion control or flow control; nor does it use acknowledgments or retransmit lost datagrams. These are functions normally expected of transport layer protocols. Thus, higher layer protocols, such as the Network File System (NFS) protocol, that use UDP must address problems related to congestion control, flow control or reliability. In essence, UDP provides higher layer protocols the ability to access the raw delivery service of IP.

UDP does provide checksumming to ensure data integrity. For applications that need the transport layer protocol to understand record boundaries, UDP is the right choice.

5.6. Internet Protocol (IP)

IP is a network layer protocol. Internet *datagrams* may traverse several networks before reaching their destination host. A *datagram* is a self-contained packet, independent of other packets, that does not require acknowledgment, and that carries information sufficient for routing from the originating host to the destination host. Since there is no explicit connection establishment phase, IP is said to be a *connectionless protocol.* The IP packet is routed transparently, but not necessarily reliably, to the destination host. It is the responsibility of the transport protocol, TCP, to ensure reliability.

5.6.1. Internet Protocol Format

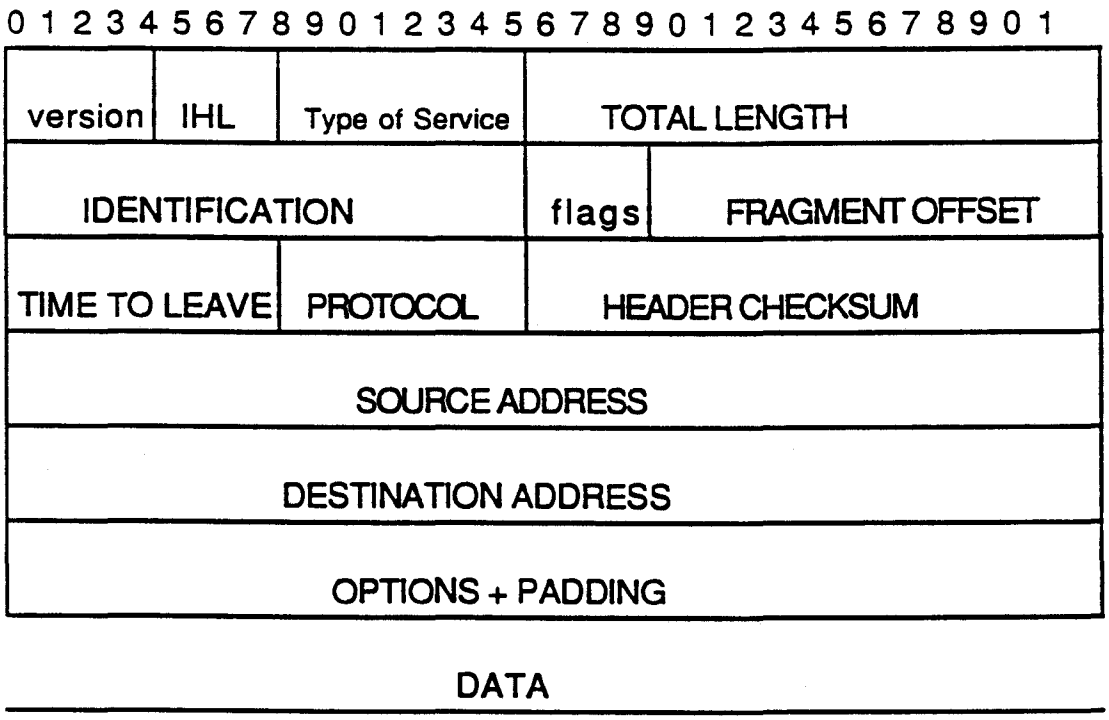


Figure 5-6: Internet Protocol Format

Figure 5-6 details the fields of the IP datagram. The fields of the IP datagram include:

- *Version (4 bits):* Version number; to allow for a transparent upgrade.
- *Internet Header Length (4 bits):* Length of IP packet header; minimum of 160 bits.
- *Type of service (8 bits):* Specifies reliability, precedence, delay, and throughput parameters.

- *Total length (16 bits):* Total datagram length, including header, in octets.
- *Identification (16 bits):* Together with source address, destination address, and user protocol intended to uniquely identify a datagram.
- *Flags (3 bits):* Flag bits used for fragmentation and reassembly.
- *Fragment offset (13 bits):* Indicates where in the datagram this fragment belongs. The unit measurement is 64 bits.
- *Time to live (8 bits):* Measured in 1-second intervals. Maximum of 255 seconds.
- *Protocol (8 bits):* Indicates the next level protocol that is to receive the data field at the destination.
- *Header checksum (16 bits):* Used for error detection.
- *Source address (32 bits):* Global network address consisting of a network identifier and a host identifier.
- *Destination address (32 bits):* Global network address consisting of a network identifier and a host identifier.
- *Options (variable):* Encodes options, such as security, source routing, error reporting, time stamping, debugging and others, as requested by the sender.
- *Padding (variable):* Used to ensure that the IP header ends on a 32-bit boundary.
- *Data (variable):* Data field must be a multiple of 8-bits in length. Total length of data field plus header is a maximum of 65,535 octets.

5.6.2. IP Operation

The transport layer takes messages passed on from higher layers and breaks them up into datagrams of up to 64K bytes each. Each datagram is transmitted through the Internet, possibly being fragmented into smaller units. The transport layer at the destination host is then responsible for datagram reassembly to form the original message.

5.6.3. ICMP

The Internet Control Message Protocol (ICMP) is a required part of the IP protocol and basically allows hosts and gateways on the Internet to report errors or provide information about unexpected circumstances. ICMP messages, encapsulated in the data portion of the IP packet, are ultimately targeted for the IP software module at the destination address. When the ICMP message arrives at the destination address, the IP software module handles the problem itself and

does not pass it on to the user process. Thus, ICMP provides a single mechanism for all Internet control and information messages.

5.7. Address Resolution Protocol (ARP)

Assume there are two hosts *cdsun1* and *cdsun2* on the network. Host *cdsun1* knows the Internet address for *cdsun2*, but does not know the physical hardware address of *cdsun2*. If host *cdsun1* wants to send a packet to host *cdsun2*, how does it translate the Internet address to *cdsun2's* physical address?

The Address Resolution Protocol (ARP) is a protocol used to solve the problem of translating Internet addresses to physical addresses such as Ethernet's 48-bit physical address.

Dynamic binding or resolution is used with ARP to solve the mapping problem. The following is an example of how dynamic binding with ARP works:

1. When host *cdsun1* needs to resolve Internet address for host *cdsun2*, it broadcasts a special packet that asks *cdsun2* to respond with its physical address.
2. Although all hosts on the network receive the request, only *cdsun2* recognizes its Internet address and responds with its physical address.

Hosts that use ARP maintain a cache of recently acquired Internet-to-physical address bindings so they do not have to use ARP repeatedly. Also, the sender's Internet-to-physical address binding is included in every ARP broadcast. Thus, receivers update the Internet-to-physical address binding information in their cache before processing an ARP packet.

The ARP message is encapsulated in Ethernet frames. Within Ethernet frames, the type field for ARP requests is set to 0806H, while ARP replies have a type field of 8035H. These values for ARP messages in the type field of the Ethernet field are universally accepted.

Figure 5-7 shows an ARP message encapsulated in an Ethernet frame.

Figure 5-7: ARP message encapsulated in an Ethernet frame

5.8. Reverse Address Resolution Protocol (RARP)

Reverse Address Resolution Protocol (RARP) is the protocol used by diskless machines to communicate with a server in order to determine its Internet address. The diskless system broadcasts its physical hardware address on the network. The RARP message (28 octets) is encapsulated in the data field portion of the Ethernet frame. The RARP message allows a machine to determine not only its own Internet address, but also that of other systems.

There must be at least one RARP server on the network for RARP to work. While all machines receive the request, only the RARP server processes the request and sends a reply. Typically, diskless machines rely on RARP to boot.

5.9. Routing Information Protocol (RIP)

The Routing Information Protocol (RIP) is used by 4.3 BSD UNIX systems to exchange routing information among a given set of systems on the network. The UNIX program, **routed** implements RIP on UNIX systems. The program **routed** changes routing tables in the operating system directly.

Gateways use RIP to broadcast their current routing database to the neighboring hosts on a periodic basis. The routing database consists of destination hosts (in gateway hops). RIP messages are either routing messages or debugging messages used to control tracing.

5.10. Simple Mail Transfer Protocol (SMTP)

The Simple Mail Transfer Protocol (SMTP) is the standard protocol for exchanging mail between Internet hosts. SMTP is not concerned with the content of mail messages except for the following two exceptions:

1. SMTP standardizes the message character set as 7 bit ASCII. For 8 bit transmission channels, the high order bit is set to 0.
2. SMTP adds information to the start of the delivered message. The information added specifies the route taken by the message.

The basic unit of SMTP activity is a TCP connection between a SMTP sender and a SMTP receiver. See chapter *Using TCP/IP* for examples of using SMTP to exchange mail with users on the Internet network.

5.11. Domain Names

Domain names are a hierarchical naming scheme used by the Internet. A domain name consists of a sequence of subnames separated by the delimiting character, the period. Consider the following Internet address:

`fnccf.fnal.gov`

Here the lowest level domain is **fnccf.fnal.gov** (indicating the Amdahl system at Fermilab). The top level domain is **gov** (for government institution). The second level domain is **fnal.gov**. The top level partitions defined by Internet are:

Domain Name	Description
COM	Commercial organizations
EDU	Educational institutions
GOV	Government institutions
MIL	Military groups
NET	Major networks support centers
ORG	Other organizations
ARPA	Temporary ARPANET domain
Country Code	Countries other than USA

Table 5-1: Internet Domains

5.12. Berkeley Internet Name Domain (BIND)

Berkeley Internet Name Domain (BIND) is a host name and address lookup service for information on the Internet network. BIND enables client systems to obtain host names and addresses from BIND servers. The BIND service consists of two components: the software interface called *name resolver* and the *name server.*

5.12.1. Name Servers and Name Resolvers

Name servers are used to map domain names such as fnccf.fnal.gov to Internet addresses such as 131.225.8.39. Typically, the name server software executes on a dedicated processor, and the machine itself is called the name server. A BIND name server is a system that runs the **named** daemon and can thus respond to BIND queries. BIND servers are of the following types:

Root Server: Root servers know about all the top-level domains (such as EDU, GOV) on the Internet network. Currently, there are seven root servers:

- ns.nasa.gov.
- sri-nic.arpa.
- a.isi.edu.
- gunter-adam.arpa.
- brl-aos.arpa.
- terp.umd.edu.
- c.nyser.net.

The period (.) at the end of each root server name indicates that the server name is not relative to the current domain. The root server name does not need any BIND name extensions to be appended.

Master Server: A master server is the authority for the current domain space and maintains the BIND data bases for its domain. It is possible that a server may be a master server for several domains, being the primary server for some domains and the secondary server for others. Each BIND domain, such as **fnal.gov**, should have at least two master servers, one primary and one or more secondary. The secondary servers act as a backup server in the event that the primary master server fails or is overloaded.

Caching Server: Caching servers have no authority for any domain. Cache servers service BIND queries by communicating with other servers such as a master server. Cache servers store the information received in a cache until the expiration date (specified in the ttl field of an IP datagram).

Forwarding Server:

Forwarding servers also known as *forwarders* have full access to the Internet network and are able to process requests by slave servers by communicating with root servers. Forwarders may be a primary or secondary master server or a caching only server. The configuration files on the slave servers define which systems the slaves will access as forwarders. Forwarders and slave servers are used in environments where it is not desirable for all local servers to access servers on the Internet network.

Slave Server: Slave servers normally do not have full access to the Internet network. Thus, if the slave server is not able to resolve a request from its cache, it cannot directly interact with the root server. Instead it forwards the query to a fixed list of forwarders.

Fermilab's primary name server is host FNBIT (Internet address 131.225.8.120). The secondary or backup name server for Fermilab is FNMFE (Internet address 131.225.17.150). *Name resolver* is the client software that uses one or more servers to translate Internet names. The name resolver consists of a group of routines that on a UNIX system generally resides in the C library **/usr/lib/libc.a**. The name resolver exchanges query packets with the BIND name server. BIND name servers have a name server daemon running in the background which services queries on a given network port. The standard port for TCP is specified on a UNIX system in the **/etc/services** file.

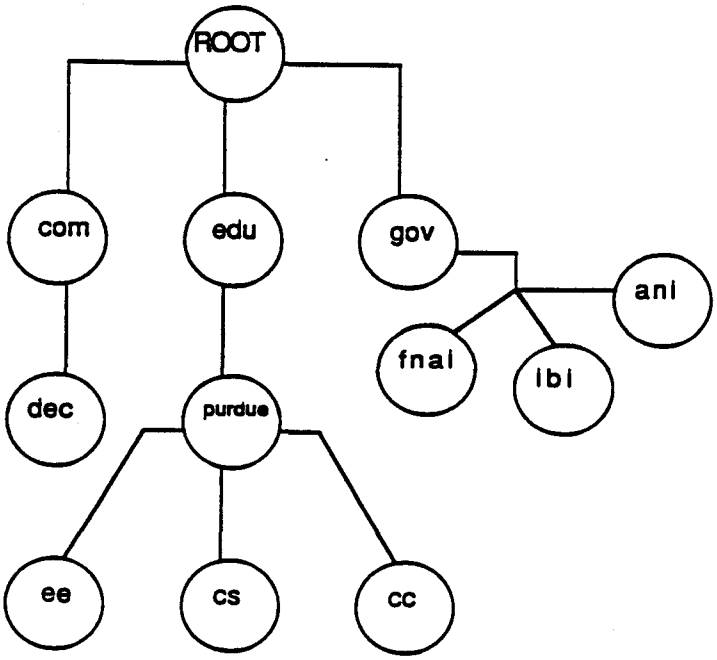


Figure 5-8: Internet Domain Name Servers

Figure 5-8 illustrates Internet domain name servers. At level 1 is the server that recognizes the top level domain. At level 2, are a set of name servers each of which recognizes subdomains such as **gov** and **edu**. At level 3, name servers such as FNBIT recognize subdomain **fnal.gov**.

How does it work?

The host seeking a server to translate a name to an address forms a domain name query that contains, besides other information, the name to be resolved. The domain name server upon receiving a request checks if the name is in the subdomain for which it is an authority. If it is, then the name is translated to an address according to it's database and sent to the client. If the server cannot resolve the name and if the client had requested a complete translation, the server contacts a higher level domain name server to resolve the name and sends the response to the client.

Since most name resolution refers to local names, translations begin with the local server. *Name caching* is a technique used to lower the cost of lookup for non-local names. Each server maintains a cache of recently used names as well as a record of where mapping information for that name was obtained. If the client sends a query for a name that is not in the server's subdomain, the server checks its cache to see if the name was resolved recently. The information, if determined from cache, is marked as *nonauthoritative* binding and sent to the client. Caching increases efficiency and works well because name to address bindings change infrequently.

5.12.2. BIND Clients

BIND clients are systems that make queries, but never resolve them locally. BIND servers, such as a master or slave server, resolve the client's requests. BIND clients do not run the **named** daemon. The only BIND file necessary is the name resolver file, **/etc/resolv.conf**. The following is an example of a typical **/etc/resolv.conf** file:

```
domain  fnal.gov
fnbit   131.225.8.120
fnmfe   131.225.17.150
```

5.13. Internet Gateways

A number of Cisco Systems' gateway servers have been installed at Fermilab. The gateway servers are also referred to as routers. Cisco Systems gateway servers support the following protocols: TCP/IP, X.25, DECnet and other protocols. Typically, gateway servers are high performance internetwork routers that can support multiple network and routing protocols concurrently; thereby, enabling communication between network equipment from different vendors.

Figure 5-9 shows the topology of Cisco gateway servers on the Fermilab local network.

The server supports digital circuits at 9.6 kbps and 19.2 kbps for synchronous serial service and 56 kbps for medium traffic connections. The gateway server also supports T1 circuits at 1.544 Mbps, T1C at 3.1 Mbps and British Telecom Megastream and CEPT DS1 circuits at 2.048 Mbps to 4.096 Mbps. The server supports satellite links, token rings, baseband and broadband coaxial cable, fiber optic links and dial-up links.

The gateway server monitors traffic on each network link and routes packets to the appropriate destination. The Interior Gateway Routing Protocol (IGRP), developed by Cisco Systems, monitors the network to determine the status of each route based on network traffic, path reliability and speed.

The server also understands messages from network segments that may be using Internet routing protocols such as RIP or HELLO. HELLO is the primary routing protocol used on the National Science Foundation network (NSFnet).

In the case where IP datagrams have to sent on an X.25 network, the gateway encapsulates the IP datagram within X.25 packets. The gateway server can also act as DECnet Level 1 and/or Level 2 router. The server supports DECnet and Internet routing on the same Ethernet segment.

5.13.1. Fermilab Router/Bridge Architecture

Figure 5-9 shows the topology of bridges and routers at Fermilab. Note that the two main backbone segments are the "highrise LAN" and "FCC hub."

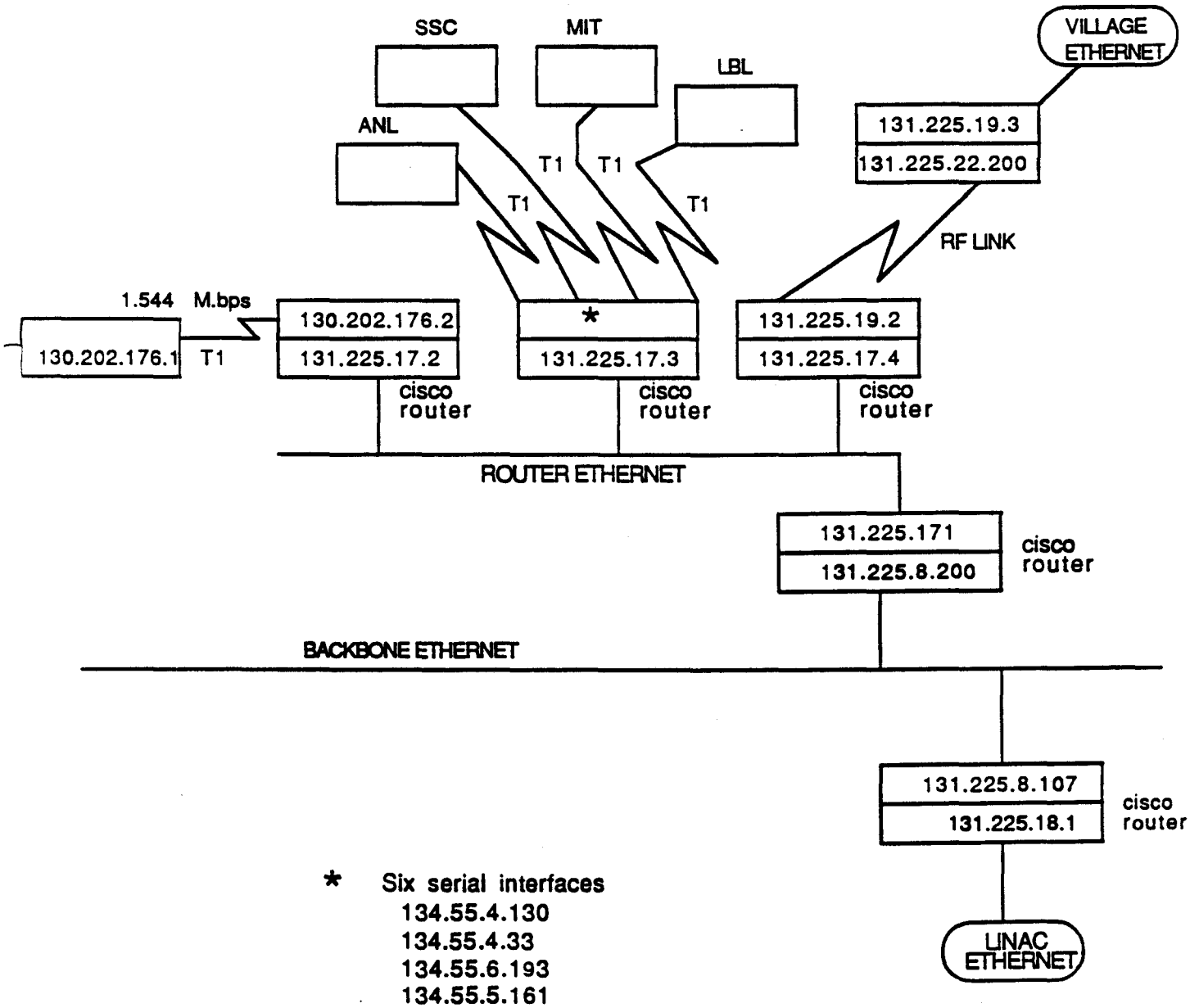
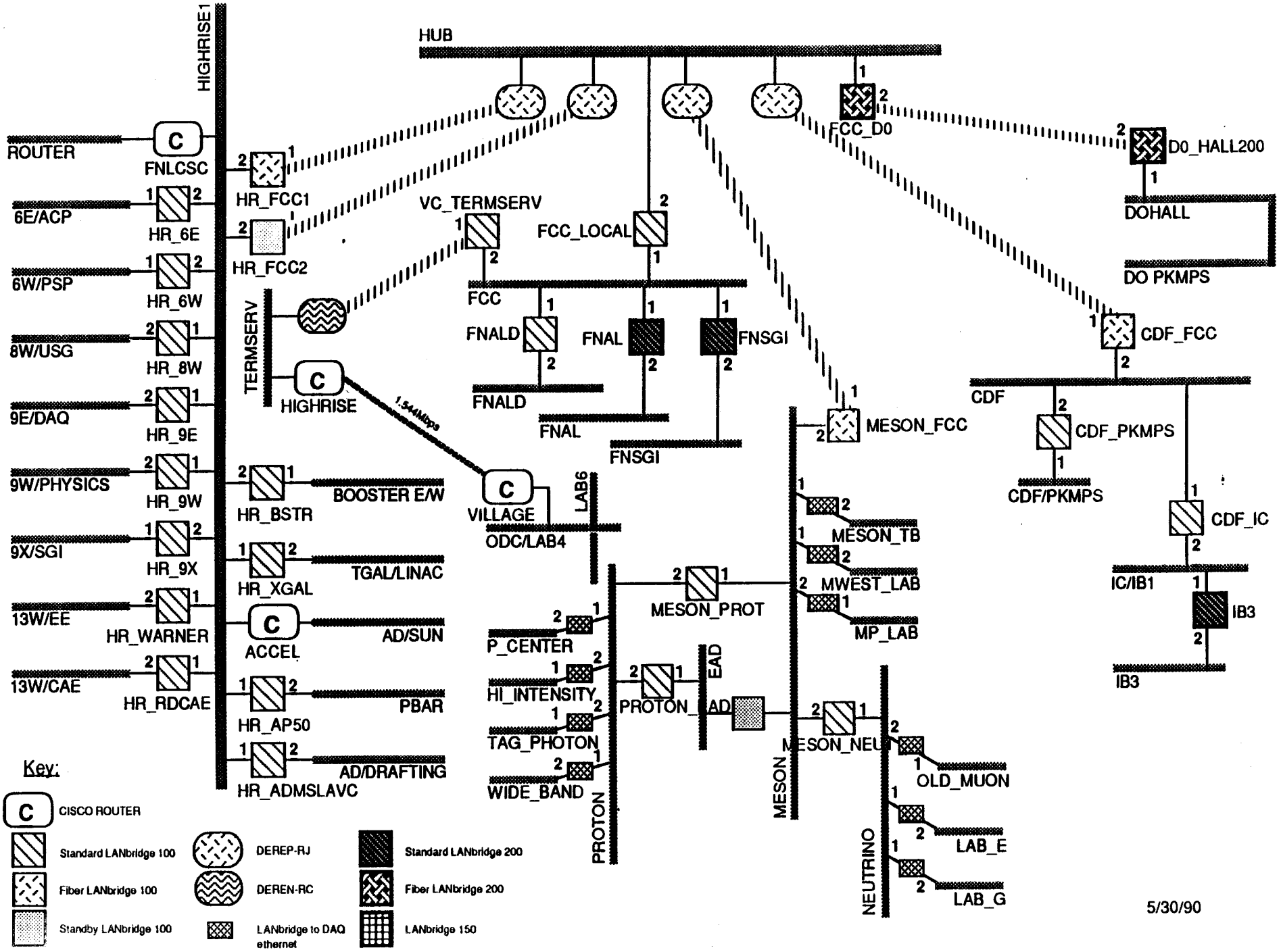


Figure 5-9: Cisco Routers (Gateways) topology at Fermilab

Fermilab Ethernet Bridge Architecture



5/30/90

5.14. References and Further Reading

1. Comer, D.. *Internetworking with TCP/IP.* Prentice Hall, Englewood Cliffs, NJ, 1988.

2. Stallings, W.. *Handbook of Computer-Communications Standards.* Macmillan Publishing Company, New York, 1988.

3. Tanenbaum, A.S.. *Computer Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1988.

This Page Intentionally Left Blank

Chapter 6

Using TCP/IP

The previous chapter described the Internet (DARPA) architecture and protocols. This chapter provides information on how the end-user can use applications and utilities based on the Internet network from VAX/VMS systems, an Amdahl running VM/CMS and UNIX platforms.

6.1. Getting an Internet Address for a System

In order to bring your system (VAX/VMS or UNIX) up on the Internet network you need to request an assigned Internet address from the Data Communications group. The Data Communications group requires the following information when you apply for an Internet address:

```
System node name:

Physical location of the system:

Operating system:

Ethernet Device (eg: MicroVAX - xqa0, Sun 3/260 - ie0):

Model name of system (eg: VAX 785, Sun 4/110):

System Manager:

Ethernet Physical Address:
```

6.1.1. Ethernet Device for VAX/VMS Systems

The Ethernet device information for a VMS system may be obtained by executing the following command:

```
$ SHOW DEVICE/FULL X          or

$ SHOW DEVICE/FULL ES
```

After executing the above command look for the following line:

```
Device xxxx: is online, network device.
```

xxxx is the Ethernet device type for the system you are logged onto. Specifically, the Ethernet device type for a Unibus system is **xea0**, for a Qbus system is **xqa0** and for a VAXstation 2000 it is **esa0.**

The Ethernet physical address for a VMS system may be obtained by executing the following sequence of commands:

```
$ RUN SYS$SYSTEM:NCP
NCP> SHOW EXEC STATUS
```

6.1.2. Ethernet Device for UNIX Systems

On a UNIX system, execute the following command to get information on the Ethernet device type:

```
# netstat -i
```

The Ethernet device type for a Sun 3/260 is **ie0**, for a HP system it is **lan0** and for a Silicon Graphics it is **ec0.**

6.2. From a VAX/VMS System

Software packages such as SRI International's and TGV's MultiNet and DIGITAL's VMS/ULTRIX Connection enable VAX/VMS systems to support the Internet networking protocols. The networking software typically consists of TCP/IP protocol software, network utilities and programming libraries. The protocol module consists of TCP, IP, ARP, ICMP and UDP. After the software has been installed on a VMS node, the user usually has access to the following applications:

- Use FTP to transfer files between VMS and UNIX systems.
- Use TELNET to establish a virtual terminal connection between VMS and UNIX systems.

- Use SMTP to transfer mail between VMS and UNIX systems.
- Use FINGER for information about users logged in to a local or remote system.
- Write distributed applications for process-to-process communications between VMS and UNIX systems.

6.2.1. Installing MultiNet

Fermilab has a site-wide license for TGV's MultiNet. MultiNet is distributed by the Distributed Computing Department in the Computing Division.

6.2.2. Getting Started with MultiNet

In order to receive on-line help on MultiNet, enter:

```
$ HELP MULTINET
```

On-line help is also available on commands such as TELNET and FTP.

In general, the format for a MultiNet command is:

```
$ MULTINET  command
```

where command specifies which MultiNet operation is to be performed.

6.2.3. Using TELNET

TELNET, a remote terminal protocol, allows you to establish a TCP connection to a login server at a remote site. TELNET passes keystrokes from the local node to the remote system. The TELNET client software allows the user to specify a remote node by Internet address as well as by domain name. Using TELNET to connect to a host on the Internet network is analogous to using SET HOST to connect to a node on DECnet.

The following example shows how the TELNET command is used to connect to node FNDAUA:

```
$ TELNET  131.225.32.12        (could also specify telnet fndaua)

Trying... Connected.


MIPS (fndaua)



login: sunrise
Password:
RISC/os  (UMIPS) 3.10 fndaua
Copyright 1986, MIPS Computer Systems
All Rights Reserved

$
```

If an Internet address or a remote host name is not specified with the TELNET command then at that point you are in the TELNET command mode. In the command mode you can enter the Internet address or the remote node name to connect to the remote host.

MultiNet TELNET and IBM 3270

The TN3270 mode under MultiNet TELNET uses the VAX/VMS screen management (SMG) runtime routines to create a full-screen IBM 3270 mode display on your terminal. It appears as though the end-user is logged on to a remote IBM system from an IBM 3270-class terminal.

By default TELNET automatically negotiates IBM 3270 emulation mode and enters in that mode only if the remote host supports it. The format for the TN3270 option are as follows:

```
/TN3270=AUTOMATIC (default)
       FORCE
/NOTN3270
```

Table 6-1 provides information on terminal window size to 3278 model mapping. TN3270 mode actually emulates an IBM 3278 terminal with the model number depending on the terminal "window" size (page width and length).

Minimum Size (Rows x Columns)	Emulated Terminal
24 x 80	3278 model 2
32 x 80	3278 model 3
43 x 80	3278 model 4
27 x 132	3278 model 5

Table 6-1: Terminal/Window Size to 3278 Model Mapping

Table 6-2 lists the mappings between 3270 function keys and the keys on VT100, VT200 and VT300 series terminals.

TN3270 Function Key Mappings

IBM Function	VT Terminal Key Sequences
Input Terminators	
Enter	CTRL/M *or* RETURN
Clear	CTRL/Z *or* ENTER
Input Editing Functions	
New line	DEL
Tab	TAB *or* CTRL/I
Backtab	CTRL/B
Left	CTRL/H *or* LEFT ARROW
Right	CTRL/L *or* RIGHT ARROW
Up	CTRL/K *or* UP ARROW
Down	CTRL/J *or* DOWN ARROW
Home	KEYPAD .
Delete	CTRL/D
Erase to EOF	CTRL/E
Erase Input	CTRL/W
Insert	CTRL/SPACE *or* ESC + SPACE
Attention Keys	
PA1	ESC + PF1 *or* CTRL/P + 1
PA2	ESC + PF2 *or* CTRL/P + 2
Local Control Keys	
TELNET Escape	CTRL/C
Master Reset	CTRL/G
Local Editing Keys	
Set Tab	ESC + ;

TN3270 Function Key Mappings	
IBM Function	**VT Terminal Key Sequences**
Delete Tab	ESC + \
Clear Tabs	ESC + :
Set Merge	ESC + ,
Set Home	ESC + .
Column Tab	ESC + DOWN ARROW
Coumn Back Tab	ESC + UP ARROW
Indent	ESC + RIGHT ARROW
Unindent	ESC + LEFT ARROW
Function Keys	
PF1	KEYPAD 1 *or* ESC + 1
PF2	KEYPAD 2 *or* ESC + 2
PF3	KEYPAD 3 *or* ESC + 3
PF4	KEYPAD 4 *or* ESC + 4
PF5	KEYPAD 5 *or* ESC + 5
PF6	KEYPAD 6 *or* ESC + 6
PF7	KEYPAD 7 *or* ESC + 7
PF8	KEYPAD 8 *or* ESC + 8
PF9	KEYPAD 9 *or* ESC + 9
PF11	PF1 + KEYPAD 1 *or* ESC + -
PF12	PF1 + KEYPAD 2 *or* ESC + =
PF13	PF1 + KEYPAD 3 *or* CTRL/F + 1 + 3
PF14	PF1 + KEYPAD 4 *or* CTRL/F + 1 + 4
PF15	PF1 + KEYPAD 5 *or* CTRL/F + 1 + 5
PF16	PF1 + KEYPAD 6 *or* CTRL/F + 1 + 6
PF17	PF1 + KEYPAD 7 *or* CTRL/F + 1 + 7
PF18	PF1 + KEYPAD 8 *or* CTRL/F + 1 + 8

TN3270 Function Key Mappings											
IBM Function	VT Terminal Key Sequences										
PF19	PF1	+	KEYPAD 9	*or*	CTRL/F	+	1	+	9		
PF20	PF2	+	KEYPAD 0	*or*	CTRL/F	+	2	+	0		
PF21	PF2	+	KEYPAD 1	*or*	CTRL/F	+	2	+	1		

Table 6-2: TN3270 Function Key Mappings

6.2.4. Using FTP

The File Transfer Protocol, FTP, may be used to transfer files between systems on the Internet network. Most implementations of TCP/IP support the FTP protocol. FTP also provides access to directories and files on local and remote hosts. In addition, FTP can convert among character representations such as EBCDIC and ASCII.

Getting Started with FTP

The following example illustrates how to get started with FTP.

```
$ FTP

FNALC.FNAL.GOV MultiNet FTP user process 2.1(91)

FTP> CONNECT CDSGI1.FNAL.GOV
<cdsgi1.fnal.gov FTP server (SGI 3.2 version 4.160 Aug 19 1989 04:53)

FTP> user pabrai
<Password required for pabrai
Password:
<User pabrai logged in.
CDSGI1.FNAL.GOV>
```

The FTP OPEN command could have also been used to connect to host CDSGI1.FNAL.GOV. As indicated in the example below, to get a listing of all valid commands, enter at the **FTP>** prompt: **?**.

```
FTP> ?

BELL                CONFIRM             CONNECT             EXIT
HASH                HELP                LCD                 LDIR
LIST                LOCAL-CD            LOCAL-DIRECTORY     LOCAL-PWD
LPWD                OPEN                PROMPT-ON-CONNECT   PUSH
QUIT                RETAIN              SET                 SPAWN
STREAM              TAKE                VERBOSE             VERSION
  or
Internet host name

FTP>
```

Transferring Files: An Example

To use FTP to transfer files to a UNIX node, FNDAUA, from a VMS node FNALC, enter:

```
$ FTP FNDAUA
```

At this point, you should see something similar to the following:

```
FNALC.FNAL.GOV  MultiNet FTP-32 user process 1.1(47)
Connection opened (Assuming 8-bit connections)
<fndaua FTP server (UMIPS - from 4.3BSD) ready.
FTP>
```

Next, at the FTP prompt enter the remote node (UNIX system in this case) userid and password.

```
FTP> user  netguru
Password:
<User netguru logged in.
```

At this point, you are connected to the remote system and may transfer files between the local and remote nodes. To get a listing of files on the UNIX system, enter:

```
FTP> dir

<Opening data connection for /bin/ls (131.225.8.44,1053) (0 bytes).
-rw-r--r--   1 netguru  fermilab     298 Jun  2 09:26 login.com
-rw-r--r--   1 netguru  fermilab     517 Jun  2 09:28 lsr8print1.for
-rw-r--r--   1 netguru  fermilab      78 Jun  2 09:19 mbox
-rw-r--r--   1 netguru  fermilab     301 Jun  2 09:28 net_image.for
-rw-r--r--   1 netguru  fermilab     517 Jun  2 09:28 target1.for
<Transfer complete.
```

You may also use the ls command to get information on files on the remote node. The following is an example of using the command and sample output:

```
FTP> ls

<Opening data connection for /bin/ls (131.225.8.44,1054) (0 bytes).
-rw-r--r--   1 netguru  fermilab     293 Jun  2 09:28 login.com
-rw-r--r--   1 netguru  fermilab     517 Jun  2 09:28 lsr$print1.for
-rw-r--r--   1 netguru  fermilab      73 Jun  2 09:19 mbox
-rw-r--r--   1 netguru  fermilab     301 Jun  2 09:28 net_image.for
-rw-r--r--   1 netguru  fermilab     517 Jun  2 09:28 target1.for
<Transfer complete.
```

The get command may be used to transfer a single file from a remote node (the node you have connected to) to the local node. The put command may be used to transfer a single file from the local node to a remote node. The format of the commands follows:

FTP> get remote_file local_file
(Indicates, get a file from the system you have connected to and copy it to the local system.)

FTP> put local_file remote_file
(Indicates, put or copy a file from the local system to the system you have connected to.)

For example, to transfer a single file, *log.file* from the VMS node, FNALC, to the UNIX node, FNDAUA, use the put command:

```
FTP> put log.file login.com

[Listening as host 131.225.8.44, port 4.41]
PORT 131,225,8,44,4,41
<200 PORT command successful.
STOR login.com
<150 Opening data connection for login.com (131.225.8.44,1065).
<226 Transfer complete.
300. bytes transferred at 240000. Baud.
Run time = 10. ms, Elapsed time = 10. ms.

FTP>
```

The mget command may be used to transfer multiple files from a remote node to a local node. The mput command may be used to transfer multiple files from the local node to a remote node. The format of the commands follows:

FTP> mget remote_files

FTP> mput local_files

For example, if you are currently logged on to the VMS node, FNALC, and need to copy all files with an extension of *.for* from the UNIX node, FNDAUA, enter at the FTP prompt:

```
FTP>mget  *.for

<Opening data connection for /bin/ls (131.225.8.44,1055) (0 bytes).
<Transfer complete.
<Opening data connection for lsr$print1.for (131.225.8.44,1056) (517 bytes).
<Transfer complete.
<Opening data connection for net_image.for (131.225.8.44,1057) (301 bytes).
<Transfer complete.
<Opening data connection for target1.for (131.225.8.44,1058) (517 bytes).
<Transfer complete.
```

To spawn out of FTP and execute DCL commands, enter **PUSH** or **SPAWN** at the FTP prompt. To return back to the FTP prompt, enter **EOJ** at the VMS **$** prompt.

Transferring Binary Files

In order to transfer a binary file, you need to use the **type** command. The following example illustrates:

```
FTP>open cdsun1
CDSUN1.FNAL.GOV>user pabrai
Password:
<User pabrai logged in.
CDSUN1.FNAL.GOV>help type
The TYPE command is used to change the FTP transfer type.  The
possible arguments to the TYPE command are ASCII, IMAGE, BACKUP and
LOGICAL-BYTE.  ASCII type is used for transferring ASCII text files.
IMAGE type is used for transferring binary files.  BACKUP is used for
transferring VAX/VMS backup savesets with 2048 byte block size.

CDSUN1.FNAL.GOV> type image
Type: Image, Structure: File, Mode: Stream

ftp>
```

FTP Command Scripts

FTP commands can be executed as part of a DCL command procedure. Consider the file, *putfile.com*, that contains the following statements:

```
set cdsgi1.fnal.gov /user:pabrai /pass:uop0257
connect cdsgi1.fnal.gov
put cdsgi1.login .login
exit
```

To execute the command procedure, *putfile.com*, enter the following:

```
$  FTP  /TAKE=PUTFILE.COM
```

Note that since the file, *putfile.com*, contains the password to the remote system it should be protected appropriately.

FTP Initialization File

When FTP is invoked it executes commands in the FTP.INIT file in your login area. Typical commands that may be extered in that file are:

BELL ON — Rings the terminal bell when a file transfer operation is completed.

HASH ON — Prints a hash mark (#) for each data buffer transferred.

PROMPT-ON-CONNECT ON — Automatically prompts for a username and password when a connection to the remote system is established.

SET *host*/USERNAME:*user* — Sets the default username for the specified host.

VERBOSE ON — Displays all responses from the remote FTP server as they are received and upon completion of file transfers, displays timing statistics.

6.2.5. Using MultiNet PING

The ping utility may be used to verify that a path between the local node and a remote node is available. The utility also provides performance statistics and error rate over the link used. Both, network packet loss and latency are measured. The utility uses *ICMP Echo Request* packets to "ping" the specified host. The format of the command is:

```
$ MULTINET PING host_name      or

$ MULTINET PING ip_address
```

For example, to verify that a path exists between the CISCO router, CISC1.FNAL.GOV (Internet address 131.225.8.200), enter at the VMS $ prompt:

```
$ MULTINET PING 131.225.8.200

PING 131.225.8.200: 56 data bytes
64 bytes from 132.225.8.200:  icmp_seq=1. time=10. ms
64 bytes from 132.225.8.200:  icmp_seq=1. time=10. ms
64 bytes from 132.225.8.200:  icmp_seq=1. time=10. ms
64 bytes from 132.225.8.200:  icmp_seq=1. time=10. ms
64 bytes from 132.225.8.200:  icmp_seq=1. time=10. ms
64 bytes from 132.225.8.200:  icmp_seq=1. time=10. ms

CTRL/C



----131.225.8.200 PING Statistics----
7 packets transmitted, 6 packets received, 14% packet loss
round-trip (ms)  min.avg/max = 0/5/10

$
```

Note that using CTRL/C to terminate the utility results in ping statistics. The ping utility is a good way to determine if a path between the local node and remote node exists. The remote-node may be specified with a node name or an IP address.

6.2.6. Using NSLOOKUP

MultiNet NSLOOKUP is a utility that allows a user on a VMS system to query a remote or local nameserver. The format of the MultiNet NSLOOKUP command is:

$ MULTINET NSLOOKUP [name] [nameserver]

where **name** is a host or domain name and **nameserver** is the nameserver to be queried for information. If MULTINET NSLOOKUP is entered without any arguments, it enters an interactive mode. The following commands may be executed at the NSLOOKUP > prompt:

If, for example, you had an address in symbolic form such as HUHEPL.HARVARD.EDU it might be useful to you to get the numeric equivalent of this address; this could be accomplished by using the following command sequence:

```
$ MULTINET NSLOOKUP HUHEPL.HARVARD.EDU

Server:  cso.uiuc.edu
Address:  128.174.5.50

Name:    HUHEPL.HARVARD.EDU
Address:  128.103.1.140
```

Command		Description
name		Print information about *name* using the default server
name server		Print information about *name* using *server*
help		Print help information
set	*option*	Set an option
	all	Dumps the Domain Nameserver cache to the file MULTINET:DOMAIN-NAME-SERVICE.DB
	[no]debug	Print debugging information
	[no]d2	Print exhaustive debugging information
	[no]defname	Appebnd domain name to each query
	[no]recurse	Ask for recurseive answer to query
	[no]vc	Always use a virtual circuit
	domain=*name*	Set default domain name to *name*
	root=*name*	Set root nameserver to *name*
	retry=*n*	Set number of retries to *n*
	timeout=*n*	Set time-out interval to *n*
	querytype=*type*	Set query type to one of A, CNAME, HINFO, PTR, MINFO, MR, MX, SOA
	type=*type*	Set query type to one of A, CNAME, HINFO, PTR, MINFO, MR, MX, SOA
server *name*		Set default server to *name*, using current default server
lserver *name*		Set default server to *name*, using current default server
root		Set current default server to the root
ls *name* [>*file*]		List the domain *name*, with output optionally going to *file*.

The above example illustrates how the numeric equivalent for the name HUHEPL.HARVARD.EDU may be obtained. The server, CSO.UIUC.EDU, responds by identifying itself, and then provides the numeric equivalent, 128.103.1.140.

Appendix C provides a complete list of Internet addresses assigned to Fermilab nodes. In order to get a listing of all Fermilab Internet addresses, execute the following commands:

```
$ MULTINET NSLOOKUP

Default Server: FNBIT.fnal.gov
Address:  131.225.8.120

>  LS FNAL.GOV
```

In the above command **fnal.gov** is the domain for all of Fermilab's Internet nodes. The output of **ls fnal.gov** is:

```
[FNNET.fnal.gov]
Host or domain name              Internet address
 fnal                             server = FNAL.GOV
 DIPOLE                           131.225.8.174
 MDTF00                           131.225.8.162
 LOCALHOST                        127.0.0.1
 EROS                             131.225.8.154
 MDTF04                           131.225.8.152
 MDTF05                           131.225.8.153
  .
  .
  .
 FNUSG4                           131.225.8.111
 FNUSG5                           131.225.8.112
 FNUSG6                           131.225.8.61
 FNUSG7                           131.225.8.113
 ACPM1                            131.225.8.3
 FNUSG8                           131.225.8.76
 ACPM2                            131.225.8.10
 ODIN                             131.225.8.150
```

If you would like to have the listing written to a file, enter:

ls fnal.gov > filename.filetype

Press CTRL/Z to exit from the **nslookup >** prompt.

6.2.7. Using MFINGER/FINGER

MultiNet's FINGER utility is typically installed as MFINGER on a VAX/VMS system. To determine what MFINGER and FINGER point to, execute the following command:

```
$ SHOW SYMBOL MFINGER

$ SHOW SYMBOL FINGER
```

The MFINGER utility provides information about users logged onto the local or remote system. The format of the MFINGER command is:

```
$ MFINGER user@host
```

For example, to get information on users logged on host SSCVX1.SSC.GOV, enter:

```
$ MFINGER @SSCVX1.SSC.GOV


Wednesday, February 14, 1990 1:19PM-CST   Up 18 17:03:55
4+8 Jobs   Load ave  2.10 2.06 2.05

User     Personal Name        Job  Subsys Idle TTY      Console Location

AGRAW    Anet Graw            6AA0 TELNET     18.nty9  TCP: [134.3.129.136]
AKA      Kathy Anderson       70B6 MAIL       .lta19   LAT: LAT_0000B5000A6F
POALLEN  Michael Allen        4865 *DCL*      5.nty19  TCP: [134.3.130.56]
ANOBREGA Alfred Nobrega       3698 *DCL*      .lta12   LATST01/PORT_12
BALLAM   Joe Ballam           66E0 *DCL*      .lta19   LAT: VIST09/PORT_29
BOBLEEDY Robert Leedy         6ECE *DCL*      14.nty30 TCP: doc.ssc.gov
BULL     Jeff Bull            68C7 EDT        .lta19   LAT: VIST09/PORT_26
```

To finger a single user on the local system, enter:

```
$ MFINGER MULVEY
```

If the user's login directory contains the file PLAN.TXT then that file is displayed when the MFINGER command is invoked. Note that you must set the file protection to world readable for MFINGER to display the information in it.

To display information about all users currently logged in to the local system, enter:

```
$ MFINGER
```

6.2.8. Using RSHELL

The RSHELL utility enables you to execute commands on remote UNIX or VAX/VMS (that have MultiNet installed) systems. To execute a command on a remote system, the remote host must list the local system in its **/etc/hosts.equiv** file. Alternatively, you may list the name of the local system in the **.rhosts** file in your login area on the remote system. For example, to get a listing of files in your HOME area on the Sun SPARCstation *cdsun1*, enter:

```
$ rshell cdsun1 ls -l
```

This assumes that the login username on the VMS system matches that on the Sun SPARCstation. If the usernames are different then use the **/username** qualifier. For example, if the username on the remote system is **shiva** then enter:

```
$ rshell/username=shiva cdsun1 ls -l
```

The Rshell command terminates when the execution of the remote command has terminated. To terminate a previously issued command, press CTRL/C.

6.2.9. Using RLOGIN

The RLOGIN utility may be used to connect to a remote system on the Internet network. The protocol is similar to TELNET. Unlike TELNET, RLOGIN automatically authenticates the user instead of requesting **username**. The **/username** qualifier may be used if the username is different on the remote system. Local and remote flow control are negotiated dynamically. For example, to connect to the Sun SPARCstation *cdsun1* enter:

```
$ rlogin cdsun1
```

If the username on the remote system is **nehru**, then enter:

```
$ rlogin/username=nehru cdsun1
```

If you would like to login to a remote system without specifying a username or a password then specify the name of the local system in the **.rhosts** file in your login area on the remote host. The following list of special characters may be used during the RLOGIN session:

```
~.   Disconnect and exit
~^Z  Spawn a subprocess
~~   To send a single ~ to remote system
```

~^Z is especially useful since it allows you to spawn a subprocess and thereby connect back to the VMS system.

6.2.10. Other MultiNet Commands

On almost all VMS hosts running MultiNet the device over which Internet traffic is transmitted is called *se0*. The command **multinet show/interface se0** provides information on the local node Internet address, subnet mask and broadcast address.

```
$ multinet  show/inter  se0

Device se0: Address=131.225.8.110, flag=43<UP,BROADCAST,RUNNING>
            Sub-Net Mask = 255.255.0.0
            Broadcast Address = 131.225.255.255
```

To get routing table information, such as the default gateway for Fermilab, enter the following:

```
$  multinet show/route
```

```
MultiNet IP Routing tables:
Destination      Gateway          Flags      Refcnt Use    Interface
-----------      -------          -----      ------ ---    ---------
LOCALHOST        LOCALHOST        Up,Host    0      0      lo0
DEFAULT-GATEWAY  131.225.8.200    Up,Gateway 1      63     se0
FERMILAB         FNUSG3.FNAL.GOV  Up         0      1542   se0
```

6.3. Exchanging Mail with Internet Addresses

MultiNet extends the VAX/VMS MAIL utility to support messages to be exchanged between nodes on the Internet network. MultiNet utilizes the *Simple Mail Transfer Protocol*, SMTP, to communicate between various nodes on the Internet network. The format for sending mail using SMTP is:

SMTP%''recipient@destination'' *or*

SMTP%''recipient@[aa.bb.cc.dd]''

where "aa.bb.cc.dd" is the destination system's Internet address in dotted-decimal notation (eg. 131.225.8.77).

For example, if you wanted to send mail to user NETMAN at Internet host HUHEPL.HARVARD.EDU, you would do the following:

```
$ MAIL
MAIL> SEND
To:   SMTP%"NETMAN@HUHEPL.HARVARD.EDU"
Subj: makin' the connection
Enter your message below.  Press CTRL/Z when complete, CTRL/C to quit:
Please respond to the Network Budget Proposal asap.

CTRL/Z

MAIL>
```

Note that GOV, COM, and NET are other Internet domains that can also be referenced in a similar manner. It is also possible to send mail to users on remote systems by just specifying the Internet address. For example, to send mail to user, GANDHI, at Internet address 131.225.8.108, use the following format:

```
$ MAIL
MAIL> SEND
To:    SMTP%"GANDHI@[131.225.8.108]"
Subj: Inspiration
Enter your message below.  Press CTRL/Z when complete, CTRL/C to quit:
It is always inspiring to think of your contribution to mankind!


CTRL/Z

MAIL>
```

Note that some UNIX mail servers may not accept address specification in the above format.

6.4. From an Amdahl VM/CMS System

The section on *From a VAX/VMS System* provided a general description of application protocols such as TELNET, FTP, RLOGIN, RSHELL, SMTP, PING, NSLOOKUP and FINGER. The applications available on the node FNCCF, Amdahl 5890-600E running VM/XA SP2, are: TELNET, FTP, TCPSEND, TCPNOTE and PING.

The objective of this section is not to repeat information on application services described earlier, but to highlight additional command options or other major differences.

A Bus-Tech, Inc. (BTI) controller is channel attached to the Amdahl. The controller is also connected to the Ethernet network. IBM's TCP/IP for VM product provides support for the following protocols: TCP, IP, UDP, ICMP, ARP, FTP, TFTP, SMTP and TELNET.

6.4.1. The INTERNET Product on the Amdahl

To access the Internet network from the Amdahl, enter:

SETUP INTERNET

The Internet host name for the Amdahl is FNCCF.FNAL.GOV and the corresponding Internet address is 131.225.8.39. Enter **HELP INTERNET** to get on-line help on the Internet product.

6.4.2. Using TCPSEND

TCPSEND may be used to send files to nodes on the Internet network. The command syntax is similar to the VM SENDFILE command. The format of the TCPSEND command is:

```
TCPSEND filename filetype filemode TO userid AT internet-node
```

For example, to send a copy of your *profile exec* file to user DOWAT on Internet node FNALB, enter:

```
TCPSEND PROFILE EXEC A TO DOWAT AT FNALB.FNAL.GOV
```

6.4.3. Using TCPNOTE

TCPNOTE may be used to send mail messages to users on the Internet. The command syntax is similar to the VM NOTE command. The format of the command is:

```
TCPNOTE userid AT internet-node
```

For example to send a mail message from the Amdahl to user JYOTI on Internet host UAEIP1.WRL.COM, enter at the READY prompt:

```
TCPNOTE JYOTI AT UAEIP1.WRL.COM
```

6.5. From a UNIX system

The section on *From a VAX/VMS System* provided a general description of application protocols such as TELNET, FTP, RLOGIN, RSHELL, SMTP, PING, NSLOOKUP and FINGER. All of these applications are available on UNIX systems. The objective of this section is not repeat information on application services described earlier, but to highlight additional command options or other major differences.

On UNIX systems command options are usually single letters preceded by a dash. Sometimes it is possible to combine options and precede them with one dash or specify them separately and precede each with a dash. Some options appear as uppercase characters.

6.5.1. Using TELNET

To connect to any system on the Internet network simply use the TELNET command. For example, to connect to host FNALB.FNAL.GOV from Sun SPARCstation CDSUN1.FNAL.GOV, enter:

```
% telnet fnalb.fnal.gov
```

6.5.2. Using FTP

MultiNet provides a limited set of FTP commands on a VMS system. Typically, UNIX systems provide the following FTP commands.

```
ftp> help

Commands may be abbreviated.  Commands are:
!               cr              macdef          proxy           send
$               delete          mdelete         sendport        status
account         debug           mdir            put             struct
append          dir             mget            pwd             sunique
ascii           disconnect      mkdir           quit            tenex
bell            form            mls             quote           trace
case            hash            nmap            rename          verbose
cd              help            ntrans          reset           ?
cdup            lcd             open            rmdir
close           ls              prompt          runique

ftp>
```

The **status** provides information on the default parameters for FTP commands. MultiNet also supports the STATUS command.

```
ftp> status

Connected to 131.225.8.44.
No proxy connection.
Mode: stream; Type: ascii; Form: non-print; Structure: file
Verbose: on; Bell: off; Prompting: on; Globbing: on
Store unique: off; Receive unique: off
Case: off; CR stripping: on
Ntrans: off
Nmap: o
```

FTP Command Options On a SunOS or IRIX System

The FTP command options available on a SunOS based system (eg. SPARCstation 1) or a IRIX system (eg. Silicon Graphics 4D/20) are:

-d Enable debugging.

-g Disable filename globbing.

-i Turn off interactive prompting during multiple file transfers.

-n Do not attempt auto-login upon initial connection. If auto-login is enabled, FTP checks the **.netrc** file in the user's home directory for an entry describing an account on the remote machine. If no entry exists, FTP prompts for the login name of the account on the remote machine.

-v Show all responses from the remote server, as well as report on data transfer statistics. This is turned on by default if FTP is running interactively with its input coming from the user's terminal.

An Example Session

Here's an example of using FTP to connect to a VAX/VMS node, FNALE.

```
$ ftp
ftp> open 131.225.8.44
Connected to 131.225.8.44.
220 FNALE.FNAL.GOV MultiNet FTP Server Process 1.0(27) at Tue 6-Jun-89
Name (131.225.8.44:netustad) : netustad
331 User name (netustad) ok. Password, please.
Password:
230 User NETUSTAD logged in at Tue  6-Jun-89 14:52, job 20800f44.
```

To receive a listing of FORTRAN files in the top-level directory on node FNALE, enter:

```
ftp> dir *.for
200 Port 4.18 at Host 131.225.32.12 accepted.
150 List started.

USR$ROOT36:[NETUSTAD]

LSR$PRINT1.FOR;2                    2     2-JUN-1989 15:59
NET_IMAGE.FOR;2                     1     2-JUN-1989 15:59
RECEIVE.FOR;1                       6    27-FEB-1989 15:50
SEND.FOR;1                          5    27-FEB-1989 15:50
TARGET1.FOR;2                       2     2-JUN-1989 15:59


Total of 16 blocks in 5 files

226 Transfer completed.
remote: *.for
379 bytes received in 0.33 seconds (1.1 Kbytes/s)
ftp>
```

6.6. What are RFC's?

Information on the Internet such as architecture, protocols, current developments can be found in a series of reports known as *Request For Comments* or (RFCs). RFCs are available electronically from the Internet Network Information Center (NIC) in the online library at NIC.DDN.MIL (Internet address: 192.67.67.20).

6.6.1. Obtaining an RFC

RFCs can be obtained via FTP from NIC.DDN.MIL. Here are the steps that you need to follow, in general, to obtain copies of RFCs:

1. Use FTP to connect to NIC.DDN.MIL.
2. Login with username ANONYMOUS and password GUEST.
3. Use the FTP **get** command to copy the RFC. The pathname is RFC:RFCnnnn.TXT (where "nnnn" refers to the number of the RFC).
4. Exit FTP.

For example, to get a copy of RFC 1140: IAB Official Protocol Standards, enter the following from a host that is connected to the Internet network:

```
$ ftp  nic.ddn.mil

Connection opened (Assuming 8-bit connections)
<NIC.DDN.MIL FTP Server Process 5Z(47)-6 at Wed 30-May-90 13:34 PDT

FTP> user anonymous
<ANONYMOUS user ok, send real ident as password.
Password:

<User ANONYMOUS logged in at Wed 30-May-90 13:34-PDT, job 18.

FTP> get rfc:rfc1140.txt 1140.txt

<ASCII retrieve of TS:<RFC>RFC1140.TXT.1  (24 pages) started.
<Transfer completed. 60501 (8) bytes transferred.

FTP> quit
```

6.7. Accessing the UUCP Network

A large number of UNIX hosts are on the UNIX-to-UNIX COPY (UUCP) network. Fermilab does not at present have any nodes on the UUCP network. To send mail to users on the UUCP network use the following syntax:

smtp%"machine_name!username@oddjob.uchicago.edu"

where machine_name is the node name of the remote system and username is the name of a user on the system. You can also use the BITNET to Internet gateway, PSUVAX1, to communicate with UUCP hosts. To get more information on this gateway, enter the following:

$ SETUP GMAIL *on the FNAL cluster*

GMAIL> help gateways *read portion on UUCP gateways*

For example, to send mail to user *shaku* at UUCP address **att!iexist!shaku**, use the following address format in mail:

shaku@iexist.att.com

This assumes that node **iexist** is both on the UUCP and Internet network. Thus, on a VMS system that is on the Internet network, a user could send mail to the above address by specifying in VAX mail:

To: **smtp%"shaku@iexist.att.com"**

6.8. References and Further Reading

1. Comer, D.. *Internetworking with TCP/IP.* Prentice Hall, Englewood Cliffs, NJ, 1988.

2. Stallings, W.. *Handbook of Computer-Communications Standards.* Macmillan Publishing Company, New York, 1988.

3. *MultiNet Users' Guide.* TGV, 1990.

Chapter 7

Network File System

The Network File System (NFS) protocol enables file systems of remote hosts to appear as though they were attached to the local system. NFS, designed and developed by Sun Microsystems, Inc., was introduced in 1984. Remote Procedure Calls (RPC) and eXternal Data Representation (XDR) are the underlying protocols used by NFS. XDR corresponds to the presentation layer of the OSI/RM. RPC does not quite fit into the OSI/RM; it does provide the functionality of the session layer. Figure 7-1 illustrates the relationship between NFS, RPC, XDR and the OSI/RM.

NFS has been ported to ULTRIX, VAX/VMS, UNICOS (Cray), UTS (Amdahl), AIX (IBM), VM (IBM), MS-DOS and a number of other systems. NFS has been licensed to run on over a 100 different computer systems.

7.1. Design and Working

NFS is designed in terms of a set of procedures, arguments, their results and effects. NFS uses the RPC mechanism to implement remote services. NFS uses User Datagram Protocol (UDP) and Internet Protocol (IP) as its transport and network layer protocol respectively.

7.1.1. Stateless Protocol

A *stateless protocol* is one that does not keep track of past requests. NFS uses a stateless protocol. Each procedure call contains all the information necessary to complete the call. The server does not "remember" past requests. Thus, if the server crashes, the client system repeats NFS requests until a response is received. The server does no crash recovery. If the client crashes, no recovery is necessary for either the client or the server. Complex crash recovery transactions between the server and client are thus avoided. Note that the client cannot tell the difference between a a server that has crashed and recovered and a slow server.

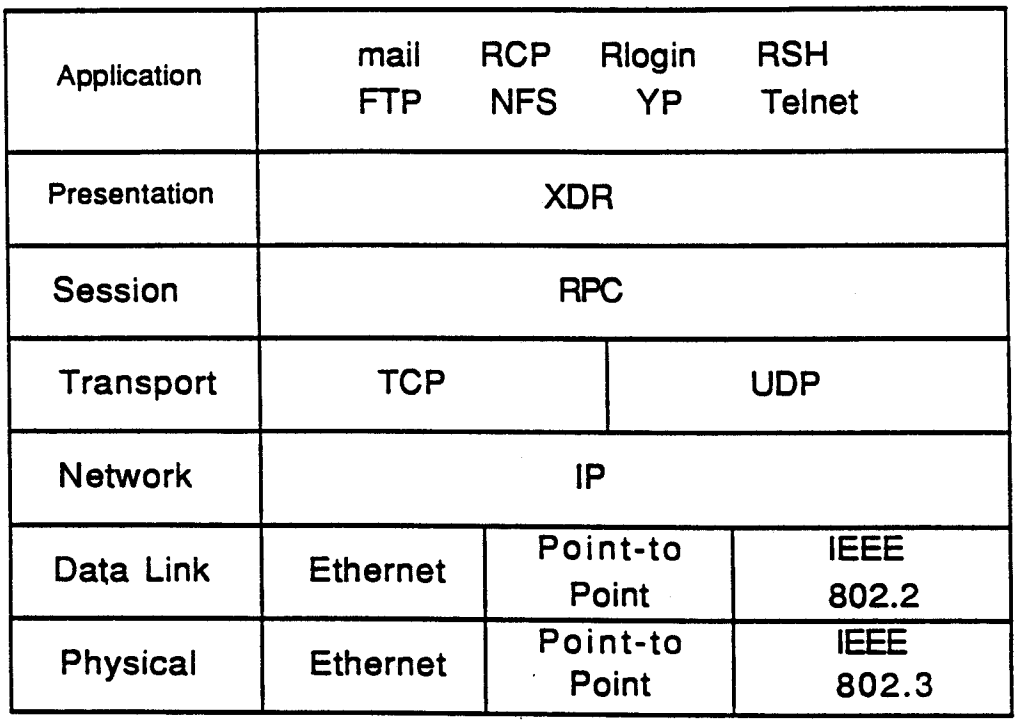


Figure 7-1: NFS, XDR, RPC and the OSI/RM

7.1.2. NFS Procedures

A structure called a *file handle* is the most common NFS procedure parameter. It is provided by the server and is used by the client to reference a file. The client never looks at the contents of a file handle. Table 7-1 provides a brief description of NFS protocol procedures.

NFS Procedure	Description
null()	Ping server and measure round trip time.
lookup()	Returns new file handle for the named file in a directory. This procedure is referred to as getfh() on MIPS systems.
create()	Creates a new file and returns its file handle and attributes.
remove()	Remove file from directory.
getattr()	Returns file attributes.
setattr()	Sets the mode, UID, GID, size, access time and modify time of a file.
read()	Returns data from a file. Returns file attributes.
write()	Writes data to a file. Returns file attributes.
rename()	Changes name of (moves) a file.
link()	Creates a link to a file on the remote host.
symlink()	Creates a symbolic link to a file on the remote host.
readlink()	Returns the string that is associated with the symbolic link file.
mkdir()	Creates a directory on the remote host and returns the new file handle attributes.
rmdir()	Removes the empty directory from the parent directory.
readdir()	Reads the contents of a directory on the remote host.
statfs()	Returns file system information such as block size and number of free blocks.

Table 7-1: NFS Protocol Procedures

7.2. Remote Procedure Calls

Using Remote Procedure Calls (RPC) a local routine can call and bind to a remote service, have the service perform some function, and read the results. The RPC interface to an application consists of three layers:

RPC Service Library:
This is the top layer and consists of a set of library routines.

RPC Interface: Three system calls are provided as part of this interface. The three calls are:

- registerrpc(): Provides a unique identification number.
- callrpc(): Used by client applications to execute a given remote procedure call.
- svc_run(): A network service makes this call after it registers itself using registerrpc() to inform the RPC dispatcher that it is ready to receive requests.

RPC Lower Layer:
Provides the use of timeouts, appropriate communications transport mechanism and multiple authentication methods.

The BSD 4.2/4.3 implementation allow RPCs to use TCP/IP, UDP/IP and sockets. A *socket* is an endpoint of communication referred to by a descriptor, just like a file or a pipe.

7.3. eXternal Data Representation

eXternal Data Representation (XDR) is a data representation standard, a data description language and a C library package. XDR is used to represent data in a standard, consistent way on the network. XDR thus provides a method of converting a computer architecture's own internal data representation into and out of XDR format. The process of converting an internal representation of data into a XDR form is known as *serialization* or *marshalling* of data. *Filters* are the set of procedures that convert data into and out of XDR encoding.

7.4. NFS Server

An NFS server when responding to a request must commit any modified data to stable storage before returning results. Thus, for UNIX based servers all requests that modify the file system must write all modified data to disk before returning from the call. The server uses the inode number, inode generation number and file system id as the file handle for a file.

7.4.1. How does it work?

The file **/etc/exports** on the NFS server's file system lists those directories that NFS clients are allowed to access and any access restrictions that apply. Typically, at boot time the script **/etc/rc.local** starts up a program called **exportfs**. This program reads the **/etc/exports** file and informs the servers kernel about access restrictions that apply to each exported file system.

7.5. NFS Client

NFS clients can attach an exported remote file system to a directory with the **mount** command. This enables the server to limit access to file systems by checking the client systems' credentials. The client deals with host names only once, at mount time. When a client mounts a file system to a directory it does not make a copy of that directory. It uses a number of RPC calls to transparently access the exported file system. Transparent access to various file systems mounted on a host is provided by a file system interface to the kernel. Each file system interface consists of two sets of operations:

1. Virtual File System (VFS) interface defines the procedures that operate on the file system as a whole.
2. Virtual NODE (VNODE) interface defines the procedures that operate on an individual file within that file system.

Figure 7-2 illustrates the relationship between an NFS server, client and the file system interface.

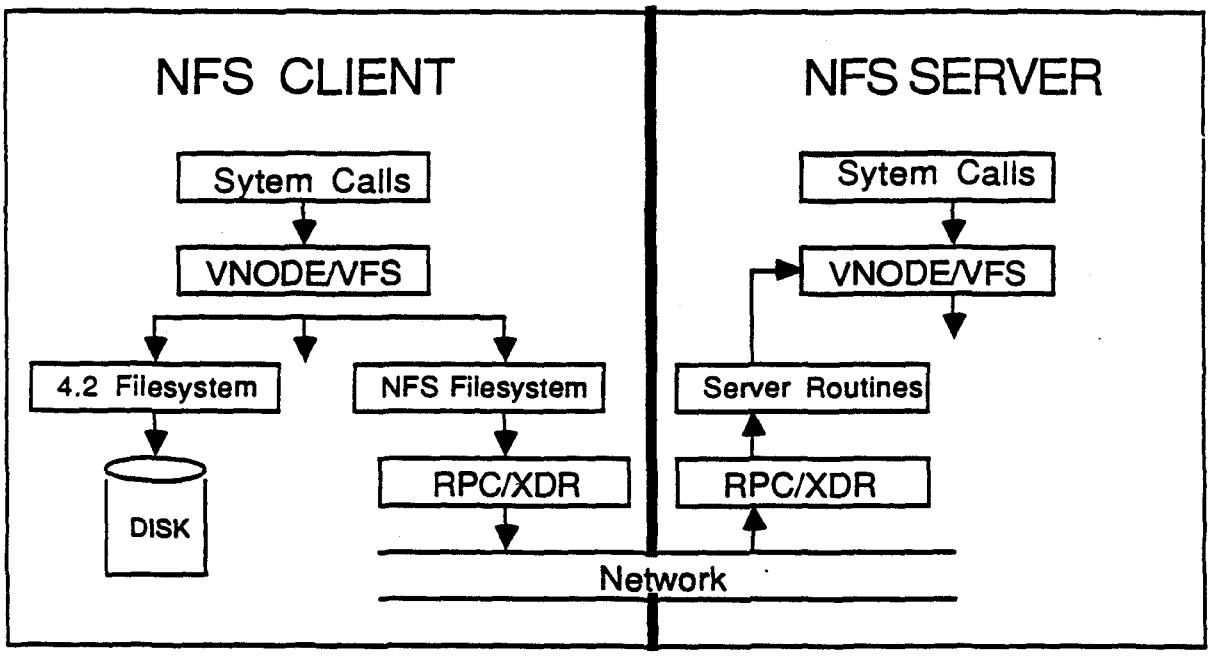


Figure 7-2: NFS Server, Client and the File System Interface

7.5.1. How does it work?

NFS binding is the process whereby a client locates the server that exports the information it needs and then sets up communication between the server and itself. NFS binding occurs during an NFS mount. The file **/etc/fstab** lists all file systems that the NFS client mounts at boot time. A client can also mount a directory during a regular session by executing the **mount** command.

7.6. Network Related Files

The system files described in Table 7-2 are necessary to have a functional NFS environment.

7.7. Network Related Daemons

Daemons are processes that usually start when the system is booted. These processes are typically owned by **root**. The daemons described in Table 7-3 are necessary to have a functional NFS environment.

7.8. NFS Related Commands

There are four main commands that NFS administrators use.

7.8.1. /usr/etc/exportfs

This command is usually run at boot time by the script, **rc.local**. It is used to export or make available for mounting a set of local file systems. **rc.local** reads the /etc/exports file and places a list of file systems to be exported in the /etc/xtab file. The format of the command is:

```
# /usr/etc/exportfs [options]  directory-name
```

exportfs -a indicates export all directories listed in the /etc/exports file. **exportfs -u** indicates *unexport* all directories previously exported.

7.8.2. /usr/etc/mount

The **mount** command is used to attach a remote file system to a local directory. The **mount** command may be used to mount local disk drives as well as NFS file systems. A list of mounted file systems is maintained in /etc/mtab. The format of the command is:

```
# /usr/etc/mount [switches] [-t type] [-o options] [filesystem] [dir]
```

Executing the mount command without any parameters will display a list of mounted file systems. **mount -a** results in an attempt to mount all file systems listed in /etc/fstab. The following example provides information on all mounted file systems on host *cdsgi1*:

Network Files	Description
/etc/fstab	Describes which file systems are available to be remotely mounted. /etc/fstab is used at boot time to make available file systems specified in this file. rc.boot file usually contains **mount** command with the -a option and this causes operating system to mount all devices and files specified in /etc/fstab.
/etc/mtab	This file contains a list of file systems currently mounted on the local system.
/etc/exports	This file contains a list of file systems on the local system that are to be exported to remote systems using NFS. This file determines if the local system is a NFS server.
/etc/xtab	This file contains a list of currently exported file systems for the local host. Information is placed in the /etc/xtab file by the **exportfs** command when it is executed.
/etc/hosts	This file contains a list of host names and their corresponding Internet addresses. The **exportfs** command checks the /etc/hosts or /etc/netgroup file in order to determine the Internet addresses named in the /etc/exports file.
/etc/ethers	This file contains a list of 48-bit Ethernet addresses for hosts on the network.
/etc/netgroups	This file contains a list of network groups. The information in this file is used for remote mount permission checking purposes.
/etc/inetd.conf	This is the configuration file for the Internet server daemon **inetd**. The **inetd** daemon starts other network service daemons on the local system if requested by the client.
/etc/protocols	This file contains a list of valid protocols used on the Internet network. Protocols specified in the /etc/inetd.conf file must be listed in this file. This file may be soon replaced by BIND or X.500.
/etc/services	This file contains a list of daemons available on the Internet or the local network.

Table 7-2: Network Related Files

```
% mount

/dev/root on / type efs (rw,raw=/dev/rroot)
/dev/dsk/dks0d6s7 on /usr2 type efs (rw)
```

Network Related Daemons	Description
/usr/etc/nfsd	The command **/usr/etc/nfsd** starts up a number of UNIX daemons on the NFS server host to service client file system access requests.
/usr/etc/biod	The command **/usr/etc/biod** starts a number of asynchronous Block I/O Daemons (biod) on a NFS client. The **biod** daemons enables a client to make read-ahead and write-behind requests to a remote file system. These requests are buffered to improve NFS performance.
/usr/etc/rpc.mountd	This daemon responds to requests from remote computer systems to mount a local file system. This daemon checks the /etc/xtab file to determine what files are available for mounting; if available, it allows NFS access requests and then provides information on file systems via the **showmount** command.
/usr/etc/inetd	This daemon reads **inetd.conf** on startup. It listens for connection requests for each service specified in /etc/inetd.conf. The daemon is used to start the Internet **inetd** daemon.

Table 7-3: Network Related Daemons

```
/dev/dsk/dks0d4s7 on /usr1 type efs (rw)
/dev/usr on /usr type efs (rw,raw=/dev/rusr)
/debug on /debug type dbg (rw)
fnboot:unix_root1: on /usr/fnboot type nfs (rw,bg)
```

7.8.3. /usr/etc/showmount

The **showmount** command is used to determine which system has mounted a file system from a given host. The format of this command is:

```
% /usr/etc/showmount [-ade] [host]
```

The default for host is the host name for the local system. **showmount -a** displays all remote mount in the form *host:directory*. **showmount -e** displays the list of all exported file systems. The following example provides information on all file systems exported by host *fnboot*:

```
% showmount -e fnboot

export list for fnboot:
/unix_root1        cddg1.fnal.gov cdsun1.fnal.gov cdsgi1.fnal.gov
```

7.8.4. nfsstat

The **nfsstat** command is used to obtain usage statistics for NFS and RPC interfaces to the kernel on the local host. The command provides information on how efficient the local system has been in servicing requests. An example of the output of **nfsstat** is given below:

```
Server rpc:
calls     badcalls   nullrecv   badlen     xdrcall
2170      0          0          0          0

Server nfs:
call      badcalls
2170      0
null      getattr    setattr    root       lookup     readlink   read
0  0%     1050 48%   0  0%      0  0%      414 19%    313 14%    320 14%
wrcache   write      create     remove     rename     link       symlink
0  0%     0  0%      0  0%      0  0%      0  0%      0  0%      0  0%
mkdir     rmdir      readdir    fsstat
0  0%     0  0%      62  2%     11  0%

Client rpc:
calls     badcalls   retrans    badxid     timeout    wait       newcred
0         0          0          0          0          0          0

Client nfs:
calls     badcalls   nclget     nclsleep
0         0          0          0
null      getattr    setattr    root       lookup     readlink   read
0 0%      0 0%       0 0%       0 0%       0 0%       0 0%       0 0%
wrcache   write      create     remove     rename     link       symlink
0 0%      0 0%       0 0%       0 0%       0 0%       0 0%       0 0%
mkdir     rmdir      readdir    fsstat
0 0%      0 0%       0 0%       0 0%
```

7.9. Installing NFS on a VAX/VMS System

This section assumes that you have MultiNet version 2.0 installed on your system. Using an example, the following sections describe how a portion of a VAX/VMS disk is exported to UNIX systems. The server accepts the following formats for mount point names:

- A device name, for example DUA0:
- A device and directory name, for example, DUA0:[UNIX] or SYS$SYSDEVICE:[UNIX]

- A logical name, for example, SYS$SYSDEVICE: OR SYS$SYSTEM:

Let us consider the case where a VAX/VMS host *fnboot* needs to export a logical name **unix$root** to a SUN SPARCstation 1 host *cdsun1*. First, the system manager for *fnboot* executes the following statements to define a NFS mount point. SYSTEM_LOGICALS.COM is the file that is modified with the following expressions:

```
!Define a NFS mount point --

$  DEFINE/SYSTEM/EXEC/TRANS_ATT=(CONCEAL,TERM) UNIX$ROOT -
   'F$LOGICAL("DISK$FNBOOT_WREN2")'[UNIX.]

!Next, create a directory called UNIX.DIR

$ set def fnboot_wren2:[000000]
$ create/dir/owner=system/prot=(s:rwe,o:rwe,g:re,w:re) [.UNIX]
```

Next, the NFS server must be enabled in the MultiNet_Server configuration file. After the MultiNet_Server process has been enabled it automatically starts the process - NFS_SERVER. The following steps describe how the server is enabled:

```
$  multinet configure/server

MultiNet Server Configuration Manager 2.0(14)
[Reading in symbols from SERVER image MULTINET:SERVER.EXE]
[Reading in configuration from MULTINET:SERVICES.MASTER_SERVER]

SERVER-CONFIG> enable nfs
SERVER-CONFIG> enable rpcmount
SERVER-CONFIG> enable rpcportmap
SERVER-CONFIG> restart

Configuration modified, do you want to save it first ? [YES] yes
[Writing configuration to SYS$COMMON:[MULTINET]SERVICES.MASTER_SERVER]
%RUN-S-PROC_ID, identification of created process is 00000312
SERVER-CONFIG>exit
[Configuration not modified, so no update needed]
$
```

Next, to export a file system or a directory do the following:

```
$ multinet configure/nfs
MultiNet NFS Server Configuration Manager 2.0(8)
[Reading in configuration from MULTINET:NFS.CONFIGURATION]
NFS-CONFIG>  add export fnboot_wren2:
[Added new Exported file system "fnboot_wren2:"]
[Current Exported File System set to "fnboot_wren2:"]
NFS-CONFIG>
```

It is highly recommended that the MOUNT-RESTRICTIONS command be used in NFS-CONFIG to provide access to only trusted hosts. By default, the file system is exported to any host that wishes to mount it. In our example, let us restrict access to the following systems: *cdsun1* and *cddec1*.

```
NFS-CONFIG>add mount-restriction   fnboot_wren2:   cdsun1.fnal.gov
NFS-CONFIG>add mount-restriction   fnboot_wren2:   cddec1.fnal.gov
```

After this is done, it is critical that a mapping be defined between VAX/VMS user names and UNIX UIDs/GIDs. MultiNet provides two ways to define mappings:

1. The ADD UID-TRANSLATION command may be used to establish specific mappings between VAX/VMS user names and UIDs/GIDs.
2. The ADD NFS-PASSWD-FILE command may be used to point the NFS server to a /etc/passwd style file. This file establishes a map between VMS user names and UNIX UIDs/GIDs.

The following example illustrates how *cdsun1* has root access to VAX/VMS files owned by **SYSTEM**.

```
NFS-CONFIG>add uid-translation system 0 0
[Added UID Translation "SYSTEM" = 0, 0]
```

The example below provides details on how an /etc/passwd style file called nfs.passwd may be used to establish a mapping netween UNIX UIDs/GIDs and VAX/VMS accounts.

```
NFS-CONFIG>add nfs-passwd-file multinet:nfs.passwd
[Added new NFS Password File "MULTINET:NFS.PASSWD"]
```

Lastly, use the RESTART command to start the NFS Server.

```
NFS-CONFIG> restart
Configuration modified, do you want to save it first ? [YES] yes
Connected to NETCONTROL server on "127.0.0.1"
< TGV.COM Network Control 2.0(3) at  17-Jan-90 12:01AM-CST
< NFS Server Started
NFS-CONFIG> exit
$
```

You may next use the **REPLY/ENABLE=NETWORK** command to check any error messages that may be displayed in the process of the file system being mounted from a UNIX host.

To mount the exported file system on host *cdsun1*, enter the following commands:

```
# mkdir /fnusg
# mount -t nfs -o soft,rw,wsize=2048  fnboot:'unix$root'  /fnusg
```

7.10. Installing NFS on a UNIX System

This section provides information on how to set up an NFS server and a NFS client. As indicated earlier the file /etc/exports lists all file systems that a server exports to its clients. It is through this file that NFS servers can control which systems have access to their file systems. In order to better understand the expressions in the /etc/exports file, let us consider the following example.

```
/usr
#
/export/root/cdsun2 -root=cdsun2,access=cdsun2
/export/swap/cdsun2 -root=cdsun2,access=cdsun2
#
/export/exec/kvm/sun4c
#
/home/people
/export/exec/sun4c
```

The above file (/etc/exports) resides on host *cdsun1*. The following file systems are available to be mounted by any system:

```
/usr
/export/exec/kvm/sun4c
/home/people
/export/exec/sun4c
```

The option **-root=***cdsun2* indicates that **root** access from host *cdsun2* is given to the directory /export/root/cdsun2. In the file **root** access from host *cdsun2* is also given to the directory /export/swap/cdsun2.

The option **-access=***cdsun2* indicates that access for directory /export/root/cdsun2 is granted only to host *cdsun2*. Thus, other systems on the Internet cannot mount host *cdsun1's* /export/root/cdsun2 directory.

Another option that is used quite often is **ro**. This option specifies that clients are limited to read-only access to the specifies directory.

After you have modified the /etc/exports file, you may execute the following command to export all file systems:

```
# /usr/etc/exportfs -a
```

exportfs updates the /etc/xtab file. As mentioned earlier, this file maintains a list of currently exported directories.

On the NFS client, you need to do the following:

- Define mount points in the client's directory structure for the directories specified in /etc/fstab.
- Modify the client's /etc/fstab file to include directories to be mounted.
- Mount/unmount directories as required.

Mount points are basically empty directories that are created by using the **mkdir** command. The /etc/fstab file on NFS client *cdsun2* looks like the following:

```
/dev/sd0a / 4.2 rw,nosuid 1 1
/dev/sd0h /home 4.2 rw 1 3
cdsun1:/export/exec/sun4c /usr nfs rw 0 0
cdsun1:/export/exec/kvm/sun4c /usr/kvm nfs rw 0 0
cdsun1:/export/share /usr/share nfs rw 0 0
```

Entries in the /etc/fstab have the following format:

```
server:server_dir  client_mountpoint  type  options  freq  pass
```

Server is the name of the NFS server exporting the disk. *Server_dir* is the name of the directory being exported by the server. *Client_mountpoint* is the mount point on the client - where it accesses directories mounted from the server. *Type* describes the type of mount. It is typically NFS or 4.2. *Options* that are typically used are **rw** or **ro**. *Freq* refers to the interval in days between dumps of the directory. *Pass* refers to the fsck pass in which the listed file system is checked. If *pass* is zero then the file system is not checked during **fsck**.

The following command mounts everything specified in /etc/fstab:

```
# /usr/etc/mount -a
```

7.11. References and Further Reading

1. *ULTRIX-32: Guide to Network File System.* DIGITAL, 1988. AA-ME99A-TE.

2. Kochan, S.G. and Wood, P.H.. *UNIX Networking.* Hayden Books, Indianapolis, IN, 1989.

3. *MultiNet NFS SYSTEM ADMINISTRATORS' REFERENCE.* TGV, 1989.

Chapter 8

Understanding DNA and DECnet

An architecture, like a building architecture, is a plan. If each of the components of an architecture is built according to the plan, they will work together. Simply put, a computer architecture is a way of making many individual components work together as one system.

Network architectures are just one of many different kinds of architectures. The network architecture begins with a complete computer system and says how these computers will communicate. A distributed network means that different computers on the network specialize in different tasks. One computer, known as a terminal server, might specialize in communicating with terminals. Another computer, known as a print server, might specialize in laser printing services. Finally, a general-purpose computer like a VAX would specialize in computation. All of these computers work together in a transparent fashion.

8.1. What is a DECnet Network?

DECnet is the collective name for the family of communications products that allow DIGITAL operating systems to participate in a network. As a part of a network, a VMS system can communicate with other VMS systems running on the full range of VAX processors, as well as with a wide range of non-VMS systems that use DECnet software.

All systems connected to a DECnet network are peers or equals. Systems can communicate with each other without having to go through a central or master system. Any system in the network can communicate with any other system in the network, not merely with those systems to which it is directly attached. Network users can gain access to software facilities that do not exist on their particular system, and can communicate freely over the whole network.

8.2. How Does a DECnet Network Work?

A DECnet network consists of two or more computing systems linked for the purpose of exchanging information and sharing resources. Network activity involves the flow of information between the systems. Data originated on one system is routed through the network until it reaches its destination on another system.

Each system on the network is called a *node*. Every node has a unique name and address. Nodes in the network are connected by *lines* over which *circuits* operate. See Figure 8-1. A line is a physical path over which data passes from one node to another in the network. A circuit can be thought of as a higher-

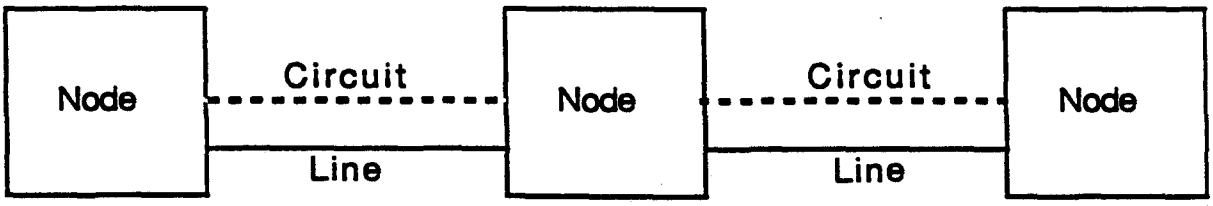


Figure 8-1: Network Nodes, Circuits, and Lines

level logical connection that operates over the physical connection. The circuit is the communications data path that carries information from one node to another. All input and output activity between nodes occurs over circuits. Multiple users can use each circuit. DECnet permits computer processes running on the same or different nodes to communicate with each other over *logical links*. A logical link connects two processes and carries a stream of two-way communications traffic between the processes over one or more circuits.

8.2.1. How Does the Network Route Messages?

In a DECnet network, the process of directing a data message from a source node to a destination node is called *routing.* The route the data travels over the circuits in the network is called the *path.*

Messages can be exchanged between any two nodes in the DECnet network, even if they are not directly connected to each other. In order for nodes that are not directly connected to be able to communicate, an intervening node along the data path must forward the data received from the source to the destination. Intervening nodes that receive data and forward it to another node are known as routing nodes (or *routers*). Nodes that cannot forward data are called *end nodes.* Both routers and end nodes can send messages to and receive messages from other nodes, but only the router can forward messages on behalf of another node. A router can have more than one active circuit connecting it to the network; an end node can only have one.

A router maintains an information database about the availability of paths to the destination node and keeps it up-to-date by regularly exchanging routing information with other routers. The routing information includes the *cost* and the number of *hops* involved in sending data down a path to a destination node. The circuit cost is a number that the system manager assigns to a circuit between two nodes; the path cost is the sum of the circuit costs along the path to a given node. A hop is the distance between two directly connected nodes; the path length is the number of hops along the path between two nodes.

The router uses current information from it is database to choose a data path through the network. The router determines the path to the destination based on the least cost. By changing the cost of a circuit, the manager of a network node can affect the flow of data through the network.

8.2.2. How Large Can the Network Be?

A DECnet network can vary in size from a small to a very large network. A maximum of 1023 nodes is possible in an undivided DECnet network; an optimum number is approximately 300-500 nodes, depending on the network topology.

Very large DECnet networks can be divided into multiple *areas*: up to 63 areas, each containing a maximum of 1023 nodes. In a multiple-area network, the network manager groups nodes into separate areas, with each area functioning as a subnetwork. Nodes in any area can communicate with nodes in other areas. DECnet supports routing within each area and a second, higher level of routing that links the areas, resulting in less routing traffic through the network. Nodes that perform routing within a single area are referred to as *level 1 routers*; nodes that perform routing between areas as well as within their own area are called *level 2 routers.*

8.3. Purpose

DECnet software design is based on the DIGITAL Network Architecture (DNA), which follows industry standards. (DNA) is a model of the structure and functions upon which DECnet implementations are based. The structured design permits a DECnet network to be extended easily and to incorporate new developments in data communications. It defines the means by which computer systems can communicate with each other to form a distributed processing system. It specifies:

- Communication protocols used to exchange information.
- Internal interfaces between protocols.
- Policies and mechanisms by which systems adapt to changing loads and configurations.
- Means by which the network can be managed, both locally and remotely.

The increasing importance of networking means that computer networks are continually getting larger. Where a few years ago networks of even a dozen systems were rare, today networks of hundreds of systems are commonplace and several much larger networks, up to tens of thousands of systems, are in use. Furthermore, this increase in the size and number of networks means that they can no longer be operated exclusively by specialists as they have in the past. DNA includes many capabilities to simplify the design, construction, and operation of large networks.

DNA specifications govern the interrelationship of the components that make up the DECnet software. The specific functional boundaries between DECnet software components residing at each node are structured as a hierarchical set of layers. Each DNA layer is a client of the next lower layer and does not function independently.

DNA specifies the functional layers in which DECnet software is arranged on each node, and the communications *protocols* through which the corresponding layers at different nodes communicate with each other. Each protocol is a set of messages with specific formats and the rules for exchanging the messages. Protocols govern the operation of a communications link.

DNA Phase IV

8.4. Phase IV Overview

DNA is able to use three different subnetwork technologies. The Digital Data Communications Message Protocol (DDMCP) is an example of a traditional data link protocol. DDCMP is used to form a point-to-point link between two computers. Another subnetwork technology is Ethernet. Ethernet consists of data and physical link protocols. To the DNA routing layer, the Ethernet topology looks like a single wire with many nodes on it. Ethernet is used by many other architectures. An important characteristic of Ethernet is that many users can coexist on the medium. The DNA routing layer is one of those users, but TCP/IP or LAT or other protocols will also be present on the same data link. X.25 is a third subnetwork technology supported by DNA. Again DNA may share the X.25 facilities with other users.
The Computer Interconnect (CI) bus is used to connect large VAX systems and a special-purpose computer called the Hierarchical Storage Controller (HSC). The CI bus operates at 70 Mbps. The HSC is used to connect high performance disk drives to a very intelligent disk controller.

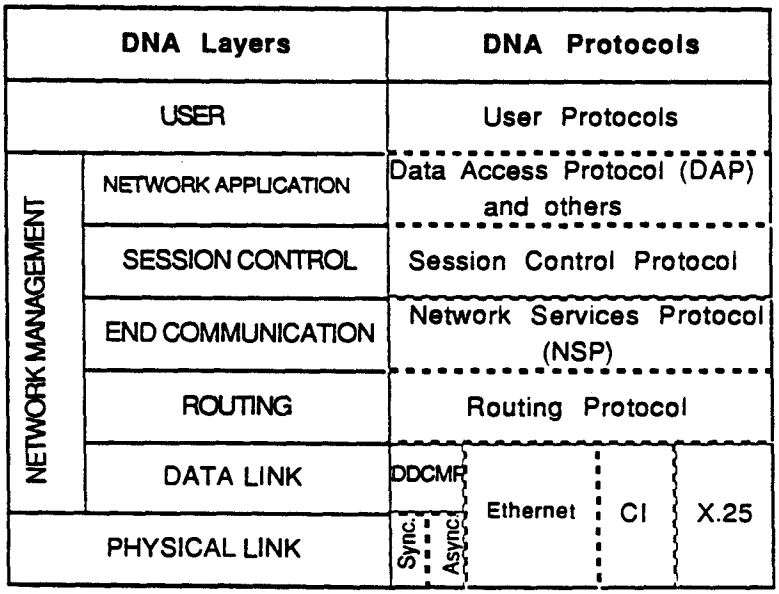


Figure 8-2: DNA Phase IV

The routing layer of DNA, which corresponds to the OSI network layer, is responsible for taking packets of data and deciding which data link the packet should travel on. A DNA network can have a highly complex topology with up

to 63,000 computers. The responsibility of the routing layer is to know what that topology is and to stay abreast of any changes in the topology caused by node or line failures.

The next two layers of DNA are the end-to-end communications and session layers. The end-to-end communications layer is identical to the OSI transport layer. The session layer provides several important functions.

1. It validates incoming connections requests.
2. It activates an interactive log-in, for example, the session layer would activate the virtual terminal service.
3. It provides a node name to address mapping service. Nodes, to users, have names of more than six characters like "MYVAX" or "GRAPHICS" or "DATABASE". The session layer turns those logical names into DECnet addresses.

There are two categories of services in DNA : network management services and network applications. Network management services are a set of programs and protocols used to manage the network. These services use the facilities of the network to exchange data on the current state of nodes, lines, and other network components. A variety of different user interfaces are present that can show the status of the network or historical utilization or indicators of activity. Network applications are services that the user will see. An example of an application is the virtual terminal service. This service allows any user on any node of the network to log into any other node on the network. The Data Access Protocols (DAP) are a set of services that allow a user to access any file on the network, subject to security restrictions. Other data access protocols, such as Distributed File Service (DFS), allow data on the network to appear as though it were local. Files anywhere on the network appear as though they were on the local disk drive.

8.5. Phase IV Physical Layer

Protocols for the physical layer are rudimentary. No special physical layer standards have been developed for DNA. Instead, it relies on industry standards for the physical layer, thereby ensuring that DECnet products can operate over available technologies and physical networks.

For local area networks, traditional coaxial cable, known as baseband, can be up to 400m long with 100 nodes connected to it. Another type of coaxial cable, known as ThinWire, allows up to 30 nodes to be connected to a cable up to 185m long. Broadband is based on the Community Antenna TV (CATV) technology used in cable TV systems. CATV cable is meant to be hung on telephone cables and it highly resistant to weather. It has a bandwidth of 300-400 MHz meaning it can serve the purposes of a number of communications

technologies, including Ethernet, voice, and video. Broadband is often used as a backbone between buildings in a campus-like environment. Segments can be 10-12 miles long.

Fiber is used in a similar role to broadband, as a backbone cabling system. It can also be used as a wide area backbone. Fiber is most often used to connect bridges or repeaters together. It can be used, as stated above, as a backbone. Also, it can actually be used to connect hosts together.

Twisted pair is an attractive implementation because it is often already in place for the phone system. RS232-D and RS-423 physical interface standards are supported by DECnet Phase IV.

8.6. Phase IV Data Link Layer

Three subnetwork technologies are supported by the DNA data link layer: DDCMP, Ethernet and X.25. Also included in the DNA data link layer is the CI bus.

8.6.1. DDCMP

DDCMP is an example of a data link protocol that manages the use of an underlying physical medium. The data link layer thus provides a point to point link between two nodes on the network. It is supported on synchronous and asynchronous communications devices. DDCMP connections can be point-to-point or multipoint configurations. Point-to-point connections are either synchronous or asynchronous. The two types of asynchronous connections are static (permanent) and dynamic (switched temporary). Multipoint connections are always synchronous.

There are three major components to a DDCMP module: framing, link management and message exchange.

Framing

Framing is the process of taking data and preparing it for transmission. The framing component of a DDCMP module monitors the media and locates the beginning and end of a message. This occurs at three levels. First, the module must locate each bit on the network. This is known as bit synchronization. Next, the process of byte synchronization groups data into 8-bit quantities. The final level is message synchronization, which groups bytes into frames.

Byte synchronization is done through the use of a special 8-bit pattern called the synchronization byte. After that has been received, the DDCMP module begins counting bits. Every 8 bits is considered a byte. Usually, a sync byte is sent for each message.

Once byte synchronization is achieved, the DDCMP module searches for one of three message start bytes:

- numbered data messages
- control messages
- maintenance messages

The normal type of message is a numbered data message that carries user data over the link. The first piece of data following the start of header (SOH), or message start byte, is a count field that tells how long the message is. The data is counted, therefore any bit pattern can be included in the user data. If the user data happens to have a piece of data equivalent to a start of message or byte synchronization pattern in it, this will not effect the transmission of the data. Only after the specified number of bytes has been received does the DDCMP module begin looking for unique patterns such as the sync byte. The frame header also includes a series of link control flags. These flags signal the receiving end when another message will immediately follow. The frame header also contains a sequence number. The receiving end must acknowledge each packet by number. Two techniques for increasing acknowledgment efficiency are pipelining and piggybacking. Pipelining means that several packets are sent and the acknowledgment signifies receipt of that packet and all lower numbered packets. Piggybacking refers to the inclusion of a response number in the message header of a user data packet being sent by the receiving station. If no data is to be sent, then the acknowledgment cannot piggyback.

Control messages are unnumbered messages used to transmit channel control information in a multipoint link. They also transmit status information between two protocol modules. An ACK is a control message. A negative acknowledgment has the same format but also includes a reason indicator.

A maintenance message is used in offline mode to test a link. It is similar to the data message but does not include any retransmit, error recovery, or other mechanisms.

Link Management

Link management manages half duplex or multipoint links. Link management is the second component of a DDCMP module. It is only necessary where the flow of data must be controlled, as in the case of either a multipoint or a half-duplex line. A full-duplex line does not need this function.

A half duplex line is under the control of the data sender. The receiver may not send data until it has received permission. On a half-duplex line, the sending node sends a selection flag when it has completed sending data. This instructs the receiving end to enter the transmit mode.

Multipoint links are a special case in DDCMP. One station, the control station, is the master of the line. A selection flag is used to assign temporary control of the line to a tributary that has data to send. The control station polls the tributaries periodically to determine if one desires temporary control of the line.

Message exchange

Message exchange is the actual transfer of data. It provides sequential, error-free data transfer. Each message sent by the message exchange component has a message number. Each message must have a positive acknowledgment or ACK. If a negative acknowledgment is received, the message is sent again. Each message also has a timer associated with it. If the timer expires, this is accepted as a negative acknowledgment of the message. DDCMP uses the CRC-16 cyclic redundancy check for error detection on protocol headers and user data. The basic message exchange mechanism uses data message sequence numbers, a positive acknowledgment control message (ACK) and a timer.

When a timer expires it is possible the ACK was lost in transmission and the data actually arrived. Rather than send the entire data packet again, a Reply to Message Number (REP) packet is sent. This requires an acknowledgment of the REP message, which implies an ACK for the original data message. It also forces the timer to be reset and the two nodes synchronize their message numbering. If a timer expires several times, the DDCMP module notifies the user (typically a DNA routing module) of the line failure.

Multipoint links are supported in DDCMP, but because it uses a polling mechanism, it is not optimal for large numbers of nodes sharing the same physical medium. In these cases, Ethernet is a more suitable data link. DDCMP does not support any kind of broadcast or multicasting facility.

8.6.2. Ethernet

Ethernet provides an alternative data link mechanism to connect nodes together. It supports the high-speed networking capabilities of a local area network. The user of the Ethernet service is the DNA routing layer. Phase IV supports the Ethernet standard as opposed to IEEE 802.3. See the *Understanding Ethernet* chapter of this document for an in depth description of Ethernet.

8.6.3. X.25

X.25 is a packet-switched data communications architecture. Support for the X.25 protocols is provided either with a dedicated X.25 gateway or by using the services of a general-purpose VAX. X.25 can be used three different ways in the DEC environment:

- As a DNA service provider. X.25 functions as a subnetwork accessible to the DNA routing layer. This allows X.25 to provide a link between two points, just like DDCMP or Ethernet.
- To allow DNA users to access remote hosts. The second way to use X.25 in the DNA environment is as a tool for establishing a connections to a heterogeneous environment. The X.25 connection is not

used to carry DNA traffic, but rather as a way of connecting a local user on the DECnet to some non-DNA host.

- To allow remote terminals to access DNA hosts.

See the *X.25* chapter of this document for an in depth description of X.25.

8.6.4. CI Bus

The CI bus is used by another DEC architecture, the System Communications Architecture (SCA) also known as VAX Clusters. VAX clusters allow several nodes to be joined into a closely coupled network. The System Communication Service is somewhat equivalent to the DNA routing layer - both offer internode communication services. The CI bus operates at 70 Mbps and as stated above, is used to connect large VAX systems and a special-purpose computer called the HSC. The HSC is used to connect high performance disk drives to a very intelligent disk controller. The main purpose of the cluster is to allow multiple computers to share disk drives at high speeds. The CI bus cluster provides memory-to-memory throughput rates between a VAX and an HSC of 2 Mbps.

A Local Area VAX Cluster (LAVC) provides the same functionality, but over Ethernet. HSC controllers cannot be connected to the Ethernet, so a general purpose VAX must provide disk services.

8.7. Phase IV Routing Layer

The routing layer is responsible for taking a packet from data link and deciding which data link to send the packet back out on. The routing layer thus takes a series of data links and forwards a packet from the source to the destination. In this case, the destination could be separated from the source by many hops (individual instances of a data link).

The routing layer thus masks the topology of the network from the users of the routing service. Higher layers of the network are able to assume that they can talk directly to their destination. The data destined for data transmission has been packetized two times. The routing layer receives data and puts a routing layer header on it. All of this information is considered to be "user data" for the data link layer which puts its own packet header on the information.

Data links between individual nodes on a network are subject to failure, as are individual systems in the network. This is a fairly common occurrence. Therefore, the routing layer has to be able to adapt to changes in the network topology. If one line goes down in the network, the routing layer is able to take an individual packet and send it over an alternative route. Only if no routes are available is the user of the routing service notified of a network error. Otherwise, the dynamic topology of the network is transparent to the upper layers.

8.7.1. Routing Algorithm

An important concept in DNA Phase IV routing is that the routing layer provides adaptive routing only for topographical failures. A particular path is designated as the best path to a particular end destination. Only if that line goes down is an alternative route found. To illustrate this concept, imagine that there are two lines to a particular node. A 56 kbps line is considered to be the best path. There is also an alternate path consisting of a 9600 bps line. Even if the 56 kbps line is nearing saturation, all traffic is pumped down that path and the 9600 bps line remains idle with respect to this particular source and destination combination. If there are two lines that have equal costs, as determined by the routing decision algorithm, packets will be distributed between the two lines in a round-robin fashion. This still does not account for actual traffic on the two lines - it is not adaptive routing based on actual loads on the system.

8.7.2. Functions

The routing module has four major functions:

- The primary function is routing.
- The initialization function sets up a data link layer and connects to adjacent nodes.
- Congestion control is a function that prevents a node from flooding the network with packets.
- Packet lifetime control destroys packets that have already visited too many nodes.

There are several things that the routing layer does not accomplish. The DNA routing layer does not provide for different classes of traffic. It does not have any capability to distinguish among different priorities of delivery. The routing layer also does not react to the amount of traffic on a line.

Routing Decisions

The actual routing decision is the most complicated component of the routing layer. Four separate processes are involved:

- the decision process,
- topology updates,
- forwarding of packets, and
- receiving packets

The decision on which path to pick in a network is a function of the cost of different routing paths. Each link between two nodes is considered to be one hop. Every hop on the network is assigned a cost. Each time a packet is received

from an adjacent node, the receiving node must decide how to forward the packet. Based on the least cost path, the node is able to decide which adjacent node should next receive the packet. Note that each node along the delivery path re-executes the decision process.

A routing node continually monitors the circuits that are directly attached to it. If the node detects a circuit failure, it must notify other routing nodes on the network of that failure so they may update their routing databases. Notification of other nodes is done through a routing control message. This message contains path cost and path length for all destinations. The Phase IV routing tables consist of a list of nodes that are reachable and the intermediate node to send a packet to, for a particular path to the end node. Each path has a cost associated with it. Although the calculation of the number of hops and costs is based on the data links, no information about this is contained in the routing table. The routing table is thus a summary table in Phase IV.

In Phase IV routing control messages, each node transmits it's summary table to the next. If one router notifies the next router that it can reach node A with three hops, the next router assumes that it can reach node A with a total of four hops. There are alternative paths to each node and routers summarize the routing information. Therefore, there may be several interpretations of how far away a particular node is. It thus takes several messages for the network to stabilize after a configuration change. With a large area and many *level 1 routers*, it is possible for a significant portion of the network bandwidth to be used on routing control messages until the situation stabilizes. Level 1 routers provide functionality within an area. To prevent routing control messages from flooding the network, the rate control frequency timer sets a minimum time period between routing messages. Usually it is set at 1 second.

On an Ethernet, routing control messages are sent via a multicast address. All routers on the Ethernet enable receipt of this multicast address as part of the data link initialization process. Special routing algorithms are required on an Ethernet because of the broadcast nature of the media. There may be several routers on an Ethernet. Each of these nodes sends routing control messages to each other. One node is known as the *designated router*. This node periodically sends messages to all end nodes on the network informing them of its address.

Areas

The process of continually updating dynamic routing tables becomes increasingly difficult as the number of nodes increases. To provide large networks, DECnet segments nodes into *areas*. An area has up to 1024 nodes and provides full adaptive routing algorithms. 63 areas can be connected together to form a multiarea DECnet. Routers that provide functionality within an area are known as level 1 routers. Routing between areas is provided by *level 2 routers*. These levels provide a hierarchical routing scheme. One set of routing decisions is made for each level. Even a network of 63,000 nodes has proved to be a limitation. Phase IV address space is only 20 bits.

Congestion Control

Congestion control is an important part of the routing process. The routing layer will arbitrarily destroy packets under certain types of congestion leaving it up to the end-to-end communications layer to resend it. There are several kinds of congestion control used in a DNA network.

- There is a limit on the number of packets that can be queued at any one time for transmission. The limit is a function of the number of routing layer buffers available and the number of active output circuits. All packets above this limit are discarded.
- The routing module is able to distinguish between local and rout-through packets. The originating packet limiter ensures that a certain percentage of routing resources is always available for route-through traffic. Under certain conditions, route-through packets are accepted while local packets are rejected because the network already has a substantial investment in the route-through packet.
- Any packets intended for an adjacent node that has failed are flushed. The sending end communications layer will have to re-transmit these packets. A single routing control message is sent network wide to notify all nodes of the failure.

8.8. Phase IV End-to-End Communications

The end-to-end communications layer corresponds to the OSI/RM transport layer and uses a set of protocols called the Network Services Protocol (NSP). The function of NSP is to form a series of logical links between users. NSP thus serves as a form of multiplexer which takes many different users and delivers a single stream of data to the network services. Each packet of data is delivered by the network services and then demultiplexed at the destination.

8.8.1. Functions of NSP

The basic NSP concept is a logical link that connects two session control modules. There are four major NSP functions:

- **Establishing and destroying logical links.** When a session layer module requests a logical link, it submits a **CONNECT-XMIT** message to the NSP module. The NSP module submits it to the routing layer, which delivers it to the target NSP module. The target NSP module sends the source NSP module an acknowledgment that it has received the request. It then notifies the session control module, which can **ACCEPT** or **REJECT** the request. When the target session control module accepts a session request, it notifies the target NSP module, which sends back a connect confirm message. Sometimes, a

connect request is received on a node that has no resources available for new logical links. The NSP module is able to reject this request without notifying the session control module of the incoming request.

- **Error control.** There are two data subchannels. The normal data subchannel is used for data passed in from higher-level modules. The other-data subchannel is used for interrupts and other out-of-band signalling. Within each subchannel, messages are numbered sequentially. Each message sent must be acknowledged or the sending NSP module will re-transmit the message. Pipelining allows the receiving NSP to acknowledge several messages by acknowledging receipt of the highest-numbered message. The receiving NSP module can only wait for a period of time that is less that the sending node's timeout factor.

- **Flow control.** Three options are no flow control, segment flow control, and session control message flow control. Segment flow control is accomplished by sending a request count parameter. The sending NSP may only have that number of messages outstanding. It looks at the highest-numbered message segment that has been acknowledged, adds the request count parameter, and then sends message segments until it reaches that sequence number. Session control flow control operates the same way, but instead of using individual message segments, the request count parameter refers to the entire message. The NSP module thus looks for the highest acknowledged end-of-message segment and adds the request count parameter to it.

  Each side of an NSP logical link may request different flow control mechanisms. A large VAX connected to a PC might have a session. The VAX, as a data receiver with a large buffer space, might request either no flow control or segment flow control with a large request count parameter. The PC would have very few restraints on the amount of data it can send. On the other hand, the PC as a data-receiving NSP module would probably request a fairly small request count parameter.

  In addition to the normal flow control, either end of a session may also use an On/Off control mechanism. The data-receiving module requests that no further messages be sent until an ON message is transmitted. The other data subchannel uses a message-based flow control mechanism. When a logical link is established, there is an implicit request count parameter of one. This means that an interrupt message must be acknowledged before a second one is sent.

- **Segmentation and reassembly of messages.** This is one of the prime functions of the NSP module. A single segment is limited by the size of data that the routing layer can accept. NSP takes data from a session control buffer, breaks it into segments, and submits each individual segment to the routing layer.

8.8.2. NSP Components

An NSP process has 3 types of components: databases, buffer pools, and modules. Figure 8-3 illustrates the structure of an NSP process.

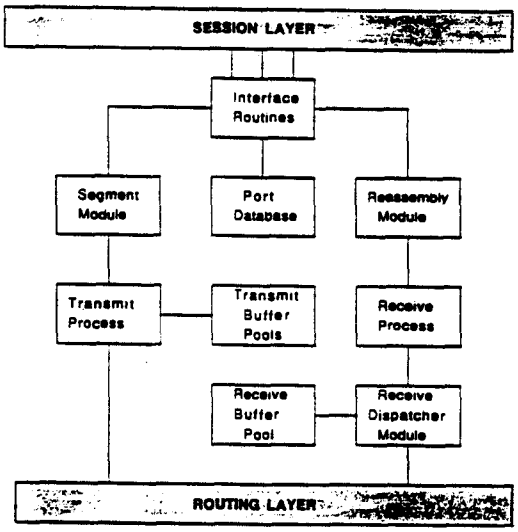


Figure 8-3: NSP Modules

Two of the databases are used internally by NSP to manage itself. The NSP internal database contains information and parameters used in the internal management of the NSP module. The *maximum number of logical links* allowed on this node is an example of a parameter. The reserved port database contains a list of resources that NSP modules use for exchanging control messages that are not mapped into any session control port. A port refers to a particular registered user of a service.

The other two databases are interfaces to other layers. The session control port database is an interface to the session control layer. When the NSP module sets up a logical link, it allocates one of the available ports to the session control layer. When the logical link is destroyed, the NSP module frees the port and returns it to the database. The node database contains information on each of the nodes with which a logical link is established.

Buffers are how data is exchanged with other modules in DNA. A receive buffer pool contains data received from the routing layer. An event buffer pool is used to queue events that are then put into an event queue for processing by the network management layer.

Several types of modules operate under the larger NSP module. Interface routines intercept all calls from session control and provide a unified calling environment. If a message is to be transmitted, it is sent to a segmentation module, which breaks the message up into segments of appropriate size. The transmit buffer pool is used as a queue to the routing layer. The transmit process polls the routing layer to determine when transmission of certain buffers has been com-

pleted. The receive dispatcher polls the routing layer for received messages and then dispatches them to the appropriate receive processes. Each logical link has its own processes, which manage the logical link state. The receive process sends data up to be reassembled, which in turn is sent up to the session control layer.

8.9. Phase IV Session Control Layer

The session layer forms a bridge to the services of DNA. While NSP is responsible for delivering data to end user services, it is the responsibility of the session layer to determine if that end user exists and to validate access in the context of the security domain of the operating system. The session control layer is also responsible for mapping a logical DECnet node name composed of an alphanumeric string into a DECnet address consisting of an area and node designation.

8.9.1. Function

When a session control module receives a logical link request from an end user on the local node, it first identifies the destination address. A connect request is then formed and passed to the destination session control module. This information might contain a user name and password. The session control module passes the connect request to the end-to-end communications layer and starts a timer. Upon receiving an incoming request, the destination network end-to-end communications module notifies the session control module of the incoming logical link address and passes a buffer containing the connect data. The session control module parses the connect data and validates the access control information. It then identifies, creates, or activates the destination end user and passes connection information to the end user processes, attempting to parse the source address into a logical node name. If the node name is not contained in the mapping database, the end user layer is notified that an unknown node is communicating. Once a session has been established, the two session control modules act as a conduit for data for the upper-layer services.

8.9.2. Session Control Databases

A **node name mapping table** translates logical addresses into the 20-bit DECnet address. One of the problems in a large Phase IV network is updating the session control database on all nodes of the network.

The **state database** is the second database kept by the session control layer. There are four different modes of operation. When the module is off, there are no logical links operational. In SHUT state, the module keeps existing links in

operation, but will not accept new requests. RESTRICTED is like the SHUT state but does accept new logical links from users with sufficient levels of privilege. On is the state for normal operation. Default connection timers can be contained in the state database. After passing a request to an upper-layer process (under no time restraint to accept or reject the request), the session layer sets a timer. Upon expiration of the timer, the session control layer can assume the request has been rejected.

8.9.3. DNA Objects

An object indicates the type of service required by the destination session control module. A connect request from an end user process usually contains an object. An object of type 0 is for nonregistered object types, which must be further specified by the requesting end user. In VMS, for example, the user would specify the destination task as:

```
NODE ::"task=taskname.com"
```

In this case, the destination VMS would determine if the requesting user has a proxy log-in or would look in a default area to see if TASKNAME.COM exists. If the end user in VMS had wished to specify an account to run the task in, the syntax would have been:

```
NODE"username password"::"task=taskname.com"
```

A list of objects available can be found using the NCP command SHOW KNOWN OBJECTS. More informtion on NCP can be found in the *Using DECnet: NCP* chapter of this document.

```
NCP> SHOW KNOWN OBJECTS SUMMARY

Known Object Volatile Summary as of 25-MAY-1990 09:00:32

   Object   Number  File/PID                  User Id     Password

TASK            0
FAL            17   FAL.EXE                   DECNET
HLD            18
NML            19   NML.EXE                   NML$SERVER
REMACP         23   2020021B
MIRROR         25                             MIRRO$SERVER
EVL            26   20200219
MAIL           27   MAIL_SERVER.EXE           MAIL$SERVER
PHONE          29   PHONE.EXE                 PHONE$SERVER
DDMF           30   DDMF.COM                  DECNET
NOTES          33   NOTES$SERVER.EXE
                    202024B7                  NOTES$SERVER
RDBSERVER      35   RDBSERVER.EXE             RDB$REMOTE
OBJ_36         36   2020021A
```

```
CTERM          42  2020021B
VPM            51  VPM.EXE
                   20200220                          VPM$SERVER
OBJ_54        -54  2020021A
SPM$REMOTE     60  SYS$SYSTEM:SPM$COLLECT.EXE SPM
DTR            63
DQS            66  DQS$SERVER.EXE                    DQS$SERVER
OBJ_112       112  20200284
FINGER        117  FINGER$ROOT:[EXE]FINGERNET DECNET
NEEDVSN       118  ACPRUN_ROOT:[AREA]NEEDVSN.
ACP_OPER      119  ACPRUN_ROOT:[AREA]ACP_OPER
ACPVSN        120  ACPRUN_ROOT:[AREA]ACPVSN
ACP9VSN       121  ACPRUN_ROOT:[AREA]ACP9VSN.
SEND          199  JAN_SYS:SEND.EXE                  DECNET
REMCOM        200  DISTRIBUTE_HOST$ROOT:[COM]        NETUSR
DO_NEWS       209  DO$DO_NEWS:NET_NEWS.COM           DONEWS$SRVR
```

8.10. Phase IV Network Management and Applications

Network management occurs at all levels in the DNA architecture except the physical layer. The Network Control Program (NCP) is a DECnet utility that the network managers use to control various aspects of the network. The *NCP* chapter of this document explores in further detail what NCP is and what the general user can manipulate.

Network applications are at the highest level. They use the underlying structure and protocols of DNA to accomplish their specific functions. The applications are the user interface to the network. The following section lists some (certainly not all) existing applications available.

8.10.1. Some Existing Applications

The virtual terminal service (VT) is one of two ways to allow a remote terminal to access a host system. The Local Area Transport (LAT) protocols are an alternative service that provides similar functionality. Both services allow a host to treat all terminals, remote or local, in the same way. The LAT protocols are a non-DECnet set of protocols that interface directly with the Ethernet. LAT is more efficient that the DECnet virtual terminal service but is limited only to use on Ethernet devices. The DECnet VT service, on the other hand, is able to operate over any DECnet configuration. The virtual terminal service is usually referred to as CTERM.

DAP is a language used to access data across the network. DAP allows users to get files from remote input and store then on a remote output device. In Phase V, DAP is being replaced by DFS and other remote data access mechanisms such as the LAVC.

Part IV

PC NETWORKS AND PROTOCOLS

This Page Intentionally Left Blank

Chapter 15

Understanding AppleTalk

AppleTalk is a network system for Macintosh computers. It provides access to print and file servers, electronic mail applications and other network services. The software modules that control communication are transparent, since the user does not need to be aware of them to use the network. AppleTalk networking capabilities have been built into the Macintosh computer since its release in 1984.

The AppleTalk network system consists of software - which runs in each device connected to the system and observes AppleTalk protocols - and network hardware. Appletalk is flexible and easy to extend. Devices on an AppleTalk network system participate as peers with all the other devices. AppleTalk protocols support a range of network types, and are built into all Macintosh computers.

Each of the networking and communications products offered by Apple Computer complements the unique user interface of the Macintosh personal computer as well as its processing power. No matter what kind of network the Macintosh connects to or which computer it communicates with, the user still enjoys the same intuitive, direct methods of accessing and manipulating information.

15.1. The Basic System

Every Macintosh computer is equipped with a built-in LocalTalk network connection, which provides a low-cost "plug and play" entry point to the AppleTalk network system. Any Macintosh can be connected to any other Macintosh or to a LaserWriter printer using only the built in capabilities as in Figure 15-1. This is the basic AppleTalk network system.

AppleTalk enables communication between network devices that may be a mixture of Apple and non-Apple products. A Macintosh can communicate with any computer supporting the AppleTalk protocols. Here at Fermilab, a number of AppleTalk networks are installed. These networks are basically used for the sharing of peripherals.

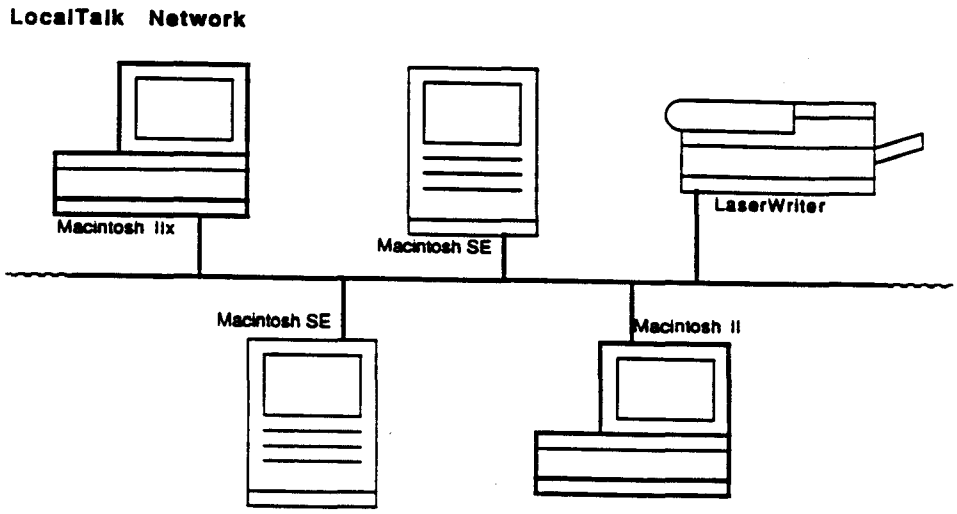


Figure 15-1: LocalTalk Network

15.2. Extending Appletalk with Apple Products

The AppleTalk network system allows Macintosh computer users to also communicate over other local and remote networking environments.

There are 3 AppleTalk protocols: *LocalTalk*, *EtherTalk*, and *TokenTalk*.

LocalTalk: is the basis of the AppleTalk network system and is built into every Macintosh computer. LocalTalk runs at 230 kilobits per second.

EtherTalk: allows a Macintosh computer running the Macintosh operating system to access AppleTalk services on an Ethernet network. EtherTalk software is used with the EtherTalk NB card. Macintosh users can then make use the AppleTalk network protocols over high-speed Ethernet (10 megabits per second). Figure 15-2 shows AppleTalk connected via Ethernet.

TokenTalk: provides connectivity to a token ring network in much the same way Ethertalk works for Ethernet networks. It also requires an interface card and it runs at 4 megabits per second.

Each of these three protocols perform OSI reference model physical and data link layer functions.

These different network types can be interconnected using the *AppleTalk Internet Router* so that they function as a single network. The AppleTalk Internet Router is software installed in a non-dedicated Macintosh computer that links networks together and maintains addressing information for each network. The word

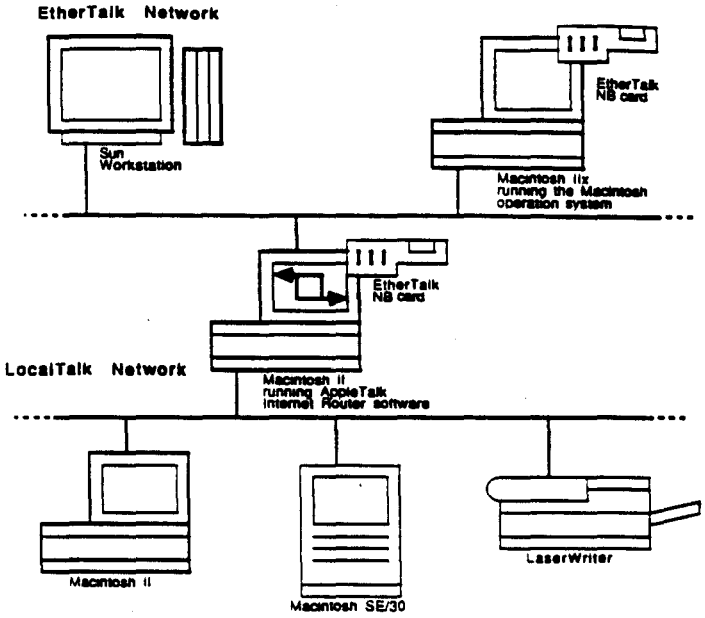


Figure 15-2: AppleTalk network connected via Ethernet

"internet" as used here refers to an AppleTalk internet. For instance, a LocalTalk and an EtherTalk network are both forms of an AppleTalk network. Connecting them with an AppleTalk Internet Router makes them an internet, but not in the sense of providing TCP/IP connectivity. Figure 15-3 shows the router and its function.

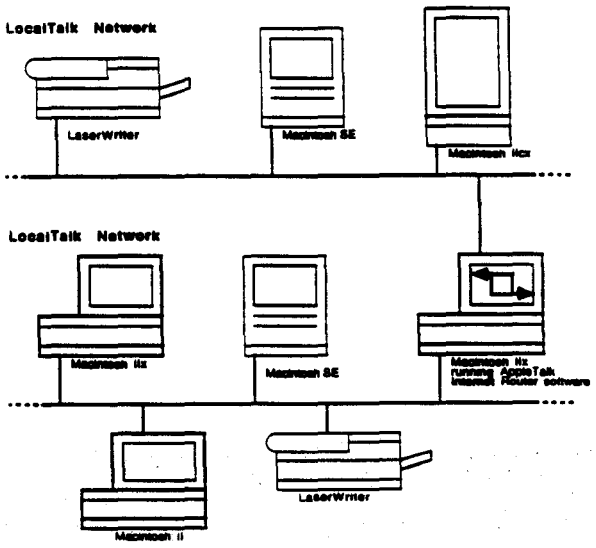


Figure 15-3: Using the AppleTalk Internet Router

MS-DOS personal computers can be connected to the AppleTalk network by using a product called *AppleShare PC*. Users of MS-DOS and Macintosh personal computers can communicate, share data files and have transparent access to Apple LaserWriter printers. Along with the AppleShare PC software a network card must be installed in the MS-DOS personal computer for it to function on the Ap-

pleTalk network system. A LocalTalk PC card from Apple Computer may be chosen, or a Multiple Link Interface (MLI) compliant card from IBM, 3Com or another manufacturer will do. Figure 15-4 illustrates the proper use of AppleShare PC.

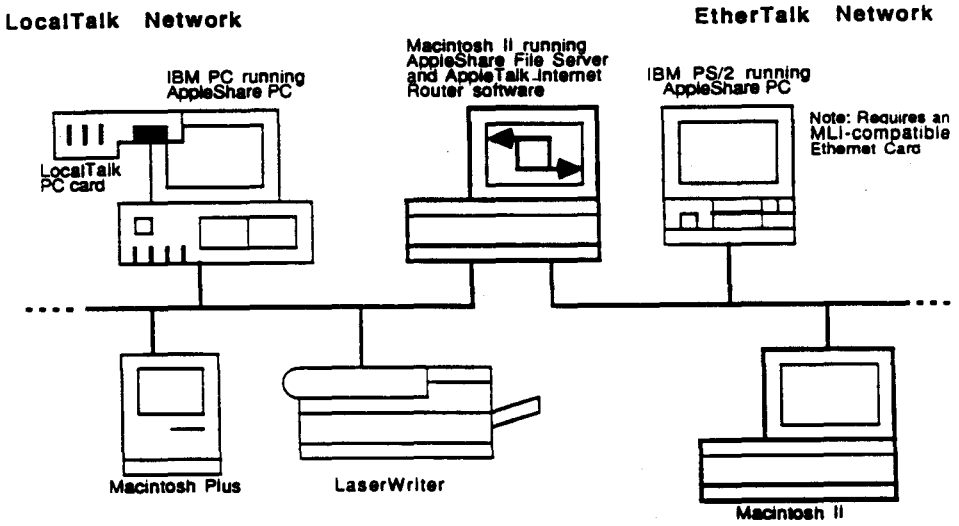


Figure 15-4: Using AppleShare PC

AppleShare File Server makes a Macintosh computer become a dedicated file server. This file server provides a central storage area and a platform for sharing information for all users on the network. *AppleShare Print Server* works as a middle man between the user and the printer. It is a software application that interacts with a printer to print documents either directly or with a print spooler. A background spooler sends a print job the the user's local disk allowing the computer to perform other tasks. The spooler runs in the background until the printer becomes available. A spooler/server is a dedicated computer. A print file is sent to the spooler/server which then takes over the task of waiting for the printer to process the job. Both types of spoolers can be used with AppleTalk.

Other extensions to the AppleTalk network include a monitoring and management tool, *Inter*Poll.*

15.3. AppleTalk Phase 2

AppleTalk Phase 2 is the result of demands from users of large network systems to extend AppleTalk beyond the local work group. It provides significant enhancements to EtherTalk and TokenTalk networks. The routing capabilities have been improved and made more efficient.

AppleTalk Phase 2 provides extended addressing, thus allowing a great many nodes to be addressed on a single network. Each individually-addressable device connected to an AppleTalk network, such as a computer or a LaserWriter printer, is known as a node. With AppleTalk Phase 2, an EtherTalk or TokenTalk network can theoretically include 16 million nodes. The previous limitation was 254 nodes.

Zones are used to logically partition a network making it easier for devices in a zone to locate and access network services. Any node on an AppleTalk network can now belong to any zone set up by the network administrator. A LocalTalk network can be associated with only one zone name. All other AppleTalk networks can be associated with a zone list. This zone naming extension provides the network administrator with more flexibility.

15.4. AppleTalk Integration into Other Environments

AppleTalk networks can also be integrated into DEC environments, IBM environments, or TCP/IP environments.

DEC Environments

AppleTalk for VMS is an implementation of the AppleTalk protocol architecture for VAX/VMS systems. With AppleTalk for VMS installed, a VAX/VMS computer system can participate in an AppleTalk internet. Likewise, Macintosh users on the AppleTalk internet can access the VMS environment of that VAX computer. A VAX can become a full-function AppleTalk router. By routing AppleTalk packets through DECnet wide area networks (a technique called tunneling) the AppleTalk for VMS router can interconnect AppleTalk internets separated by long distances.

The Macintosh keeps its own user interface, and a user need only know how to use AppleTalk services to communicate over the network. To the user, the interactions with the VAX appear as interactions with any other AppleTalk internet node. A Macintosh user in one AppleTalk network in California can send information across a coast-to-coast DECnet link to print a document on a LaserWriter in another AppleTalk network in New York.

IBM Environments

AppleTalk connectivity to IBM environments is provided by TokenTalk hardware and a variety of software packages.

- **TokenTalk software** provides transparent access to AppleTalk network services over token ring networks.
- **SMB File Transfer Utility** allows the user to transfer files between a Macintosh computer and an IBM SMB (Server Message Block) server on a token ring network.
- **MacDFT software** is a full-feature 3270 terminal emulation program.

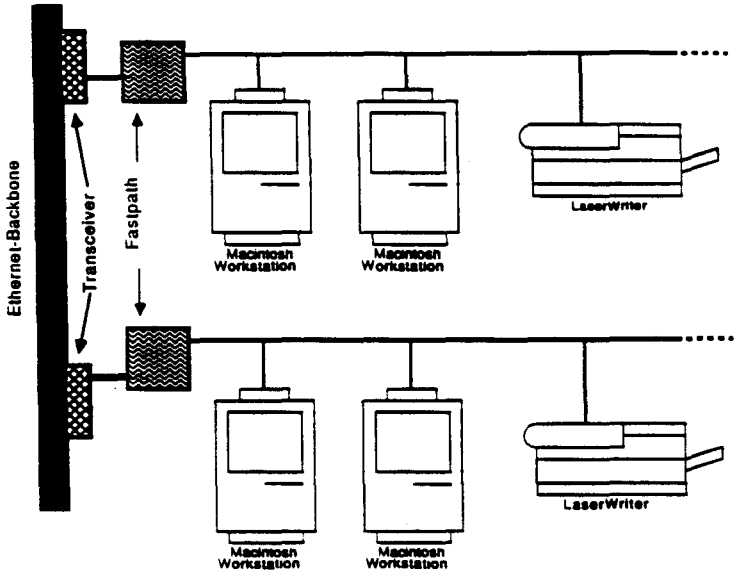
TCP/IP Environment

MacTCP brings the ease of the Macintosh user interface to the TCP/IP environment. MacTCP is a product for application developers who want to create Macintosh applications that use TCP/IP networks. Using MacTCP for example, a user can create a Telnet application that allows him to log on to a host computer from his Macintosh over a TCP/IP network. MacTCP runs under the Macintosh operating system and is co-resident with AppleTalk protocols. It provides a standard platform for developing applications and services using C and Assembly language interfaces.

15.5. Non-AppleTalk Configurations for the Macintosh

There are several products produced by third-party vendors that allow Macintoshs' not necessarily a part of an AppleTalk network to communicate over Ethernet.

- Kinetics Inc. makes a product called the *Fastpath gateway.* Fastpath converts Appletalk protocol into Ethernet packets and vice-versa. It can be used to connect many AppleTalk networks. It enables an entire AppleTalk network to run DECnet. The only drawback is speed. Fastpath is only as fast as LocalTalk (230 kilobits per second). Here at Fermilab, Fastpath is one of the methods used to connect AppleTalk networks to an Ethernet backbone. See Figure 15-5.
- *PacerLink* (formerly PcLink) made by Pacer Software Inc. is an excellent software package for Macintosh connectivity. It supports a number of Ethernet controller boards and provides file transfer, terminal emulation, task-to-task communication as well as many other capabilities.
- *TSSNET* is made by Thursby Software Systems. This package is a software implementation of Digital's DECnet. A Mac/Workstation becomes a DECnet end node. TSSNET only supports one controller board, the Etherport SE controller board from Kinectics Systems.



• **Figure 15-5:** An example of a Fermilab AppleTalk Network

However it is an excellent clone of Digital's DECnet providing a number of DECnet services. TSSnet is used here at Fermilab to connect Macintoshs' to DECnet.

- *VersaTerm* is a terminal emulation program. It is widely used within the Macintosh community here at Fermilab. It allows a user to use a Macintosh as a terminal to VM and other systems.

15.6. References and Further Reading

1. Gibbons, R., "Appletalk Connectivity For The Macintosh", Tech. report CN0042, Fermi National Accelerator Laboratory, 1988.

2. Apple Computer, Inc., *A Guide to Apple Networking and Communications Products,* 1989, M5116

This Page Intentionally Left Blank

Chapter 16

Using Kermit

Kermit is a computer file transfer protocol developed at Columbia University. The protocol is named after Kermit the Frog, star of "The Muppet Show". One reason for Kermit's popularity is that it's free (there could possibly be a distribution fee to pay for the medium and shipping). Another is that Kermit always comes with source code and documentation.

Its rules are designed to ensure that computer files can be transferred from one computer to another correctly and completely, despite the many pitfalls that lie in the way. Kermit is needed to provide communication between computers that are not networked. Users can establish a connection to any other computer that can be dialed on the telephone or reached with a cable.

16.1. How the Kermit Protocol Works

A file is transferred from one computer to another by a pair of Kermit programs, one running on each computer. The Kermit programs carry out the Kermit protocol by sending messages to each other through their communication ports.

The Kermit protocol is character-oriented. Data is transmitted in the form of discrete characters, like A, B, C, rather than in some other form. Most computers agree about how characters are represented, and they agree that there are 128 of them altogether, of which 95 are printable and the other 33 are reserved for control or formatting purposes. The control characters sometimes cause computers and communication devices to react unpredictably. To promote transparency, Kermit encodes control characters as printable character sequences during transmission.

Kermit transfers data by breaking it up into pieces and putting these pieces into packets. A Kermit packet is shown in Fig. 16-1.

The MARK identifies the beginning of the packet. The length field (LEN) specifies how long the rest of the packet is. The sequence number (SEQ) is used to detect lost or duplicated packets. The TYPE field indicates the purpose or

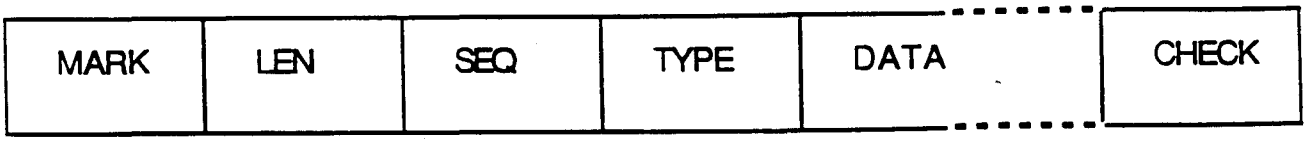


Figure 16-1: Kermit packet layout

contents of the packet: file name, file data, end of file, etc. The CHECK field contains a quantity formed by combining all the other characters in the packet in some way. The sender of the packet computes this value and includes it at the end of the packet. The receiver of the packet does the same computation and checks the result against the value recorded in the packet. If the two values agree, the packet is accepted. If they disagree, the packet has been corrupted and retransmission is required.

Figure 16-2 shows how a typical file transfer proceeds. The file sender waits for a response to each packet before sending the next one, thus making the transfer synchronized. The receiver has time to file away the data. The file sender begins the transfer by transmitting a Send-Initiation packet to the receiver (packet zero in the figure). The "Send-Init" and its responses are greeting messages, in which the two programs settle upon etiquette: the longest packet that will be tolerated, how long to wait for a packet before timing out, and so forth. The sender then transmits a File-Header packet (packet 1) to tell the receiver the name of the file that is about to arrive. Following the file header information are as many File-Data packets as necessary to transmit the entire contents of the file. The File-Header, File-Data, End-of-File sequence is repeated for each file to be sent, and the transaction is closed by an End-of-Transaction packet (packet 54).

An ACK (positive acknowledgement) is sent by the receiver back to the sender for each packet that has been received correctly. Both Kermit programs advance their current packet sequence numbers and move on. If a packet is corrupted in transit, the check will be wrong and the file receiver will NAK (negatively acknowledge) it, causing the sender to retransmit the same packet.

If the file sender does not receive an ACK within a certain time interval (packet 30 in figure), it retransmits the same packet. If the file receiver does not receive an expected packet within the timeout interval, it sends a NAK for the expected packet. The packet number is used to determine if the same packet arrives more than once and avoid writing redundant data into the file. If the same packet is retransmitted too many times, the protocol will declare that the transfer has failed.

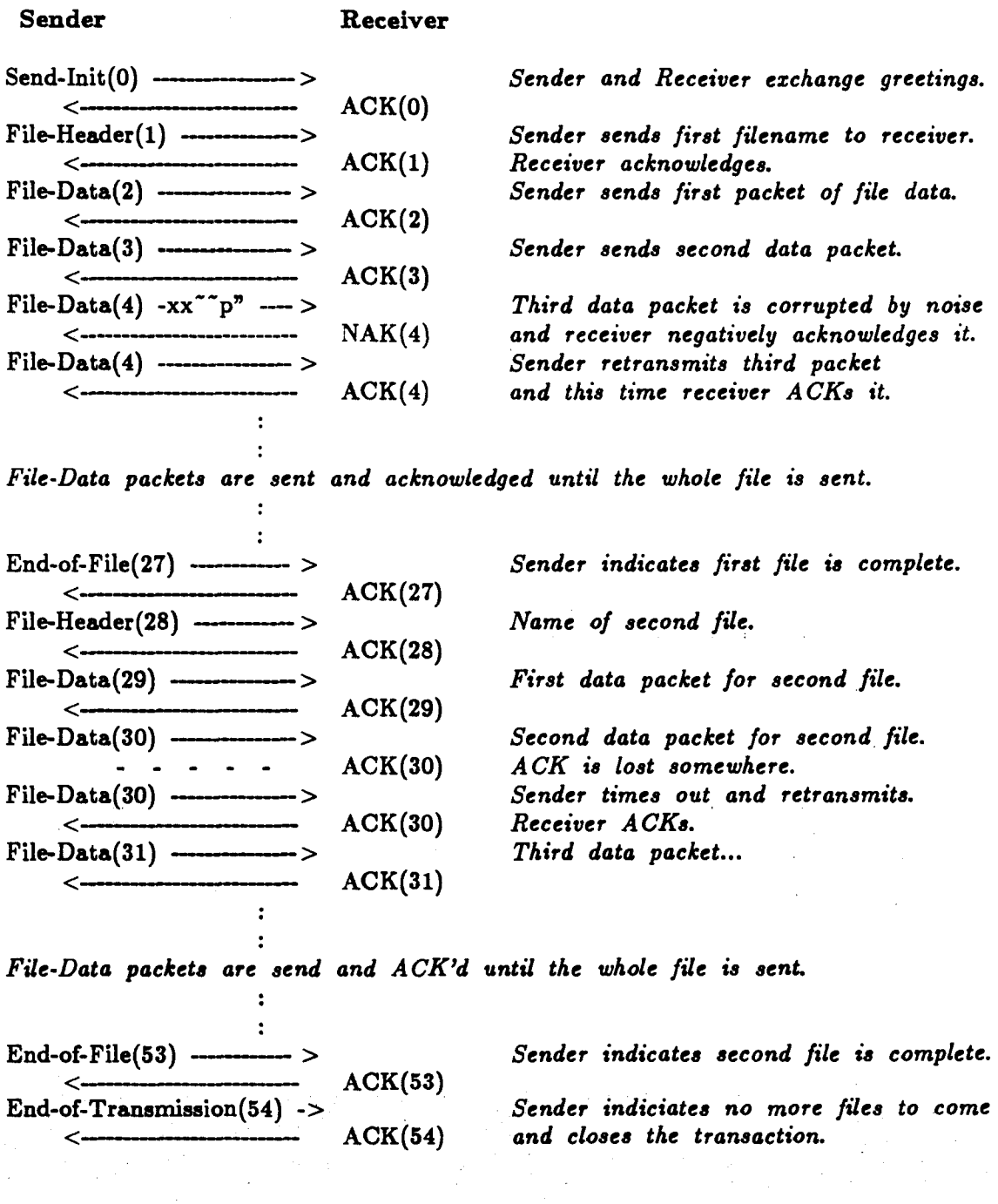


Figure 16-2: Kermit File Transfer Example

16.2. How to Transfer Files with Kermit in General

The most common use of Kermit is between a PC and mainframe. The following discussion assumes that the user has a working PC connected directly or via dialup to the mainframe. It is also assumed that the PC has a Kermit program available and sufficient free disk space to store new files. The user must be able to log into the mainframe and store or read files there. A Kermit program must be available on the mainframe.

First the user runs the local Kermit program (on PC) and issues the CONNECT command. Then the user must login to the remote host and run the remote Kermit program (on mainframe). When Kermit programs are running at each end of the connection, the user must tell *each* Kermit what to do. For instance, to transfer a file from the remote system to the PC, a user must tell the remote Kermit to SEND the file, escape back to PC Kermit, and tell the local Kermit to RECEIVE the file. The transfer will begin and eventually complete. Now the user must clean up. CONNECT back to the remote host, exit from Kermit on the host, log out from the host, escape back to PC Kermit, and exit from it. Transferring a file in the other direction works the same way, but with the SEND and RECEIVE commands interchanged.

When exchanging several files in both directions, the escaping back and forth and typing SEND and RECEIVE can be a bit cumbersome. Most mainframe Kermit programs have a server mode of operation which simplifies the process considerably. Most PC Kermit programs provide special commands required for communicating with Kermit servers. A Kermit server takes all its commands in packet form from the local Kermit program. For example, if local Kermit it told to SEND a file, the remote Kermit server need not be told to RECEIVE it because it will do so automatically. To initiate a file transfer, a user must first issue a command (SEND, RECEIVE, or SERVER) to the remote Kermit, then excape back to the local Kermit and issue the corresponding command (RECEIVE, SEND, or GET).

16.3. Basic Commands

- **?** List the commands, options, or operands that are possible at this point.
- **HELP** Display a summary of Kermit commands and what they do.
- **CONNECT** Act as a terminal to the remots system until the escape sequence is given.
- **SET parameter** Establish or modify various parameters for file transfer or terminal connections. Values can be examined with the SHOW command.

- **SET BAUD** *number* Set the serial port's speed to the given baud rate. Sometimes available as SET SPEED.
- **SHOW** [option] Displays various parameters and information. SHOW ALL will display all current parameter settings. These settings can be changed using the SET command.
- **SEND** *filespec* Send the file or file group specified by *filespec* to the other Kermit, which must be given a RECEIVE command, or else must be in server mode.
- **RECEIVE** Passively wait for a file or file group to arrive from the other Kermit, which must be given a send command.
- **GET** *filespec* Actively request a Kermit server to send the specified file or files.
- **REMOTE** *command* Some Kermit servers may be asked to perform functions beyond sending and receiving files. The specified command is sent to the remeote server. For example, REMOTE DIRECTORY will ask the remote Kermit server to send a file directory listing of the specified remote files to your screen.
- **LOCAL** *command* Very similar to REMOTE command. The specified command will be executed on the local system.
- **BYE** Ask the server to terminate and log out your job from the remote system, so that you need not CONNECT back and clean up.
- **FINISH** Ask the server to terminate, but leave your remote job active so that you can CONNECT to it again.
- **EXIT** Exit from the Kermit program.

16.4. File Types

Kermit is capable of transferring text files and binary files, although it is normally set up to work with text files. Binary files sometimes need special measures. A text file is one that has been created by a human or is intended for reading by a human. A binary file is not intended to be direcly understood by humans.

When text files are transferred between unlike systems, they need to be converted to the format of the target system. Kermit programs will perform format conversions appropriate to text files unless instructed otherwise. Binary files generally should not and cannot be converted to another system's format because their contents are often only useful on the home machine. In order to transfer files without conversion, users must take care. Not all types of binary files can be transferred and recovered intact. In general, issuing the SET FILE TYPE BINARY to *each* Kermit program is one method of transferring binary files.

16.5. Examples

16.5.1. Text File Transfers using Server Mode

The following example will explain how to transfer a *text* file from a PC to the VAX cluster here at Fermilab. The transfer is initiated at the PC. The VMS Kermit will be given the SERVER command and operate as a server. Remember, when using VAX Kermit in server mode, the transfer must be initiated at the PC or other computer. The user must first know where the local PC Kermit resides (is it in a subdirectory?) to invoke it. The user is going to transfer the hypothetcal file PC.DOC from the PC to the VAX.

```
C>Kermit                                    Run Kermit on the PC.
IBM-PC Kermit-MS V2.29  26 May 86
Type ? for help

Kermit-MS>                                  This is the Kermit prompt for the PC.
Kermit-MS>CONNECT                           Connect to the port selector.

FERMILAB [A]= 72                            Respond to prompt.
EXIT NODE ''A''
GO

VISTA> C FNAL                               Connect to FNAL cluster.

USERNAME: username                          Log in.
PASSWORD: password
Last login: Mon Oct 7 18:42:16

$setup kermit                               Run setup procedure.
$kermit server                              Run Kermit on the VAX.

^]c                                         Escape back to local kermit on PC

Kermit-MS>SEND PC.DOC                       Send the file

           Filename : PC.DOC                The adjacent display will appear on your
  KBytes transfered : 3                     screen and continue to be updated until the
Percent transferred : 87%                   transfer is complete.
            Sending : In Progress

  Number of packets : 56
  Number of retries : 0
         Last error : None
      Last warning : None

Kermit-MS>BYE                               Disconnect from server and log out of remote
                                            system - VAX in this case.
Kermit-MS>EXIT                              Exit from PC Kermit
C>
```

Now the user is going to transfer the file VAX.DOC from the VAX to the PC. The transfer will still be initiated at the PC and again the VMS Kermit will operate as a server. Essentially, the same procedure is followed, except the user wants to GET instead of SEND a file. Please note that in either the above ex-

ample or the following example, a group of *text* files can be sent using a wildcard (*) in the file specification for the SEND or GET command.

```
C>Kermit                                        Run Kermit on the PC.
IBM-PC Kermit-MS V2.29  26 May 86
Type ? for help

Kermit-MS>                                      This is the Kermit prompt for the PC.
Kermit-MS>CONNECT                               Connect to the port selector.

FERMILAB [A]= 72                                Respond to prompt.
EXIT NODE ''A''
GO

VISTA> C FNAL                                   Connect to FNAL cluster.

USERNAME: username                              Log in.
PASSWORD: password
Last login: Mon Oct 7 18:42:16

$setup kermit                                   Run setup procedure.
$kermit server                                  Run Kermit on the VAX.

^]c                                             Escape back to local kermit on PC
                                                The procedure is identical to the above
                                                up to this point.

Kermit-MS>GET VAX.DOC                           Get the file

            Filename : VAX.DOC                  The adjacent display will appear on your
  KBytes transfered  : 5                        screen and continue to be updated until the
Percent transferred  : 62%                      transfer is complete.
          Receiving  : In Progress

  Number of packets  : 67
  Number of retries  : 0
         Last error  : None
       Last warning  : None

Kermit-MS>BYE                                   Disconnect from server and log out of remote
                                                system - VAX in this case.
Kermit-MS>EXIT                                  Exit from PC Kermit
C>
```

16.5.2. Text File Transfer : Non Server Mode

Both transfers made above can be done without putting the VMS Kermit into server mode. If a server mode of operation is available, use it, as it simplifies matters. Otherwise, the user must use the procedure shown in the next example. It illustrates the slightly more complex method of transferring text files in nonserver mode. In this case, after using the PC Kermit to connect to the VAX, the user would do the following:

```
$ kermit                          Run Kermit on the VAX - non server mode.
Kermit-32>RECEIVE                 Tell VAX Kermit to receive from PC
              or                  or
          SEND VAX.DOC            Send VAX.DOC to PC.

^]c                               Escape back to local Kermit on PC.]

Kermit-MS>SEND PC.DOC             Send PC.DOC from PC to VAX
              or                  or
          RECEIVE                 receive file from VAX to PC.

                                  Now clean up.
Kermit-MS>CONNECT                 Connect back to VAX - this time
                                  no log in necessary.

Kermit-32>EXIT                    Exit from remote (VAX) Kermit.
$ LOGOUT                          Logout of VAX.
^]c                               Escape back to local Kermit on PC.

Kermit-MS>EXIT                    Exit from local (PC) Kermit.
C>
```

16.5.3. Binary File Transfer : Server Mode

Binary file transfer is not as straightforward. In the following example, the user is sending a file PC.EXE which originated on the PC to the VAX. Then the user wants to recover PC.EXE back to the PC. When the VAX is in server mode, the transfer is initiated at the PC. The major difference between this transfer and a text file transfer it the setting of the file type. First the user transfers the file from the PC to the VAX. After using PC Kermit to connect to the VAX:

```
$ KERMIT SERVERB                  Run Kermit on the VAX (server/binary).
^[c                               Escape back to PC Kermit

Kermit-MS> SET FILE TYPE BINARY   Prepare to send binary data.
Kermit-MS> SEND PC.EXE            Start sending file.

      Transfer in progress display until complete.

Kermit-MS> BYE                    Disconnect from server and log out of remote
                                  system - VAX in this case.
Kermit-MS> EXIT                   Exit from PC Kermit.
C>
```

When the user wishes to retrieve the binary file from the VAX back to the PC, after using PC Kermit to connect to the VAX:

```
$ KERMIT SERVERB                  Run Kermit on the VAX (server/binary).
^[c                               Escape back to PC Kermit
```

```
Kermit-MS> SET FILE TYPE BINARY          Prepare to receive binary data.
Kermit-MS> GET PC.EXE                    Start receiving file.

      Transfer in progress display until complete.

Kermit-MS> BYE                           Disconnect from server and log out of remote
                                         system - VAX in this case.
Kermit-MS> EXIT                          Exit from PC Kermit.
C>
```

If the user wishes to transfer files originating on the VAX to another computer, instead of setting the file type to binary in the receiving Kermit, set the file type to fixed. When recovering this file back to the VAX, do the same (and make sure the VAX Kermit is in server binary mode).

16.5.4. Binary File Transfer : Non Server Mode

A binary transfer in nonserver mode is identical to a text file transfer in nonserver mode except the user must remember to set the file types in both Kermit programs before beginning the transfer. Kermit will still transfer a binary file even if the user forgets to set the file type, but the resulting file will be garbage.

16.6. References and Further Reading

1. da Cruz, Frank. *KERMIT A File Transfer Protocol.* Digital Press, 1987.

2. McQueen, R.C. and Bush, N. VAX/VMS KERMIT. Tech. Rept. PU0051, Fermi National Accelerator Laboratory, 1985. Stevens Institute of Technology.

This Page Intentionally Left Blank

Part V
SECURITY

This Page Intentionally Left Blank

Chapter 17

OSI/RM & Security

We are in the midst of a period wherein global voice and data networks are being designed and developed and there is a conscious effort to bring connectivity to every computing resource. It is imperative that, as we as a society increase our dependence on networked computers, we have security mechanisms in place to prevent against various types of threats. This chapter describes the security architecture that is used as a basis to define security services, mechanisms and protocols. This chapter also defines terminology used in computer and network security and elaborates on the motivation to secure computer networks.

17.1. ISO Security Architecture 7498-2

The International Standards Organization (ISO) 7498 describes the Basic Reference Model for Open Systems Interconnection (OSI/RM). ISO 7498-2 extends the OSI/RM to cover security aspects which are general architectural elements of communication protocols, but which are not discussed in the OSI/RM. The definition of computer and network terminolgy is based on the ISO 7498-2 security architecture. The ISO 7498-2 security architecture:

1. Provides a general description of security services and related mechanisms, which may be provided by the OSI/RM.
2. Defines the positions (layers) within the OSI/RM where the services and mechanisms may be provided.

17.2. Definition of Security

ISO defines *security* in the sense of minimizing the vulnerabilities of assets and resources. An *asset* is defined as anything of value. A *vulnerability* is any weakness that could be exploited to violate a system or the information it contains. A *threat* is a potential violation of security.

17.3. Motivation for Security

As dependence on networked computers is increased, we need to protect against various types of threats. Increasingly, organizations are moving towards distributed computer and data centers. There is a need to secure computer network transactions between these centers since critical information such as design specifications, product strategies and future development plans may be some of communication activities between systems. Another motivation for security is the significant dependence on network applications such as electronic mail as a means to communicate. Also, with society rapidly moving towards global networks it is critical that suppliers of network software and hardware demonstrate system integrity and privacy.

17.4. Security Terminology & Abbreviations

The definition of computer and network security terminology in this section and chapter is based on the ISO 7498-2 security architecture.

Terminology	Description
Access Control	The prevention of unauthorized use of a resource.
Access Control List	A list of entities, together with their access rights, which are authorized to have access to a resource.
Accountability	Property that ensures that entity actions may be traced uniquely to the entity.
Active Threat	Threat of a deliberate unauthorized change to the state of the system.
Authentication Info.	Information used to establish validity of a claimed entity.
Authentication Exch.	Mechanism used to ensure identity of an entity by means of information exchange.

Terminology	Description
Authorization	Granting of rights including those based on access rights.
Availability	Property of being accessible and useable upon demand by an authorized entity.
Channel	Information transfer path.
Ciphertext	Data produced through the use of encipherment. The semantic content of the resulting data is not available.
Cleartext	Intelligible data, the semantic content of which is available.
Confidentiality	Property that information is not made available or disclosed to unauthorized individuals, entities, or processes.
Credentials	Data transferred to establish the claimed identity of an entity.
Cryptanalysis	The analysis of a cryptographic system and/or its inputs and outputs to derive confidential variables and/or sensitive data including cleartext.
Cryptography	Cryptography determines the methods used in encipherment and decipherment. An attack on a cryptographic principle, means, or method is cryptanalysis.
Data Integrity	Property that data has not been altered or destroyed in an unauthorized manner.
Decipherment	The reversal of a corresponding reversible encipherment.
Decryption	See *decipherment.*
Denial of service	Prevention of authorized access to resources or the delaying of time-critical operations.
Digital Signature	Data appended to, or a cryptographic transformation of, a data unit that allows the recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g. by the recipient).
Encipherment	The cryptographic transformation of data to produce ciphertext.
Encryption	See *encipherment.*
End-to-end Encipherment	Encipherment of data within or at the source end system, with the corresponding decipherment occurring only within or at the destination end system.

Terminology	Description
Identity-based Sec. Pol.	A security policy based on the identities and/or attributes of users, a group of users, or entities acting on behalf of the users and the resources/objects being accessed.
Key	A sequence of symbols that controls the operations of encipherment and decipherment.
Key Management	The generation, storage, distribution, deletion, archiving and application of keys in accordance with a security policy.
Link-by-link Encipherment	The individual application of encipherment to data on each link of a communications system. Thus, data will be in cleartext form in relay entities.
Manipulation Detection	A mechanism used to detect whether a data unit has been modified (accidentally or intentionally).
Masquerade	The pretence by an entity to be a different entity.
Notarization	The registration of data with a trusted third party that allows the later assurance of the accuracy of its characteristics such as content, origin, time and delivery.
Passive Threat	The threat of unauthorized disclosure of information without changing the state of the system.
Password	Confidential authentication information, usually composed of a string of characters.
Peer-entity Authentication	The corroboration that a peer entity in an association is the one claimed.
Physical Security	The measures used to provide physical protection of resources against deliberate and accidental threats.
Privacy	The right of individuals to control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.
Repudiation	Denial by one of the entities involved in a communication of having participated in all or part of the communication.
Routing Control	The application of rules during the process of routing so as to chose or avoid specific networks, links or relays.

Terminology	Description
Rule-based Sec. Pol.	A security policy based on global rules imposed on all users. These rules usually rely on a comparison of the sensitivity of the resources being accessed and the possession of corresponding attributes of users, a group of users, or entities acting on behalf of users.
Security Audit	An independent review/examination of system records/activities in order to test for adequacy of system controls, to ensure compliance with established policy and operational procedures, to detect breaches in security, and to recommend any indicated changes in control, policy and procedures.
Security Audit Trail	Data collected and potentially used to facilitate a security audit.
Security Label	The marking bound to a resource (may be a data unit) that names or designates the security attributes of that resource.
Security Policy	The set of criteria for the provision of security services.
Security Service	A service, provided by a layer of communicating open systems, which ensures adequate security of the systems or of data transfers.
Sel. Field Protection	The protection of specific fields within a message which is to be transmitted.
Sensitivity	The characteristic of a resource which implies its value or importance, and may include its vulnerability.
Signature	See *digital signature*.
Threat	A potential violation of security.
Traffic Analysis	The inference of information from observation of traffic flows (presence, absence, amount, direction and frequency).
Traff. Flow Confid.	A confidentiality service to protect against traffic analysis.
Traffic Padding	The generation of spurious instances of communication, spurious data units and/or spurious data within data units.
Trusted Functionality	That which is perceived to be correct with respect to some criteria, e.g. as established by a security policy.

Table 17-1: ISO 7498-2 Security Terminology/Abbreviations

17.5. Security Threats

Security threats to a system may be classified as accidental or intentional and may be active or passive.

17.5.1. Accidental Threats

Accidental threats are those that exist with no premeditated intent. An example of realized accidental threat may be system malfunction.

17.5.2. Intentional Threats

Intentional threats may range from casual examination of computer or network data to sophisticated attacks using special system knowledge.

17.5.3. Passive Threats

Passive threats are those which, if realized, would not result in any modification to any information contained in the system(s) and where neither the operation nor the state of the system is changed.

17.5.4. Active Threats

Alteration of information or changes to the state or the operation of the system is defined as an active threat to a system. An example would be modification of the routing tables of a system by an unauthorized user.

17.6. Types of Attacks

Systems that exist on a network may be subject to specific types of attacks. This section describes some types of attacks.

17.6.1. Masquerade

A masquerade is where an entity pretends to be a different entity. Typically, masquerade is used with other forms of active attack such as replay and modification of messages.

17.6.2. Replay

A replay occurs when a message, or part of a message, is repeated to produce an unauthorized effect.

17.6.3. Modification of Messages

Modification of a message occurs when the content of a data transmission is altered without detection and results in an unauthorized effect. For example, if message "Allow 'Pradeep Patel' to read confidential file 'paper.mss'" is changed to "Allow 'Jake Choudhary' to read confidential file 'paper.mss'".

17.6.4. Denial of Service

Denial of Service occurs when an entity fails to perform its proper function or acts in a way that prevents other entities from performing their proper functions. The attack may involve suppressing traffic or generating extra traffic. The attack may also disrupt the operation of a network, especially if the network has relay entities that make routing decisions based upon status reports received from other relay entities.

17.6.5. Insider Attacks

Insider attacks occur when legitimate users of a system behave in unintended or unauthorized ways. Most known computer crimes involved insider attacks that compromised the security of a system.

17.6.6. Outsider Attacks

The techniques that may be used for outsider attacks include wire tapping, intercepting emissions, masquerading as authorized users of the system and bypassing authentication or access control mechanisms.

17.6.7. Trapdoor

When an entity of a system is altered to allow an attacker to produce an unauthorized effect on command or at a predetermined event or a sequence of events. An example would be, a password validation could be modified so that, in addition to its normal effect, it also validates an attacker's password.

17.6.8. Trojan Horse

When introduced to the system, a Trojan Horse has an unauthorized function in addition to its authorized function. For example, a relay that also copies messages to an unauthorized channel is a Trojan Horse.

17.7. Security Services

The security services described below are included in the ISO 7498-2 security architecture. The services may be placed at appropriate layers. Security services that may be provided optionally within the OSI/RM include the following.

Authentication — These services provide for the authentication of a communicating peer entity and the source of data (data origin). Peer entity authentication provides corroboration to the (N+1)-entity that the peer entity is the claimed (N+1)-entity. Data origin authentication provides corroboration to an (N+1)-entity that the source of data is the claimed peer (N+1)-entity.

Access Control — This service provides protection against the unauthorized use of OSI or non-OSI resources accessible via OSI protocols.

Data Confidentiality — This service provides for the protection of data from unauthorized disclosure. Data confidentiality services include connection confidentiality; connectionless confidentiality; selective field confidentiality; and traffic flow confidentiality.

Data Integrity — This service provides protection against active threats. Data integrity services include connection integrity with recovery; connection integrity without recovery; selective field connection integrity; connectionless integrity; and selective field connectionless integrity.

Non-repudiation — Repudiation is defined as the denial by one of the entities involved in a communication of having participated in all or part of the communication. Non-repudiation service may take one or both of two forms: non-repudiation with proof of origin and/or non-repudiation with proof of delivery.

17.8. Security Mechanisms

Security mechanisms implement security services. Security mechanisms are of two types: specific security mechanisms and pervasive security mechanisms.

17.8.1. Specific Security Mechanism

Specific security mechanisms may be incorporated into an appropriate layer in order to provide some of the security services described earlier. Specific security mechanisms include:

Encipherment — May be used to provide confidentiality of either data or traffic flow information.

Digital Signature Mechanisms — This mechanism may be used to define two procedures: signing a data unit and verifying a signed data unit.

Access Control Mech. — Access control mechanisms may be involved at the origin or any intermediate point to determine if the sender is authorized to communicate with the recipient and/or to use the resources. Mechanisms may be based on authentication information such as passwords; security labels; duration of access; time of access or route of attempted access.

Data Integrity Mech. — Mechanisms such as time stamping, sequence numbering, cryptographic chaining may be used to provide integrity of a single data unit or field and the integrity of a stream of data units or fields.

Authentication — Some of the techniques that may be applied include authentication information such as passwords; use of characteristics and/or possessions of the entity; digital signature and/or notarization.

Traffic Padding — Traffic padding may be used to provide various levels of protection against traffic analysis.

Routing Control — Routes may be chosen either dynamically or by prearrangement so as to use only physically secure sub-networks, relays or links.

Notarization — Each instance of communication may use digital signature, encipherment and integrity mechanisms as appropriate to the service being provided by the notary. Properties such as data origin, time and destination can be assured by the provision of a notarization mechanism.

17.8.2. Pervasive Security Mechanisms

These mechanisms are not specific to any particular security service and are in general directly related to the level of security required.

Trusted Functionality
: May be used to extend the scope of or establish the effectiveness of other security mechanisms.

Security Labels
: May be used to indicate sensitivity level. It is additional data associated with the data transferred or may be implied by the use of a specific key to encipher data.

Audit Trail
: Permits detection and investigation of breaches of security. The logging or recording of information is considered to be a security mechanism.

Security Recovery
: Deals with requests from mechanisms such as event handling and management functions, and takes recovery actions as the result of applying a set of rules.

17.9. Security Management

OSI security management is concerned with the management of security services and mechanisms. Distribution of cryptographic keys, the setting of administratively imposed security selection parameters and the reporting of security events are examples of the management and operation of services and mechanisms. There can be many security policies imposed by the administration of distributed open systems and the OSI security management standards provide the support for such policies.

17.10. References and Further Reading

1. *Information Processing Systems -- OSI Reference Model -- Part 2: Security Architecture.* ISO, 1989. Publication No. 7498 Part 2.

Chapter 18

UNIX and Security

In the past few years several vendors have submitted secure versions of UNIX to the NCSC for formal evaluation. Vendors that have submitted products for evaluation include Sun Microsystems' SunOS Multi-Level Security (MLS), AT&T's UNIX System V/MLS, and IBM's Secure Xenix (now marketed as Trusted Xenix by Trusted Information Systems, Inc.). AT&T System V/MLS was recently evaluated at B1 level of security. AT&T's System V/MLS differs from SunOS/MLS in that it was primarily designed for its 3B2 minicomputer with multiple dumb terminals. SunOS/MLS was designed for a distributed workstation computing environment. SunOS/MLS is currently being evaluated by the NCSC for a B1 rating. Sun Microsystems' decided to forego a formal C2 evaluation. Most UNIX systems, if evaluated, would have a C1 or C2 level of security.

The objective of this chapter is to define a model for UNIX security without the addition of third-party security software. The first section includes a discussion on how system administrators/managers may enhance security for an existing environment. The next section describes how the end-user environment can be made secure. This includes information on restricting permissions on newly created objects; directory and file security; search path and parameters set and defined in login procedures such as *.profile*, *.cshrc* and *.login*. The last section details how programs and applications developed on the system may have security implications. Information and recommendations are provided on source code management systems; analysis of return codes; development of privileged programs; and how signals and pipes may affect security.

18.1. State of the System

How can a system administrator/manager use security-related utilities, files and functions to define a model for a secure environment. Careful planning and awareness of the types of threats that a system may experience are key to defining a security policy that leads to a secure environment.

18.1.1. Super-User Sessions

What defines an account to be a super-user account? Any login name that has a UID of zero is a super-user account. Typically, on most UNIX systems the login name for a super-user account is **root**. The super-user account is the most privileged account on the system: it can read/write any file irrespective of the permissions defined, execute any file with an execute permission bit on, change any permission or ownership attribute, create device special file and do other operations restricted to UID zero.

Disabling root Account

On a UNIX system the file */etc/passwd* is an administrative database that contains information on usernames, passwords (encrypted), user ID (UID), group ID (GID), user's real name, pathname of the HOME (login) directory and the login shell. An administrator may choose to modify the */etc/passwd* file to reflect the following:

```
indolusa:EHCbM85S41wKc:0:1:privileged account:/:/bin/csh
root:*:0:1:privileged account:/:/bin/csh
```

Access to the **root** account is disabled. The super-user account that may be used to login is **indolusa**. It is more difficult for an intruder (with no account on the system) to break-in to the system since he would have to guess both the account name and a password.

Disabling Direct Login to the root Account

While an administrator may choose not to define another privileged account such as ***indolusa***, it is critical that users use **/bin/su** and not **login** to switch to the privileged account. Direct login to the privileged account should be disabled if possible. For example, on SunOS 4.0.3 to disable direct login to the **root** account remove the word "secure" from every entry in the */etc/ttytab* file. In the following expression remove the word "secure."

console "usr/etc/getty std.9600" sun on secure

This prevents direct login to the privileged account. Also, when entering single-user mode the system requires the super-user password to be entered before the shell is started. This is a security measure that is especially critical if the systems are not physically secure. On a SunOS system it is possible for someone to

enter the aborting sequence, L1-A, and then boot the system in single-user mode. Removing the word "secure" from the file */etc/ttytab* requires a user to enter the root password even if the system is booted in single-user mode. To disable direct login on a DIGITAL ULTRIX 4.0 system remove the word "secure" from all entries in the file, */etc/ttys*.

When typing **/bin/su**, it is important that the "-" option of **su** be specified. The "-" option performs a complete login. Variables HOME and SHELL are defined based on the */etc/passwd* file; PATH is set to */usr/ucb /bin /usr/bin* and the current working directory is set to /. The following example shows the differences in the definition of variables after **/bin/su -** has been executed:

```
cdsun1.fnal.gov{pabrai}2: printenv
TERM=vt100
HOME=/home/pabrai
SHELL=/bin/csh
USER=pabrai
PATH=/bin:/usr/local:/usr/ucb:/usr/bin:/home/pabrai/bin
LOGNAME=pabrai
PWD=/home/pabrai

cdsun1.fnal.gov{pabrai}3: /bin/su -
Password:
cdsun1.fnal.gov# printenv
USER=pabrai
LOGNAME=root
HOME=/
SHELL=/bin/csh
PATH=/usr/ucb:/bin:/usr/bin:/etc:/usr/etc:/usr/sundesk/bin:\
    /usr/sunlink/dni
TERM=vt100
PWD=/
cdsun1.fnal.gov#
```

Also, it is critical that the full pathname, such as */bin/su*, be used to specify all programs that are invoked as a super user.

18.1.2. Directory Permissions

Discretionary access control may be applied on files, groups of files and directories. File permission and other information are stored as a 16-bit word within the *inode* (information node). Table 18-1 provides information on permission bits and their octal values:

Numbering the bits from the least significant bit, the first nine bits of the word contain data representing the actual file permissions (r,w,x) or indicate that file access is denied (-). The next three bits provide information relevant to file operation if the file is an executing program. Together, these 12 bits comprise the mode of operation. If the 12^{th} bit is on (SUID) when the file is executed,

Bits 16-13	Information Node	File Identification Bits	
Bit 12	(Set User-Id) SUID	Program Execution Bits	4000
Bit 11	(Set Group-ID) SGID		2000
Bit 10	(Sticky-bit) s-bit		1000
Bit 9	Owner:r	Owner Permission Bits	0400
Bit 8	Owner:		0200
Bit 7	Owner:x		0100
Bit 6	Group:r	Group Permission Bits	0040
Bit 5	Group:w		0020
Bit 4	Group:x		0010
Bit 3	Other:r	Other Permission Bits	0004
Bit 2	Other:w		0002
Bit 1	Other:x		0001

Table 18-1: Directory and File Permissions

the effective owner of that process will be changed from the invoker's UID to the UID of the file's owner. If the 11^{th} bit is on (SGID) when the file is executed, the effective group of that process will be changed from the invoker's GID to the GID of the file's owner. For directories in 4.x BSD, an enabled GID bit indicates that newly created files take the group of their parent directory; a disabled GID bit indicates that files take the effective group of the owner (as in System V). If the sticky-bit (s-bit) is on for an executable file, the creating process image will be retained in swap space after execution is completed. For directories in 4.x BSD, the sticky-bit indicates that the file's owner, the directory's owner, or the super-user can remove it.

The command **ls -agl** (in 4.x BSD)or **ls -l** (in Sstem V) provides information on group codes and permissions associated with files. File permissions may be expressed either as octal numbers or symbolic values. Table 18-2 provides information on protections recommended for directories and files on an DIGITAL Ultrix 4.0 system.

File/Dir	Owner	Group	Mode	Security Relevance
COMMAND DIRECTORIES				
/	root	system	755	root of all file system and superuser home directory
/usr	root	system	755	a file system hierarchy
/bin	root	system	755	user commands
/usr/bin	root	system	755	additional user commands
/etc	root	system	755	commands for system maintenance and administration
/usr/etc	root	system	755	more commands for system maintenance and administration
/etc/sec	root	system	755	commands for security administration
/usr/etc/sec	root	system	755	more commands for security administration
/lib	root	system	755	a link to /usr/lib
/usr/lib	root	system	755	many system executables, such as compiler and system libraries
/usr/ucb	root	system	755	certain .commands that are Berkeley extensions
/usr/new	root	system	755	additional optional commands
/usr/local	root	system	755	commands with a local origin
/usr/local/bin	root	system	755	additional commands with a local origin
SPECIAL FILES				
/dev	root	system	755	home directory for special files
/dev/<disktype>*	root	system	600	buffered disk system
/dev/r<disktype>*	root	system	600	unbuffered disk system
NETWORK FILES				
/etc/exports	root	system	644	names of file systems available for NFS export (including restrictions)
/etc/hosts	root	system	644	names and addresses of remote hosts with network access to local host
/etc/hosts.equiv	root	system	600	grants remote user access to local system without password
MISCELLANEOUS FILES				
/usr/etc/subsets \				
/*.inv	root	system	644	inventories of software subsets

File/Dir	Owner	Group	Mode	Security Relevance
/etc/fstab	root	system	644	file system configuration
/etc/ttys	root	system	644	terminal port initialization data
/etc/gettytab	root	system	644	terminal configuration data
/usr/lib/crontab	root	system	644	commands & instructions for their execution by the cron command
/etc/rc	root	system	640	controls automatic startup process
/etc/rc.local	root	system	640	site-specific startup info
/var/adm/crash	root	system	700	directory for system dump files
/etc/svc.conf	root	system	644	system configuration information

Table 18-2: Protections for files and directories

UNIX contains many security-related utilities. Table 18-3 lists UNIX security-related utilities and provides a brief description.

Utility	Description
acctcom	display audit information
chgrp	change file group owner
chmod	change file access permissions
chown	change file owner
chroot	change root directory to restrict user environment
crash	system debugger (restrict to super-user only)
crontab	handle user crontab file
crypt	encrypts/decrypts data using non-DES algorithm
ed -x	editor: encrypts/decrypts data using non-DES algorithm
find	walk file system tree (may be used by security audit)
fsdb	file system debugger (restrict to super-user only)
id	display effective and real UID and GID
login	allow user to login
logname	get login name of owner of current process
makekey	output of 2-byte salt and 11-byte key using DES algorithm
mknod	create an ordinary, directory or special file
ncheck -i	list file name(s) associated with i-node
ncheck -s	list set-user-ID and special files
newgrp	change GID
passwd	change login password
ps	display list of active processes
red	restricted editor: encrypts/decrypts using non-DES algorithm
rsh	restricted shell
su	change effective and real UID to that of another user
umask	set file-creation mask
vi -x	editor:encrypts/decrypts data using non-DES algorithm

Table 18-3: UNIX Security-Related Utilities.

Table 18-4 lists UNIX security-related files.

File/Library	Description
/dev/console	system console (should not be writable to others)
/dev/kmem	kernel memory (should not be accessible)
/dev/mem	system memory (should not be accessible)
/dev/sctfd*	floppy disk special files
/dev/sctmt*	tape special files
/dev/tty?*	terminals (should not be writable to others)
/etc/group	group file (passwords encrypted)
/etc/passwd	password file (password encrypted)
/etc/utmp	current status of login activity
/etc/wtmp	log of login activity
/lib/libc	C-language library with crypt
/lib/libp/libc	profiling C-language library with crypt
/usr/adm/*acct*	command audit trail
/usr/adm/sulog	log of su activity
/usr/adm/wtmp	login session audit trail

Table 18-4: UNIX Security-Related Files.

Table 18-5 lists UNIX security-related functions.

Function	Description
chmod()	change file access permissions
chown()	change file owner and/or group owner
chroot()	change root directory to restrict user environment
creat()	create file with specified access permissions
crypt()	generate an encrypted password using DES algorithm
cuserid()	get login name
encrypt()	encrypt/decrypt data using DES algorithm
endgrent()	close /etc/group after using getgrent() or setgrent()
endpwent()	close /etc/passwd after using getpwent() or setpwent()
fstat()	obtain status information for an open file
ftw()	walk file system tree (may be used by security audit)
getgrent()	get next group entry from /etc/group
getgrgid()	get group entry from /etc/group for specified GID
getgrname()	get group entry from /etc/group for specified group name
getegid()	get effective GID of process
geteuid()	get effective UID of process
getgid()	get real GID of process
getlogin()	get login name of current process
getpass()	display prompt and read password (with echo suppressed)
getpwent()	get next password entry from /etc/passwd
getpwnam()	get entry from /etc/passwd for specified login name
getpwuid()	get entry from /etc/passwd for specified UID
getuid()	get real UID of process
link()	make a new path to an existing file
logname()	get login name of owner of current process
mknod()	create an ordinary, directory, or special file
putpwent()	write password file entry to specified file
setgid()	set effective GID **root** sets real GID also)
setgrent()	rewind /etc/group file
setkey()	set the key for subsequent use with *encrypt()*
setpwent()	rewind */etc/passwd* file

Function	Description
setuid()	set effective UID (root sets real UID also)
signal()	specify action to be taken on receipt of a specified signal
stat()	obtain status information for a file
umask()	set file-creation mask
unlink()	remove (a path to) a file

Table 18-5: UNIX Security-Related Functions.

18.1.3. su to Another Account

It is possible to use the /bin/su command to assume the identity of another user account. However, the /bin/su command should not be used to login to a general user account that has UID 0 privileges since it is unnecessary for the system administrator to learn the password of the general user in order to (say) change the characteristics of the user's objects. Either the administrator can use the privileged account to make the changes required or have the user assist in the operation.

18.1.4. Pseudo-User Accounts

A pseudo-user account is one that does not correspond to a human user of the computer. The **root** account is an example of a pseudo-user account. Pseudo-user accounts are also referred to as default accounts. Other pseudo-user accounts include: **sync, sys, acct, audit, mail, lp, uucp,** etc. The National Computer Emergency Response Team (CERT) recently verified several reports of unauthorized access to the Internet connected UNISYS UNIX systems. The intruder gained access to these systems by logging into vendor supplied pseudo-user accounts; accounts that had not been given passwords by the systems administrators.

Typically, pseudo-user accounts on a SunOS system include:

```
root:S4hXmjajC1JqI:0:1:Operator:/:/bin/csh
nobody:*:65534:65534::/:
daemon:*:1:1::/:
sys:*:2:2::/:/bin/csh
bin:*:3:3::/bin:
uucp:*:4:8::/var/spool/uucppublic:
news:*:6:6::/var/spool/news:/bin/csh
ingres:*:7:7::/usr/ingres:/bin/csh
audit:*:9:9::/etc/security/audit:/bin/csh
sync:*:1:1::/:/bin/sync
sysdiag:*:0:1:Old System Diagnostic:/usr/diag/sysdiag:\
   /usr/diag/sysdiag/sysdiag
sundiag:*:0:1:System Diagnostic:/usr/diag/sundiag:\
   /usr/diag/sundiag/sundiag
sundiag:*:0:1:System Diagnostic:/usr/diag/sundiag4c:\
   /usr/diag/sundiag4c/sundiag
dni:*:376:376:Default DNA Account:/tmp:
```

An "*" in the password field (the second field) indicates that logins to those accounts are disabled. CERT reported in October, 1989 that a problem had been discovered in SunOS 4.0.x rcp. If exploited, this problem could allow users of other trusted machines (listed in */etc/hosts.equiv* or */.rhosts*) to execute **root** privilege commands. The immediate fix suggested for this problem dealt with the pseudo-user account, **nobody**. The **nobody** account is assigned under certain conditions to users that access a system over the network. The patch recommended changing the **nobody** entry in the */etc/passwd* file from

```
nobody:*:-2:-2::/:
```

to

```
nobody:*:32767:32767:Mismatched NFS ID's:/nonexistent:/nosuchshell
```

SunOS 4.1 systems are currently being shipped with both the UID and GID fields set to 65534 for POSIX compliance.

Specific individuals should be responsible for pseudo-user accounts. Also, pseudo-accounts should not be shared. If more than one individual needs to access a specific pseudo-user account then separate accounts should be created. These separate accounts must belong to the same group. For example, pseudo-users **vmsprint** and **vmprint** may be separate accounts that through group permissions belong to the "lp-grp" subsystem. Other groups that may be created include gen-admin, mail-grp and uucp-grp.

A new pseudo-user account should be created for users who need specific access to certain security-related programs and data files. The pseudo-user account can then be made the owner of security objects without having to give out unlimited privilege.

18.1.5. UID, GID, SUID and SGID

All objects, such as files and directories, have attributes indicating UID and GID. UIDs are identified in */etc/passwd* and GIDs are identified in */etc/group*. UID and GID are numbers that the system uses to identify users and groups. A process can change an objects UID, GID and permissions if the process effective UID matches the object's UID or if the process effective UID is 0. For example, executing **/bin/su** creates a new process with the following security characteristics:

1. Real UID and GID (obtained from */etc/passwd*).
2. Effective UID and GID (UID & GID currently in effect).
3. Mask (restrictions on the attributes applied to objects created by the process).

A process can create another process using *fork*. These subprocesses or child processes inherit most of the parent's current attributes. Each process is assigned four numbers to indicate who the process belongs to: real and effective UID and real and effective GID. Typically, the effective UID and GID are the same as the real UID and GID - thus when you execute a program the effective and real UIDs and GIDs are set to the user's UID and GID respectively. When the set-user-id (SUID) permission is set on a program, all processes created from that program have the effective UID of the program's owner and not the user's UID or GID. Since file access permissions are determined from the effective UID and not the real UID, the process created from a SUID program has the same access permissions as the owner of that program, regardless of who executes the program.

A program is typically given SUID/SGID attributes so as to allow its users the same access as the program's owner has to certain objects, such as a database. Normally, there is no reason for general users to own SUID/SGID programs. If a user does own a SUID/SGID program, it is important to use **chmod** to deny anyone else permission to read or write it. SUID programs should be kept in write-protected directories to prevent their replacement by unauthorized users.

SUID programs created by the **root** account and SGID programs of any of the system pseudo-accounts should be monitored frequently - since an intruder who has broken into the **root** account once may install and hide a SUID program that would enable superuser privileges even if the original security hole is fixed. SUID and SGID programs, particularly those with shell escapes must be very carefully written to prevent improper use of the program.

Effective GIDs behave like effective UIDs. A process whose Set-Group-ID (SGID) permission is set, runs with the group access rights of the group associated with the program. Setting the SGID is safer than SUID since group permissions may give equal access. However, if there are a large number of users in a group then the SGID permission would provide access to a large number of files owned by users in the group. A solution to this problem is to create a new group and

add just one user (owner of the SGID program) to it. Next, change the group associated with the SGID program to the group that previously existed. Thus, if someone finds a security hole it will not compromise any other user on the system. The command to find all SUID and SGID files on the system is:

```
# find / -type f -a \( -perm -4000 -o -perm -2000\) -print
```

The **find** command is a general-purpose command for searching the file system. The output of the above command lists all SUID and SGID programs. Programs that are not in directories such as */bin*, */etc*, */usr/bin*, */usr/ucb* and */usr/etc* should especially be checked. CERT had reported recently that SunOS 4.0.3 systems had been distributed with the SUID bit on for the file, */usr/etc/restore*. The SUID bit should be reset since it can be exploited by an intruder. Enter the following command to reset the SUID bit:

```
# chmod 750 /usr/etc/restore
```

You can copy (**cp**) an SUID or SGID program from another user's account if you have read and execute permission on the source directory. The copy in your account retains its SUID/SGID permissions, but you become the owner. However, if the source file is owned by root, the system always removes SUID/SGID permissions when the file is copied. If the file has the same name as the SUID/SGID program, the copy takes the place of the original file, but the original permissions apply.

You can move (**mv**) an SUID/SGID program from another user's account to your account if you have write and execute permission on the source directory. If both files are on the same file system, the operation is a renaming of the file, therefore, the ownership is not changed, and SUID/SGID permissions are not removed. If the files are on different file systems, the operation is a copy and a delete, therefore, the ownership is changed and SUID/SGID permissions are removed.

18.1.6. The Network File System

The Network File System (NFS) is a Remote Procedure Call (RPC) based service that enables hosts to share file systems across a network. The file server's */etc/exports* file lists the exported directories and which clients are authorized to access them. The three options that must be considered when exporting file systems are: **ro**, **root=** and **access=**. **ro** specifies that clients are limited to read-only access to the specified directory. The **root=***hosts* option indicates that super-user access to the directory is given only to super-users from the specified hosts. The **access=***client* indicates that access to a particular directory should be given to the named client, clients or netgroup. The following is an example of the file, */etc/exports*:

```
/usr    -access=cdsun2
/home   -access=cdsun2
#
/export/root/cdsun2 -root=cdsun2,access=cdsun2
/export/swap/cdsun2 -root=cdsun2,access=cdsun2
/export/exec/sun4c
```

If no **access=*client*** is specified for a file system then any host on the network can use NFS to mount the exported file system.

NFS has been known to have the following security problems:

1. Verification of credentials occurs only at mount time when the client gets from the server a piece of information that is its key to all further requests: the file handle. Security can be broken if you can figure out a file handle without contacting the server (perhaps by tapping into the network or by guessing).
2. After an NFS file system has been mounted, there is no checking of credentials during file requests. Thus, if a file system has been mounted from a server that servers multiple clients there is no protection against someone who has **root** permission on their machine using **/bin/su** - gaining unauthorized access to other people's files.
3. The server method that NFS uses to circumvent the problem of not being able to authenticate remote client super-users: denies them super-user access altogether.

However, a new authentication system based on DES corrects the above specified problems. Guessing file handles is no longer a problem since in order to gain unauthorized access, the intruder will also have to guess the right encrypted timestamp to place in the credential. The DES based system authenticates machines in the network.

For example, the SunOS 4.1 system uses authentication system based on DES encryption and public key cryptography to authenticate both users and machines in the network.

18.1.7. The Trivial File Transfer Protocol

The Trivial File Transfer Protocol (TFTP) may be used to download fonts to X-terminals or to allow diskless workstations to boot over the network. Typically, implementations of the TFTP have been known to have security holes. The Department of Energy (DOE) Computer Incident Advisory Capability (CIAC) had reported some time ago that there are security holes related to ***tftp*** and ***rwalld*** that leave certain systems vulnerable to intrusion. The holes, when used in a very specific scenario, permit an intruder to attack UNIX systems and assume super-user privileges. The ***tftp*** hole allows any user without first logging in to read any readable file and write any writeable file on a remote system using the Internet network. The hole existed in SunOS 3.x systems but has been fixed in

SunOS 4.x systems. The following test verifies that the version of TFTP installed on the system has been patched to prevent some security holes:

```
%  tftp
tftp>  connect  hostname
tftp>  get /etc/passwd  stolen.passwd

Error code 1: File not found

tftp>  quit %
```

18.2. End-User Environment and Security

This section provides information on how the end-user environment can be made secure. Information is provided on password selection; **umask** and **chmod** commands that can be used to create and maintain secure file permissions; search path and definition of critical terms in the user login files.

18.2.1. Password

Critical information associated with each user account is maintained in the file, */etc/passwd*. This is obviously the most important system file from the perspective of security. Enter the following command to ascertain that there are no empty password fields in the file, */etc/passwd*:

```
%   awk -F:   '$2 == "" {print}'  /etc/passwd
```

The only line displayed as a result of executing the above command is +::0:0:::. Further, no two users should have the same UID. Execute the following command to verify that this is the case:

```
% sort -t: +2n /etc/passwd | awk -F: '{if (prv == $3) print; prv = $3}'
```

The only output displayed should be:

```
+0::0:0::: root:encrypted passwd:0:1:/:/:/bin/csh
```

The file permission for */etc/passwd* should be set to 644. The file must be owned by **root**. The password associated with the **root** account should be changed frequently. When manually editing the password file to add a new account it is important to indicate that the new user's password has aged. This directs login to force the user to choose a new password the first time the user logs in. In general the password aging interval must be set so that users change their password periodically.

Since the file, */etc/passwd*, contains user account information including encrypted passwords, it is possible for a user to obtain a copy of */etc/passwd* and decrypt

the password entry for commonly used passwords. Passwords based on the following are relatively easy to decrypt: your username; a word in the dictionary; individual names, pet names; addresses and places; any of the previously mentioned spelled backwards; passwords less than six characters in length.

Robert Morris and Ken Thompson conducted experiments to determine typical user's habits in the choice of passwords. In a collection of 3,289 passwords gathered over a long period of time:

```
15 were a single ASCII character
72 were strings of two ASCII characters
464 were strings of three ASCII characters
477 were strings of four alphanumerics
706 were five letters, all upper-case or all lower-case
605 were six letters, all lower-case
```

A total of 2,831 or 86% of passwords fell into one of the above specified categories. A dictionary search alone (which required only 5 minutes of computing time) reproduced a third of all passwords. It is hence important that we select good passwords.

Conventional wisdom states that good passwords are typically based on a combination of the following: more than six characters in length; use a mix of alphabetic and non-alphabetic characters such as underscore "_" or square bracket "[" or period "."; create an acronym for an uncommon phrase (thus, "India and USA are the two biggest democracies" could be IaUat2bd); use numbers and upper and lower case characters. Finally, when selecting a password, choose one that is reasonable in complexity and isn't difficult to remember.

18.2.2. Search Path

The shell utility uses the PATH variable to select the file that may have been specified as part of the command syntax. The syntax for setting the PATH variable is:

```
PATH=pathname:pathname:pathname...                (in the Bourne Shell)

set path = (pathname pathname pathname ...)       (in the C Shell)
```

For example, a typical definition for the search path in the *.profile* file maybe:

```
PATH=/bin:/sys/bin:/usr/bin:/usr/ucb:$HOME/bin:.  (in the Bourne Shell)
export PATH

set path = (/bin /sys/bin /usr/bin /usr/ucb $HOME/bin .)
                                                  (in .cshrc for the C Shell)
```

The statement "export PATH" exports the search path to all subprocesses. "." refers to the current working directory - this should be avoided if possible or specified last in the search path. An intruder may place a Trojan horse in place

of a common system command in a user area. If the current working directory is specified last in the search path, the Trojan horse will not be executed. After symbolic substitution, only absolute pathnames should be used in the search path. If you have to occasionally use programs or applications from another user's area, it is safer to set an environment variable to the proper directory instead of including their directory in the PATH definition. If you need to frequently run applications from another area then it is best to define an ALIAS in the *.cshrc* file that points to the program or application. For example, to display to the terminal the content of file USERUIDGID from */home/kalidas/util*, enter the following expression in the *.cshrc* file:

```
alias    USERUIDGID        more    /home/kalidas/util/USERUIDGID
```

Note that null entries in the path list point to the current working directory. It is a good practice to create a *$HOME/bin* directory in your area to hold private executables and scripts. The directory *$HOME/bin* may then be appended to the path list definition in .profile. Finally, open or temporary directories should not be included in the path list.

18.2.3. Objects: umask and chmod

The commands, **umask** and **chmod**, can be used to create and maintain secure file permissions. The umask command sets or displays your default file permission; although, several programs set the default permission to 666 for ordinary files and 777 for directory and executable files. The file creation mask removes permissions from the default permissions, thus determining actual permissions assigned to a new file. The resulting permissions are the original permissions "anded" with the one's complement of the umask mask. The following example illustrates how umask works.

```
Default File Permission (666)   420 420 420   (666)     rw- rw- rw-
umask(077)                      000 421 421   (077)     --- rwx rwx
                                ===========             ===========
                                420 000 000             rw- --- ---
```

If you need to grant read and execute permissions to members of your group; and read, write and execute permissions to yourself then set your umask to 027 in either *.profile* for the Bourne Shell or *.login* for the C Shell. Table 18-6 provides information on umask values.

The **chmod** command may be used to change the permission of any file owned by you. If you want to change the existing permissions to absolute permissions then octal arguments may be specified with the chmod command. Relative permissions may be used to add or subtract from existing file permissions. The following example explains the concept.

```
   -rwxr-xr-x                (initial permission for proton.f)
%  chmod 700                 (absolute permission)
   -rwx------                (final permission for proton.f)
```

umask value	Description	Text File	Executable File or Directory
002	No (w) for other	rw-rw-r--	rwxrwxr-x
006	No (rw) for other	rw-rw----	rwxrwx--x
007	No (rwx) for other	rw-rw----	rwxrwx---
022	No (w) for group+other	rw-r--r--	rwxr-xr-x
026	No (w) for group;		
	No (rw) for other	rw-r-----	rwxr-x---
027	No (w) for group;		
	No (rwx) for other	rw-r-----	rwxr-x---
077	No (rwx) for group+other	rw-------	rwx------

Table 18-6: Description of umask values

```
   -rwxr-x-r-x                (initial permission for electron.f)
%  chmod g-rx, o-rx           (relative permission)
   -rwx------                 (final permission for electron.f)
```

Consider the case where each member of a group has rwx permission for files in a project directory, *Unix_validation*. If you need to restrict the deletion of files from the project directory to only the owner of the file, use the chmod command to set the sticky bit on the project directory. When the sticky bit is set on a directory, only the owner of a file, the owner of the directory, or the superuser can remove the file from the directory. The following command sets the sticky bit on the project directory, *Unix_validation*:

```
% chmod 1770   Unix_validation
```

2.2.4 Login Procedure

The first entry in the *.login* (for C Shell) or *.profile* (for the Bourne Shell) should define the PATH variable. The second entry should be "mesg -n" to prevent users from sending messages to your terminal. The third entry must be umask 027; this indicates that files created in this session have permission set to 750. Finally, **alias** commands (typically entered in the *.cshrc* file) should reference the absolute pathname. Table 18-7 provides information on recommended file protection for users.

Filename	Owner	Group	Mode	Security Relevance
.profile	username	user's group	640	environment file for Bourne-Shell
.login	username	user's group	640	read in after .cshrc
.cshrc	username	user's group	640	environment file for C-Shell
.logout	username	user's group	640	read in when you exit from login
.forward	username	user's group	600	determines where mail should be forwarded
.rhosts	username	user's group	640	user-owned file that grants remote user access to local system without password
.netrc	username	user's group	600	information used with FTP login and initialization

Table 18-7: File Protection Recommendations for Users.

18.3. Programming Methodology and Security

This section provides information on how programs and applications developed on the system may have security implications. Information is also provided on source code management systems such as System V's SCCS or Berkeley sccs. Factors that must be considered when developing privileged programs and analyzing return codes are investigated in the last part of this section.

18.3.1. Source Code Management Systems

Both System V and Berkeley UNIX provide a source code management system to help manage software projects by assigning ownership of individual modules to programmers making changes. On System V the code management system is known as Source Code Control System (SCCS). Although SCCS is not part of Berkeley UNIX, a number of Berkeley systems provide a simplified interface to SCCS called sccs. The Revision Control System (RCS) is available on Berkeley UNIX and is used for version management. SCCS is useful from the security perspective since it makes it easy to identify if a security breach involved deliberate changes to programs or applications.

SCCS provides a change history that makes it easy to evaluate each revision of the source code. It is important to use SCCS to control who is allowed to update files. By default, the list of users who can make changes to a file is empty, which implies that anyone can change an SCCS file and record changes in SCCS. Within SCCS the set of changes made is referred to as *delta*. The ***admin*** utility may be used to define a set of users authorized to make deltas to SCCS files. The following example authorizes only user1 and user2 to make deltas:

```
% admin -user1 -user2 s.telecom
```

SCCS also allows you to lock releases of a file so that no one can change them. The following example illustrates how the -f option is used to set the l flag to lock release 2 of s.telecom:

```
% admin -fl2 s.telecom
```

18.3.2. Analysis of Return Codes

Table A.4 lists all UNIX security-related functions. Virtually all calls return numerical result codes. Programs should verify that each service request reports success and if there is a failure to determine why and what the appropriate response should be. There are several reasons why a call may not be successful. For example, the *fork()* call can fail if there is insufficient space in memory or in the process table to create another process.

Programs can also test the global variable *errno* to obtain additional information on why certain calls have failed. This can help the program determine how to respond or produce a useful diagnostic message. If a security violation is detected then it is important that the diagnostic message should not disclose the nature of the security violation.

18.3.3. Privileged Programs

A number of security problems in UNIX have been caused by SUID programs. As indicated earlier, SUID programs are those that run with the UID of the owner of the file the program lives in, rather than that of the invoking process, which is usually your own UID. The owner can use the **chmod** command to make a program SUID:

```
% chmod u+s program
```

Anyone who runs this program has permissions of the file's owner. There are two important guidelines that should be considered when writing secure C programs: temporary files and reseting the UID.

Temporary files created by a program or application must not contain sensitive information. If sensitive information is to be stored, it should be in an encrypted format. Temporary files should preferably be kept in a private directory, where the umask has been previously set to 077. In general the protection for */tmp* should be set by the system administrator to 2777 (SGID) so that files can only be deleted by the owner.

It is important that the UID be reset for SUID/SGID programs before any commands are executed. One way of doing this would be:

```
int  saveid saveid = geteuid();        /* saveid = effective UID */
setuid(getuid());                      /* get the real UID  */
system("/bin/vi"); setuid(saveid);     /* reset UID to effective UID*/
```

This must be compiled using the System V compatibility library */usr/5bin/cc* instead of */bin/cc*. Some versions of the operating system do not allow regaining an effective UID/GID. In this case, use a child process to do the real UIC/GID operation.

User specified arguments must not be passed to ***system()*** or ***popen()*** - if arguments must be passed then check for special shell characters. A program should not be SUID to **root**. Also, a SUID/SGID command must not be writable by group or others. The mode for SUID commands should be 4755 or higher; the mode for SGID commands should be 2755 or higher. To prevent the execution of the strings command on the executable, the modes should be set to 4111 for SUID commands and 2111 for SGID commands. Consider using an authorization file that determines who is entitled to use a privileged program. The ***access()*** system call checks file access based on the real UID.

18.4. References and Further Reading

1. Addison, K., et al. Computer Security at Sun Microsystems, Inc. Proceedings of the 10th National Computer Security Conference, Sun Microsystems, Inc., Sep., 1987.

2. Addison, K., and Sancho, J. *Secure Networking at Sun Microsystem, Inc.* Sun Microsystems, Inc., Mountain View, CA, 1988.

3. Anderson, D.P., and Rangan, P.V. A Basis for Secure Communications in Large Distributed Systems. IEEE Symposium on Security and Privacy, April, 1987, pp. 162-172.

4. Bell, D.E., and La Padulla, L.J. *Secure Computer Systems: Unified Exposition and Multics Interpretation.* MITRE Corporation, Bedford, Massachusetts, 1976. MTR-2997 Rev. 1.

5. Branstad, D.K. "Considerations for Security in the OSI Architecture". *IEEE Network Magazine* (April 1987), 34-39.

6. *Department of Defense Trusted Computer System Evaluation Criteria.* Department of Defense, 1985. Dod 5200.28-STD.

7. Graft, D. Pabrai, M. and Pabrai, U. Methodology for Network Security Design. IEEE International Phoenix Conference on Computers and Communications (IPCCC), March, 1990.

8. Grampp, F.T., and Morris, R.H. *UNIX Operating System Security.* AT&T Bell Laboratories, 1984. Vol. 63.

9. Harrison, B.T. "Industry-Standard Computing". *HP Professional 4*, 3 (March 1990), 52-59.

10. *Information Processing Systems -- OSI Reference Model -- Part 2: Security Architecture.* ISO, 1989. Publication No. 7498 Part 2.

11. Kochan, S.G. and Wood, P.H.. *UNIX Networking.* Hayden Books, Indianapolis, IN, 1989.

12. Leffler, S.J., et al.. *The Design and Implementation of the 4.3 BSD UNIX Operating System.* Addison-Wesley, Reading, MA, 1989.

13. Lidinsky, W.P. "Data Communications Needs". *IEEE Network Magazine 4*, 2 (March 1990), 28-33".

14. *Glossary of Computer Security Terms.* NCSC, 1988. NCSC-TG-004, .Version 1.

15. Nessett, D.M. "A Systematic Methodology for Analyzing Security Threats to Interprocess Communication in a Distributed System". *IEEE Transactions on Communications COM-31*, 9 (Sep. 1983), 1055-1063.

16. Nessett. D.M. "Factors Affecting Distributed System Security". *IEEE Transactions on Software Engineering Se-13*, 2 (Feb. 1987), 233-248.

17. Noakes-Fry, K. and Hubley, M. "UNIX Security: New Solutions, Old Problems". *Datapro InfoSecurity 5*, 11 (Nov. 1989), 1-3.

18. Pabrai, U. Network Security Design for Unix Systems in a Distributed Environment, Chapter 2. Master Th., Illinois Institute of Technology,in progress: scheduled completion date, August, 1990.

19. Sobell, M.G.. *A Practical Guide to the UNIX System.* Benjamin/Cummings Publishing Company, Inc., 1989.

20. *Security Features Guide.* SUN, 1988. 800-1735-10 Rev. A.

21. Tanenbaum, A.S.. *Computer Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1988.

22. *ULTRIX Security Guide for Users.* DIGITAL Inc., 1990. Part No. PBKQA-TE.

23. Wood, P.H., and Kochan, S.G.. *UNIX System Security.* Hayden Books, Indianapolis, IN, 1985.

Appendix A

Glossary

Some of the terms in this glossary are from *Data and Computer Communications*, William Stallings, 1988 and the *Glossary of Telecommunication Terms*, FED-STD-1037A, National Communications Systems, 1986.

Access point
A networkwide reference to a device and directory at a server node running DFS

Administrative Domain
A collection of systems and subnetworks operated by a single organization or administrative authority. It may be subdivided into a number of routing domains.

AppleTalk
A network system for Macintosh computers.

Area
The group of systems which constitute a single Level 1 routing subdomain in a DECnet network.

ARPANET
Services provided include file transfer, email, and remote login. Services are supported by protocols FTP, SMTP, and TELNET. ARPANET is an advanced research and development tool for DARPA and an operational network supporting many DARPA-sponsored researchers in universities, national laboratories and industry. ARPANET has also served as a test for the development of protocols, such as TCP/IP.

Asynchronous transmission
Transmission in which each information character is individually synchronized (usually by the use of start elements and stop elements).

Attenuation
A decrease in magnitude of current, voltage, or power of a signal in transmission between points.

Balanced transmission
Signals carried on a line consisting of two conductors.

Bandwidth
The difference between the limiting frequencies of a continuous frequency spectrum.

Baseband
Transmission of signals without modulation. In a baseband local network, digital signals (1's and 0's) are inserted directly onto the cable as voltage pulses. The entire spectrum of the cable is consumed by the signal. This scheme does not allow frequency division multiplexing.

Big-endian
A format for storage or transmission of binary data in which the most significant byte (bit) comes first. The Internet's standard network byte order is big-endian.

BITNET
Purpose was to create a university network. Consists of about 175 sites in the U.S. and about 260 sites in Europe (supported via EARN). Services include file transfer, email and remote job entry.

Bridge
A store and forward device that links two local networks. A bridge must contain addressing and routing intelligence. Bridges are data link layer devices. Bridges can add/delete fields from the frame header. However, bridges cannot make changes to headers at layer 3 and above.

Broadband
The use of coaxial cable for providing data transfer by means of analog (radio-frequency) signals. Digital signals are passed through a modem and transmitted over one of the frequency bands of the cable.

Broadcast
The simultaneous transmission of data to a number of stations.

CATV
Community Antenna Television. CATV cable is used for broadband local networks.

Circuit
Virtual communication path between nodes. Circuits operate over physical lines and are the medium on which all I/O occurs. They refer to a logical stream of data between two users of the network.

Circuit Switching
A method of communicating in which a dedicated communications path is established between two devices through one or more intermediate switching nodes. Unlike paket switching, digital data are sent as a continuous stream of bits. Bandwidth is guaranteed, and delay is essentially limited to propagation time. The telephone system uses circuit switching.

Clearinghouse
In DNA, a collection of (copies of) directoreis stored on a particular node.

Coaxial cable
An electromagnetic transmission medium consisting of a center conductor and an outer, concentric conductor.

Codec
Coder-decoder. Transforms analog data into a digital bit stream (coder), and digital signals into analog data (decoder).

Collison
A condition in which two packets are being transmitted over a medium at the same time. Their interference makes both unintelligible.

Connectionless Data Transfer
A protocol for exchanging data in an unplanned fashion and without prior coordination (e.g., datagram).

Connection-oriented Data Transfer
A protocol for exchanging data in which a logical connection is established between the endpoints (e.g., virtual circuit).

Contention
The condition when two or more stations attempt to use the same channel at the same time.

Controller
One of the main components of the Ethernet physical architecture. It controls the exchange of data between the coaxial cable and the attached station.

CRC
Cyclic Redundancy Check. A numeric value derived from teh bits in a message. The transmitting station calculates a number that is attached to a message. The receiving station performs the same calculation. If the results differ, then one or more bits are in error.

CSMA
A medium access control technique for multiple-access transmission media. A station wishing to transmit first senses the medium and transmits only if the medium is idle.

CSMA/CD
A refinement of CSMA in which a station ceases to transmit if it detects collision.

CSNET
Purpose of the network is to facilitate research and advance development in Computer Science by providing a means for increased collaboration among those working in the field.

DCE
In a data station, the equipment that provides the signal conversion and coding between the DTE and the line. For example, a modem.

Datagram
In packet switching, a self-contained packet, independent of other packets, that does not require acknowledgment, and that carries information sufficient for routing from the originating DTE, without relying on earlier exchanges between the DTEs and the network.

DELNI
DIGITAL's local network interconnect product that provides eight separate network interfaces from a single transceiver tap.

DEMPR
DIGITAL's multiport repeater that provides eight Thinwire Ethernet drops from a single standard Ethernet connection.

DEQNA
DIGITAL's network adapter that connects MicroVAX and Q-bus based systems to Ethernet.

DESTA
DIGITAL's station adapter that acts as a Thinwire Ethernet transceiver. A DESTA allows you to connect a workstation with a transceiver cable to Thinwire Ethernet.

DEUNA
DIGITAL's network adapter that connects VAX and UNIBUS-based systems to Ethernet.

DFS
Allows remote files to appear as though they were locally mounted on a system. DFS uses DNS to locate files. Analogous to NFS.

DNA
DNA defines standards for protocols, interfaces, and communications functions. These standards allow various operating systems, communication devices, and computer hardware to function in a network. DNA is the basis for the design of all DECnet products.

DNS
A directory lookup service that provides a name translation, such as a DECnet node name to DECnet address. Also, used to locate DFS files.

Domain
In the Internet, a part of the naming hierarchy. A domain name consists of a sequence of names (labels) separated by periods (dots).

DTE
Equipment consisting of digital end instruments that convert user information into data signals for transmission, or reconvert the received data signals into user information. For example, a terminal.

Executor node
The node at which an NCP command actually executes. Usually the executor is the local node.

Full-duplex
Data transmission in both directions at the same time.

Gated
GATEway Daemon. A program that runs under 4.3 BSD UNIX on a gateway to allow the gateway to collect information from one autonomous system and to advertise routes to another autonomous system.

Gateway
Is a network layer device that stores and forwards packets . In general, networks connected by a gateway can differ much more than those connected by a bridge. Gateways are used to connect networks with incompatible addressing formats. Typically, gateways transfer 10,000 packets/sec.

Half-duplex
Data transmission in either direction, one direction at a time.

ISDN
A planned worldwide telecommunication service that will use digital transmission and switching technology to support voice and digital data communication.

Internet
A collection of packet-switched and broadcast networks that are connected together via gateways.

Internet protocol
An internetworking protocol that provides connectionless service across multiple packet-switched networks.

Internetworking
Communication among devices across multiple networks.

Kermit
A character oriented computer file transfer protocol developed at Columbia University.

LAN
Any physical network technology that operates at high speed (usually tens of Mbps through several Gbps) over short distances (up to a few thousand meters). Examples include Ethernet and Token-ring networks.

Level 1 router
A router which performs routing within a single area. Messages for destinations in other areas are routed to the nearest Level 2 router.

Level 2 router
A router which acts as a Level 1 router within its own area, but in addition routes messages between areas.

Line
The network management component that provides a distinct physical data path.

Little-endian
A format for storage or transmission of binary data in which the least significant byte (bit) comes first.

Local loop
Transmission path, generally twisted pair, between the individual subscriber and the nearest switching center of the public telecommunications network.

Logical link
A carrier of a single stream of full-duplex traffic between two user-level processes.

MAN
Any physical network technology that operate at high speeds (usually hundreds of Mbps through a few Gbps) over distances sufficient for a metropolitan area.

Manchester encoding
A way of encoding to get zero-DC binary waveform. In this encoding scheme, half of the bit interval is transmitted with a positive signal and the other half is transmitted with a negative signal.

Message Switching
A switching technique using a message store-and-forward system. No dedicated path is established. Rather, each message contains a destination address and is passed from source to destination through intermediate nodes. At each node, the entire message is received, stored briefly, and then passed on to the next node.

MFENET
Magnetic Fusion Energy NETwork. A network used to link laboratories and computers performing similar research.

Microwave
Electromagnetic waves in the frequency range of about 2 to 40 GHz.

Multiplexing
In data transmission, a function that permits two or more data sources to share a common transmission medium such that each source has its own channel.

Modem
Device that converts data signals to analog signals for transmission over the voice telephone network.

Name Resolution
The process of mapping a name into a corresponding address. The Internet domain name system resolves machine names into Internet addresses for those machines.

Nameserver
A system with at least one active clearinghouse

Namespace
A tree of directories starting at a root

NetBIOS
The standard interface to networks on IBM PC/compatibles.

Network Control Program
A DECnet-VAX utility that accepts terminal commands to configure, control, monitor, and test a DECnet network.

NSFNET
Function is to provide email and file transfer services as well as access to supercomputing centers and national research facilities. Consists of a T1 (1.5 Mbps) terrestrial transcontinental backbone that interconnects a number of networks such as BARRNET, Merit, SURANET, etc.

Null modem
A configuration (enabled by interconnecting leads) that allows two DTEs to signal each other.

Offered load
The actual load or traffic demand presented to a local network.

PAD
A device used with an X.25 network to provide service to asynchronous terminals.

Packet Switching
A method of transmitting messages through a communication network, in which long mesages are subdivided into short packets. The packets are then transmitted as in message switching. Usually, packet switching is more efficient and rapid than message switching.

Piggybacking
A technique used to increase acknowledgment efficiency during packet transmission. A response number is included in the message header of a user data packet being send by the receiving station.

Pipelining
A technique used to increase acknowledgment efficiency during packet transmission. Several packets are sent and the acknowledgment signifies receipt of that packet and all lower numbered packets.

Protocol Converter
Is a device used to connect dissimilar networks at the transport layer and above. It's primary function is to convert from one protocol to another.

Repeater
Is a physical layer device that amplifies electrical signals. It's function is to copy individual bits between cable segments.

Replica
In DNA, a copy of a directory stored in a particular clearinghouse.

Round robin
A medium access protocol in which everybody takes a turn.

Routed
ROUTE Daemon. A program that runs under 4.3 BSD UNIX to propagate routes among machines on a local network. Uses the RIP protocol. Pronounced "route-d."

Routing domain
A collection of systems and subnetworks which operate according to the same routng procedures and which is wholy contained within a single administrative domain.

RS-232-C
A standard interface between terminals and communications equipment. Designed for speeds up to 19.2 kbps over a maximum distance of 15 m.

RS-449/RS-422-A/423-A
A newer standard issued by the EIA to replace RS-232-C. Rs-422-A/423-A defines the electrical characteristics of the DTE-DCE interface. RS-422-A specifies balanced transmission and achieves 100 kbps at 1200 m to 10 Mbps at 12 m. RS-423-A specifies unbalanced transmission and achieves 3 kbps at 1000 m to 300 kbps at 10 m. RS-449 defines the mechanical, functional, and procedural characteristics of the new interface.

Simplex
Data transmission in one preassigned direction only.

Skulker
In DNA, the special process responsible for providing the convergence among replicas.

Slot time
An upper bound on the time it takes to detect a collision using the CSMS/CD algorithm. It is teh scheduling quantum for retransmission.

SNA
A network architecture, designed by IBM, to provide a coherent environment for loosely coupled distributed processing. Arbitrary topology of hosts and LANs are supported.

Soft link
In DNS, a form of alias, or indirect pointer, which allows a single directory entry to be reachable by more than one name.

Socket
Abstraction provided by Berkeley 4.3 BSD UNIX that allows a process to access the Internet. A process opens a socket, specifies the service desired, such as reliable strean delivery, binds the socket to a destination address, and then sends or receives data.

Subnet Address
An extension of the Internet addressing scheme that allows a site to use a single Internet address for multiple physical networks. Beyond the site using subnet addressing, routing continues as usual by dividing the destination address into an Internet portion and a local portion.

Synchronous Transmission
Data transmission in which the time of occurrence of each signal representing a bit is related to a fixed time frame.

T1 system
A digital communication system designed to handle 24 voice channels at 64 kbps each.

Tap
An analog device that permits signals to be inserted or removed from a twisted pair of coaxial cable.

TELENET
A public packet switched network using the CCITT X.25 protocols owned and operated by GTE.

TELNET
Terminal-remote host protocol developed for ARPANET. It is the Internet standard protocol for remote terminal connection service (for DECnet users, analagous to SET HOST functionality).

Terminator
A special connector used on one or both ends of an Ethernet segment that provides the 50-ohm termination resistance needed for the cable.

Token ring
A medium access control technique for rings. A token circulates around the ring. A station may transmit by seizing the token, inserting a packet onto the ring, and then retransmitting the token.

Topology
The structure, consisting of paths and switches, that provides the communications interconnection among nodes of a network.

Transceiver
A device that provides a single physical connection between standard Ethernet and Ethernet communication equipment.

Transmission medium
The physical path between transmitters and receivers in a communications system.

Twisted pair
A transmission medium consisting of two insulated wires arranged in a spiral pattern.

UltraNet
A high-speed (1000 Mbps) data link offering FDDI connections exhibiting a high degree of interconnectivity with existing networks.

Unbalanced transmision
Signals are transmitted on a single conductor. Transmitter and receiver share a common ground.

USENET
A UNIX news facility, based on the UUCP network, that provides a news bulletin board service.

UUCP
File transfer and remote command execution are the main objectives. Most host machines run the UNIX operating system. In addition to mail, much traffic is generated in response to USENET news.

Value-added network
A privately owned packet-switched network whose services are sold to the public.

Virtual circuit
A packet-switching service in which a connection (virtual circuit) is established between two stations at the start of transmission. All packets follow the same route. Thus, packets do not carry a complete address and always arrive in sequence.

X.21
A network access standard for connecting stations to a circuit-switched network. Includes OSI layers 1-3 functionality.

X.25
The CCITT three-layered interface architecture for packet switching connecting a DTE to a DCE.

X.75
An internetworking protocol that provides virtual circuit service across multiple X.25 networks.

X.400
CCITT developed the X.400 Family of Standards for MHS. The standard defines the MHS model consisting of user agents and message transfer agents, discusses naming and addressing, defines interpersonal messaging and message transfer services.

X.500
CCITT standard for directories for X.400 networks.

XMI
a 100-Mbps bus used to connect CPUs, BI buses and memory on VAX 6200 series of parallel processors.

Yellow Pages

A distributed data base lookup service that maintains a set of data bases and propagates updated data bases among the systems on the network ensuring consistency.

This Page Intentionally Left Blank

Appendix B

List of Acronyms

Acronym	Definition
AFI	Authority and Format Identifier.
ANSI	American National Standards Institute
ARP	Address Resolution Protocol
ARPANET	Advanced Research Projects Agency NETwork
ASCII	American Standard Code for Information Interchange
ASN	Abstract Syntax Notation
BBN	Bolt Beranek and Newman Communications Corporation
BITNET	Because It's Time NETwork
CATV	Community Antenna TV
CCITT	International Consultative Committee on Telegraphy and Telephony
CI	Computer Interconnect
CISC	Complex Instruction Set Computer
CLNS	Connectionless-mode Network Service
CONS	Connection-mode Network Service
CMIP	Common Management Information Protocol
CRC	Cyclic Redundancy Check
CSMA/CD	Carrier-Sense Multiple Access with Collision Detection
CSNET	Computer Science NETwork

Acronym	Definition
DAP	Data Access Protocol
DARPA	Defense Advanced Research Projects Agency
DCE	Data Circuit-Terminating Equipment
DDCMP	Digital Data Communications Message Protocol
DEBNA	Digital Ethernet BI Network Adapter
DELNI	Digital Ethernet Local Network Interconnect
DELUA	Digital Ethernet LAN UNIBUS Adapter
DEMPR	Digital Ethernet MultiPort Repeater
DEQNA	Digital Ethernet LAN Q-bus Adapter
DESPR	Digital Ethernet Single-Port Repeater
DESTA	Digital Ethernet Station Adapter
DESVA	Digital Ethernet Small VAX Adapter
DFS	Distributed File Service
DNA	Digital Network Architecture
DNS	Distributed Name Service
DQS	Distributed Queueing Service
DSP	Domain Specific Part
DSAP	Destination Service Access Point
DTE	Data Terminal Equipment
EARN	European Academic Research Network
EBCDIC	Extended Binary Coded Decimal Interchange Code
EIA	Electronics Industries Association
EMA	Enterprise Management Architecture
FAL	File Access Listener
FDM	Frequency Division Multiplexing
FDDI	Fiber Distributed Data Interface
FIPS	Federal Information Processing Standard
FTAM	File Transfer, Access, and Management
FTP	File Transfer Protocol
Gbps	Giga bits per second
GOSIP	Government Open Systems Interconnect Profile

Acronym	Definition
HDLC	High-Level Data Link Control
HPPI	High-Performance Parallel Interface
HSC	Hierarchical Storage Controller
ICMP	Internet Control Message Protocol
IDI	Initial Domain Identifier
IDP	Initial Domain Part
IEEE	Institute of Electrical and Electonics Engineers
IGRP	Interior Gateway Routing Protocol
IMP	Interface Message Processor
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
JANET	Joint Academic NETwork (Great Britain)
LAN	Local Area Network
LAP-B	Link Access Protocol - Balanced
LAT	Local Area Transport protocol
LAVC	Local Area Vax Cluster
LLC	Logical Link Control
LSP	Link State PDU
MAC	Medium Access Control
MAN	Metropolitan Area Network
MAP	Manufacturing Automated Protocol
Mbps	Millions of (Mega) bits per second
MFENET	Magnetic Fusion Energy NETwork
MHS	Messahe Handling Systems
MICE	Management Information, Control, and Exchange
MOTIS	Message-Oriented Text Interchange Systems
MTU	Maximum Transfer Unit
NBS	National Bureau of Standards
NCS	Network Computing System
NetBIOS	Network Basic Input Output System
NCL	Network Control Language

Acronym	Definition
NCP	Network Control Program
NeWS	Network Extensible Window System
NFS	Network File System
NIC	Network Information Center
NPDU	Network layer Protocol Data Unit
NSAP	Network Service Access Point
NSDU	Network Service Data Unit
NSF	National Science Foundation
NSFNET	National Science Foundation NETwork
NSP	Network Services Protocol
OSI	Open Systems Interconnection
OSI/RM	Open Systems Interconnection Reference Model
PAD	Packet Assembler Disassembler
PBX	Private Branch Exchange
PCSA	Personal Computing Systems Architecture
PDN	Public Data Network
PDU	Protocol Data Unit
PING	Packet InterNet Groper
POTS	Plain, Ordinary Telephone System
PSN	Packet Switch Node
RARP	Reverse Address Resolution Protocol
RIP	Routing Information Protocol
RISC	Reduced Instruction Set Computer
RMS	Record Management Services
RPC	Remote Procedure Call
SAA	System Application Architecture
SCA	System Communication Architecture
SCA	Subsidiary Communications Authorization
SMTP	Simple Mail Transfer Protocol
SNA	Systems Network Architecture
SNAP	Subnetwork Access Protocol
SNMP	Simple Network Management Protocol

Acronym	Definition
SPARC	Scalable Processor ARChitecture
SSAP	Source Service Access Point
TOP	Technical and Office Protocols
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
UDP	User Datagram Protocol
UUCP	UNIX to UNIX CoPy
VAN	Value-Added Network
VMTP	Versatile Message Transaction Protocol
VTAM	Virtual Telecommunications Access Method
VTP	Virtual Terminal Protocol
XDR	eXternal Data Representation
XNS	Xerox Network Standard

This Page Intentionally Left Blank

Appendix C

Fermilab Internet Addresses

The following is the list of Internet addresses assigned to various hosts at Fermilab. Note that Fermilab is a Class B Internet network with the first two fields of its address beginning with 131.225. For more information refer to the Internet chapter.

IP ADDRESSES SORTED BY NODE NAME

Host or domain name	Internet address
fnal	server = fnal.gov
DCDMBL	131.225.100.109
DEVL	131.225.8.105
Rlmt2	131.225.45.12
acpm1	131.225.8.3
acpm2	131.225.8.10
adcalc	131.225.8.104
adcprint	131.225.100.70
adcs00	131.225.100.71
adcs01	131.225.100.72
adcs02	131.225.100.73
adms00	131.225.100.100
adms01	131.225.100.101
adms02	131.225.100.102
adms03	131.225.100.103
adms04	131.225.100.104
adms05	131.225.100.105
adms06	131.225.100.106
adms07	131.225.100.107
adms08	131.225.100.108
adsuna	131.225.8.100
b0gwf	131.225.102.22
b0host	131.225.100.31

b0serv	131.225.8.129
b0sga	131.225.100.30
b0vs02	131.225.8.135
bison	131.225.32.2
boffo	131.225.18.175
boxcar	131.225.100.201
bsndbg	131.225.32.1
calvin	131.225.18.178
cddec1	131.225.85.4
cdsgi1	131.225.85.1
cdsun1	131.225.85.2
cdsun2	131.225.85.3
chip	131.225.13.3
cisc1	131.225.8.200
citoh1	131.225.85.14
credit	131.225.8.151
d0hsa	131.225.100.37
d0sgi1	131.225.100.46
d0tba	131.225.100.38
daffy	131.225.13.1
dale	131.225.13.4
dan	131.225.8.209
dcdadt	131.225.100.45
dcdfjn	131.225.100.43
dcdkc	131.225.100.44
dcdrjh	131.225.100.42
digi1	131.225.105.5
digi2	131.225.105.6
dipole	131.225.18.174
don	131.225.8.213
elmer	131.225.13.2
eros	131.225.8.154
esfnal	131.225.17.3
espc9	131.225.16.10
fasic1	131.225.90.1
fig	131.225.18.173
flibber	131.225.18.172
fnacc5	131.225.102.2
fnacc6	131.225.102.3
fnaccc	131.225.8.225
fnaccd	131.225.8.226
fnacp	131.225.8.1
fnacp1	131.225.8.2
fnacp5	131.225.102.4
fnacpbat	131.225.8.4
fnal07	131.225.100.41
fnal27	131.225.8.217
fnalb	131.225.8.77
fnalc	131.225.8.78
fnalf	131.225.102.1

fnalh	131.225.8.43
fnali	131.225.8.45
fnalj	131.225.8.46
fnalm	131.225.102.5
fnalm	131.225.8.44
fnalna	131.225.102.7
fnalnb	131.225.102.8
fnalnc	131.225.102.9
fnalnd	131.225.102.10
fnalne	131.225.102.11
fnalnf	131.225.102.12
fnalng	131.225.102.13
fnalnh	131.225.102.14
fnalni	131.225.102.15
fnalnj	131.225.102.16
fnalnk	131.225.102.17
fnalnl	131.225.102.18
fnalnm	131.225.102.19
fnalnn	131.225.102.20
fnalno	131.225.102.21
fnalo	131.225.102.6
fnbit	131.225.8.120
fnboot	131.225.85.10
fnccf	131.225.8.39
fncnfc	131.225.34.3
fnda01	131.225.32.14
fnda03	131.225.32.13
fnda05	131.225.32.18
fnda07	131.225.32.15
fnda08	131.225.32.16
fnda09	131.225.32.9
fnda12	131.225.32.19
fnda13	131.225.32.20
fnda14	131.225.32.7
fnda15	131.225.32.6
fnda17	131.225.32.10
fnda18	131.225.32.17
fnda22	131.225.32.21
fnda23	131.225.32.8
fndaqt	131.225.32.5
fndaqu	131.225.32.11
fndaqv	131.225.32.3
fndaqw	131.225.32.4
fndaua	131.225.32.12
fndaub	131.225.32.23
fndgu1	131.225.46.2
fndgu2	131.225.46.3
fndgux	131.225.46.1
fnext	131.225.17.2
fngate	131.225.8.145

fnint	131.225.17.1
fnl26a	131.225.101.1
fnl26b	131.225.101.2
fnl26c	131.225.101.3
fnl26d	131.225.101.4
fnl26e	131.225.101.5
fnl26f	131.225.101.6
fnl26g	131.225.101.7
fnl26h	131.225.101.8
fnl26i	131.225.101.9
fnmac6	131.225.8.56
fnmark	131.225.8.56
fnmfe	131.225.17.150
fnnet	131.225.100.200
fnnet1	131.225.17.10
fnpspa	131.225.40.11
fnpspb	131.225.40.12
fnpspc	131.225.40.13
fnpw01	131.225.8.172
fnpw02	131.225.8.173
fnpw03	131.225.8.174
fnpw04	131.225.8.175
fnpw05	131.225.8.176
fnpw06	131.225.8.177
fnpw07	131.225.8.75
fnsgia	131.225.63.1
fnsgib	131.225.63.2
fnsgic	131.225.63.3
fnsgid	131.225.63.4
fnsgie	131.225.63.5
fnsgif	131.225.63.6
fnsgig	131.225.63.7
fnsgih	131.225.63.8
fnsgii	131.225.63.9
fnsgij	131.225.63.10
fnsgik	131.225.63.11
fnsgil	131.225.63.12
fnsgim	131.225.63.13
fnsgin	131.225.63.14
fnsgio	131.225.63.15
fnsgip	131.225.63.16
fnsgiq	131.225.63.17
fnsgir	131.225.63.18
fnsgis	131.225.63.19
fnsgit	131.225.63.20
fnsgiu	131.225.63.21
fnsgiv	131.225.63.22
fnsgiw	131.225.63.23
fnsgix	131.225.63.24
fnsgiy	131.225.63.25

fnts06	131.225.22.62
fntst1	131.225.8.83
fnucd	131.225.101.101
fnusg1	131.225.85.11
fnusg2	131.225.85.12
fnusg3	131.225.85.13
fnusg4	131.225.85.14
fnusg5	131.225.85.15
fnusg6	131.225.85.16
fnusg7	131.225.85.17
fnusg8	131.225.85.18
fnusga	131.225.85.19
fnusgb	131.225.85.20
fnusgx	131.225.100.99
funix	131.225.100.97
hepmac1	131.225.8.188
hepnet	131.225.100.1
hobbes	131.225.18.180
iris	131.225.8.53
irsi1	131.225.8.54
jack	131.225.8.211
jeff	131.225.8.210
linac	131.225.18.176
lludev	131.225.8.170
lluopr	131.225.8.84
lluptf	131.225.8.203
lmt	131.225.45.11
lmt3	131.225.45.13
localhost	127.0.0.1
mars	131.225.8.157
mdtf	131.225.45.1
mdtf00	131.225.45.2
mdtf01	131.225.45.3
mdtf04	131.225.45.7
mdtf05	131.225.45.8
mdtf06	131.225.45.4
mdtf07	131.225.45.5
mdtf08	131.225.45.6
mdtf09	131.225.22.101
mdtf10	131.225.22.102
mdtf11	131.225.45.9
mdtf12	131.225.45.10
mdtfl2	131.225.22.100
mtf12	131.225.46.100
myrtle	131.225.18.171
nagy	131.225.13.6
ncdx1	131.225.45.15
panda	131.225.101.200
peter	131.225.8.212
pigdev	131.225.22.150

ps	131.225.8.98
quad	131.225.18.177
sext	131.225.18.181
sexton	131.225.11.1
sgi1	131.225.51.71
sgi2	131.225.51.72
sgi3	131.225.51.73
sgi4	131.225.51.74
sgi5	131.225.51.75
sgi6	131.225.51.76
sgi7	131.225.51.77
sgi8	131.225.51.78
sja	131.225.8.159
sprs6a	131.225.100.34
thomas	131.225.13.5
village1	131.225.100.76
village2	131.225.19.3
x19_1	131.225.45.14
xxxxx	131.225.17.150
xyp2	131.225.105.1
xyp3	131.225.105.2
xyp4	131.225.105.3
xyp5	131.225.105.4
zippy	131.225.18.179

IP ADDRESSES SORTED BY SUBDOMAIN
(Third field of address)

Host or domain name	Internet address
fnal	server = fnal.gov
localhost	127.0.0.1
fnacp	131.225.8.1
fnacp1	131.225.8.2
acpm1	131.225.8.3
fnacpbat	131.225.8.4
acpm2	131.225.8.10
fnccf	131.225.8.39
fnalh	131.225.8.43
fnalm	131.225.8.44
fnali	131.225.8.45
fnalj	131.225.8.46
iris	131.225.8.53
irsi1	131.225.8.54
fnmac6	131.225.8.56
fnmark	131.225.8.56
fnpw07	131.225.8.75

fnalb	131.225.8.77
fnalc	131.225.8.78
fntst1	131.225.8.83
lluopr	131.225.8.84
ps	131.225.8.98
adsuna	131.225.8.100
adcalc	131.225.8.104
DEVL	131.225.8.105
fnbit	131.225.8.120
b0serv	131.225.8.129
b0vs02	131.225.8.135
fngate	131.225.8.145
credit	131.225.8.151
eros	131.225.8.154
mars	131.225.8.157
sja	131.225.8.159
lludev	131.225.8.170
fnpw01	131.225.8.172
fnpw02	131.225.8.173
fnpw03	131.225.8.174
fnpw04	131.225.8.175
fnpw05	131.225.8.176
fnpw06	131.225.8.177
hepmac1	131.225.8.188
cisc1	131.225.8.200
lluptf	131.225.8.203
dan	131.225.8.209
jeff	131.225.8.210
jack	131.225.8.211
peter	131.225.8.212
don	131.225.8.213
fnal27	131.225.8.217
fnaccc	131.225.8.225
fnaccd	131.225.8.226
sexton	131.225.11.1
daffy	131.225.13.1
elmer	131.225.13.2
chip	131.225.13.3
dale	131.225.13.4
thomas	131.225.13.5
nagy	131.225.13.6
espc9	131.225.16.10
fnint	131.225.17.1
fnext	131.225.17.2
esfnal	131.225.17.3
fnnet1	131.225.17.10
xxxxx	131.225.17.150
fnmfe	131.225.17.150
myrtle	131.225.18.171
flibber	131.225.18.172

fig	131.225.18.173
dipole	131.225.18.174
boffo	131.225.18.175
linac	131.225.18.176
quad	131.225.18.177
calvin	131.225.18.178
zippy	131.225.18.179
hobbes	131.225.18.180
sext	131.225.18.181
village2	131.225.19.3
fnts06	131.225.22.62
mdtf12	131.225.22.100
mdtf09	131.225.22.101
mdtf10	131.225.22.102
pigdev	131.225.22.150
bsndbg	131.225.32.1
bison	131.225.32.2
fndaqv	131.225.32.3
fndaqw	131.225.32.4
fndaqt	131.225.32.5
fnda15	131.225.32.6
fnda14	131.225.32.7
fnda23	131.225.32.8
fnda09	131.225.32.9
fnda17	131.225.32.10
fndaqu	131.225.32.11
fndaua	131.225.32.12
fnda03	131.225.32.13
fnda01	131.225.32.14
fnda07	131.225.32.15
fnda08	131.225.32.16
fnda18	131.225.32.17
fnda05	131.225.32.18
fnda12	131.225.32.19
fnda13	131.225.32.20
fnda22	131.225.32.21
fndaub	131.225.32.23
fncnfc	131.225.34.3
fnpspa	131.225.40.11
fnpspb	131.225.40.12
fnpspc	131.225.40.13
mdtf	131.225.45.1
mdtf00	131.225.45.2
mdtf01	131.225.45.3
mdtf06	131.225.45.4
mdtf07	131.225.45.5
mdtf08	131.225.45.6
mdtf04	131.225.45.7
mdtf05	131.225.45.8
mdtf11	131.225.45.9

mdtf12	131.225.45.10
lmt	131.225.45.11
Rlmt2	131.225.45.12
lmt3	131.225.45.13
x19_1	131.225.45.14
ncdx1	131.225.45.15
fndgux	131.225.46.1
fndgu1	131.225.46.2
fndgu2	131.225.46.3
mtf12	131.225.46.100
sgi1	131.225.51.71
sgi2	131.225.51.72
sgi3	131.225.51.73
sgi4	131.225.51.74
sgi5	131.225.51.75
sgi6	131.225.51.76
sgi7	131.225.51.77
sgi8	131.225.51.78
fnsgia	131.225.63.1
fnsgib	131.225.63.2
fnsgic	131.225.63.3
fnsgid	131.225.63.4
fnsgie	131.225.63.5
fnsgif	131.225.63.6
fnsgig	131.225.63.7
fnsgih	131.225.63.8
fnsgii	131.225.63.9
fnsgij	131.225.63.10
fnsgik	131.225.63.11
fnsgil	131.225.63.12
fnsgim	131.225.63.13
fnsgin	131.225.63.14
fnsgio	131.225.63.15
fnsgip	131.225.63.16
fnsgiq	131.225.63.17
fnsgir	131.225.63.18
fnsgis	131.225.63.19
fnsgit	131.225.63.20
fnsgiu	131.225.63.21
fnsgiv	131.225.63.22
fnsgiw	131.225.63.23
fnsgix	131.225.63.24
fnsgiy	131.225.63.25
cdsgi1	131.225.85.1
cdsun1	131.225.85.2
cdsun2	131.225.85.3
cddec1	131.225.85.4
fnboot	131.225.85.10
fnusg1	131.225.85.11
fnusg2	131.225.85.12

fnusg3	131.225.85.13
fnusg4	131.225.85.14
citoh1	131.225.85.14
fnusg5	131.225.85.15
fnusg6	131.225.85.16
fnusg7	131.225.85.17
fnusg8	131.225.85.18
fnusga	131.225.85.19
fnusgb	131.225.85.20
fasic1	131.225.90.1
hepnet	131.225.100.1
b0sga	131.225.100.30
b0host	131.225.100.31
sprs6a	131.225.100.34
d0hsa	131.225.100.37
d0tba	131.225.100.38
fnal07	131.225.100.41
dcdrjh	131.225.100.42
dcdfjn	131.225.100.43
dcdkc	131.225.100.44
dcdadt	131.225.100.45
d0sgi1	131.225.100.46
adcprint	131.225.100.70
adcs00	131.225.100.71
adcs01	131.225.100.72
adcs02	131.225.100.73
village1	131.225.100.76
funix	131.225.100.97
fnusgx	131.225.100.99
adms00	131.225.100.100
adms01	131.225.100.101
adms02	131.225.100.102
adms03	131.225.100.103
adms04	131.225.100.104
adms05	131.225.100.105
adms06	131.225.100.106
adms07	131.225.100.107
adms08	131.225.100.108
DCDMBL	131.225.100.109
fnnet	131.225.100.200
boxcar	131.225.100.201
fnl26a	131.225.101.1
fnl26b	131.225.101.2
fnl26c	131.225.101.3
fnl26d	131.225.101.4
fnl26e	131.225.101.5
fnl26f	131.225.101.6
fnl26g	131.225.101.7
fnl26h	131.225.101.8
fnl26i	131.225.101.9

fnucd	131.225.101.101
panda	131.225.101.200
fnalf	131.225.102.1
fnacc5	131.225.102.2
fnacc6	131.225.102.3
fnacp5	131.225.102.4
fnalm	131.225.102.5
fnalo	131.225.102.6
fnalna	131.225.102.7
fnalnb	131.225.102.8
fnalnc	131.225.102.9
fnalnd	131.225.102.10
fnalne	131.225.102.11
fnalnf	131.225.102.12
fnalng	131.225.102.13
fnalnh	131.225.102.14
fnalni	131.225.102.15
fnalnj	131.225.102.16
fnalnk	131.225.102.17
fnalnl	131.225.102.18
fnalnm	131.225.102.19
fnalnn	131.225.102.20
fnalno	131.225.102.21
b0gwf	131.225.102.22
xyp2	131.225.105.1
xyp3	131.225.105.2
xyp4	131.225.105.3
xyp5	131.225.105.4
digi1	131.225.105.5
digi2	131.225.105.6

This Page Intentionally Left Blank

Appendix D

BITNET Nodes

Critical off-site BITNET nodes:

UICVM	FNALVM	CODVM1	FNBIT

Fermilab BITNET nodes:

FNALB	FNALC	FNALE	FNALF
FNALH	FNALI	FNALJ	WISCPSLC
FNALBSN	FNBOOT	BRFAPESP	FNALAD
FNALMDTF	FNMFE	FNALNET	FNALDBG
FNACP	FNALB0	FNFAF	FNAL26
FNAL27	SLACASP	FNALM	FNALO
FNALNA	FNALNB	FNALNC	FNALND
FNALNE	FNALNF	FNALNG	FNALNH
FNALNI	FNALNJ	FNALNK	FNALNL
FNALNM	FNALNN	FNALNO	FNPW01
FNPW02	FNPW03	FNPW04	FNPW05
FNPW06	FNPW07	FNBCD1	FNPW10
FNPW11	FNPW12	FNPW13	FNMU01
FNMU02	FNMONT	FNUSG1	FNUSG2
FNUSG3	FNUSG4	FNUSG5	FNUSG6
FNUSG7	FNUSG8	FNUSG9	FNUSGA
FNUSGB			

This Page Intentionally Left Blank

Appendix E

MFENET

The Magnetic Fusion Energy NETwork (MFENET) was created to support magnetic fusion energy research. It links laboratories and computers doing similar research. This research often requires the use of the CRAY computers located at Lawrence Livermore National Laboratory. MFENET provides the capability to access these CRAY systems.

Several DOE sites have access to MFENET. At each site, however, there is only one host designated as the local MFENET gateway. It is the only system that has access to the full range of MFENET commands. At Fermilab the local MFENET gateway resides on node FNMFE.

MFENET uses the Energy Sciences network (ESnet) to carry its traffic. The ESnet backbone is a high speed backbone providing DECnet connectivity between DOE sites. The rate of T1 links making up the ESnet backbone is approximately 1.5 Mbps. The ESnet backbone supports TCP/IP as well as DECnet over the same lines. The current MFENET topology is shown in Figure E-1.

MFENET has migrated to TCP/IP as its main transport protocol. Users on local DECNET hosts can login to the MFENET hosts.

E.1. FNMFE

FNMFE is a MicroVax II that runs DECNET and the MFENET networking utilities. In order for a prospective user to gain access to the full set of MFENET commands they would have to:

1. **$ SET HOST FNMFE**
2. Login to a valid account on FNMFE

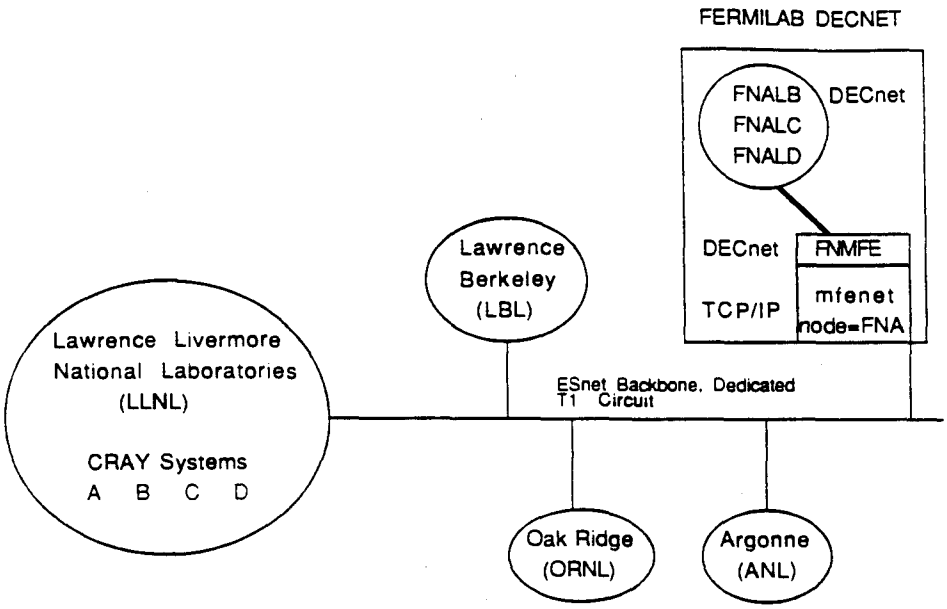


Figure E-1: MFENET Topology

E.2. Commands Available Only On MFENET Hosts

Once the user is logged onto an MFENET host, he will have access to the full set of MFENET utilities. The following commands are only available on MFENET hosts.

NETOUT: This command allows the user to send files to the top level directory (SYS$LOGIN area) of any user on an MFENET host. The following example illustrates how to send the local file, XYZ.DAT, to MFENET user SMITH at ARGONNE.

```
$ NETOUT/user=SMITH/dest=ANL XYZ.DAT
```

TELL: This command initiates a mail-like utility that allows the user to send and receive mail from other MFENET users. The command sequence shown below will start the process of reading new mail:

```
$ TELL
```

Sending outbound mail can be more conveniently accomplished using regular VMS mail.

MFEMSG: This command is part of the mail utility. It is used to copy, delete, or read messages. Note that VMS mail is considerably more convenient.

HELP @MFENET:
Allows users to obtain information on the usage of various MFENET commands.

HELP mfe_hosts: Provides a list of host names.

E.3. Commands Available from the FNAL VAX Cluster

Some MFENET commands can be used from the FNAL VAX cluster or other local systems. These commands are listed below:

NETTY: This command is the MFENET equivalent of "SET HOST" or "Telnet". The example shown below would allow a user on the FNAL Vax cluster or FNMFE to login to the MFENET node called LBL:

```
$ NETTY LBL
```

VMS MAIL: Any system that is running the JNET software should be able to send mail to users on MFENET. The format that is used is: MFENET%"username@node.MFENET". The following example illustrates how a local user would send mail to user JONES on MFENET node LBL:

```
$ MAIL
MAIL> SEND
To: MFENET%"JONES@LBL.MFENET"
Subj ect: Important discovery!!
    .
    .
    .
```

E.4. The Future of MFENET

Since the migration of MFENET away from a proprietary routing protocol to TCP/IP, it seems likely that users will find it increasingly more convenient to use standard TCP/IP network utilities (FTP and TELNET) rather than the MFENET supplied utilities. This development will pick up much greater momentum when the CRAY systems at Livermore develop both client and server TCP/IP capabilities. At present the CRAYs are only equipped to act as TCP/IP servers and not as clients. TCP/IP client functionality would appear to be about 6-12 months away from completion.

This Page Intentionally Left Blank

Index