Independent Information
Technology Training Pty Ltd

Welcome!

It gives us great pleasure to welcome you as a delegate to our practical and interactive *Hands-on TCP/IP Networks* course. We will strive to meet your needs and hopefully exceed your expectations.

During the course, you will have the opportunity to discuss pertinent issues relating to you and your company, both with your instructor Richard Sharpe and with the other delegates. Richard is here to answer your questions and advise you on all aspects of TCP/IP networking. We hope you take full advantage of Richard's knowledge and maximise your benefits from the course.

Please make notes in the workbook provided, as you will find it to be a valuable reference source for your office.

Thank you for choosing IIT Training - enjoy the course!

Yours sincerely,

Lisbeth Larsson
Managing Director

Independent Information Technology Training Pty Limited
Level 10, 35 Pitt Street, Sydney, NSW 2000, Australia
PO Box R817, Royal Exchange, Sydney, NSW 2000, Australia

ACN 001 555 700
Phone: +61 2 252 2844
Fax: +61 2 247 1048

Company Profile

IIT Training Pty Limited, is a wholly owned Australian company. Its mission is to provide excellence in high end technology training to Australian users of information technology.

Since its inception, the company has developed a range of IT courses, acquired an extensive network of local consultant instructors and proven its new technology training concepts in the field. The concepts include the provision of theory training, using the theory in practical sessions/workshops and finally the revision through private challenge (questions) relating to the subject matter.

All IIT Training courses give IT professionals the information they really need. These specifically designed courses provide solutions to many problems and concerns that plague the real world of PCs and associated networks. The practical nature of the courses supports the concepts of hands-on or practical involvement. Additionally, by working on the latest technological equipment available in Australia, the participant becomes aware of its operational prowess. Information gained on these courses can therefore assist an organisation in their equipment decision making process.

The company's objective to provide a high end superior training service to IT professionals is achieved through excellent local instructors with a unique blend of technical and presentation skills and a practical knowledge of the IT industry. Further, high quality, locally produced and regularly updated course materials along with equipment intensive practicals ensure maximum skills retention is achieved.

IIT Training have relations with major equipment suppliers to facilitate the latest in technology learning. While the company is independent of these equipment suppliers, its links with them provide IT staff with the latest on a broad equipment front.

IIT Training markets its courses through traditional avenues and provides courses in both the public and private arenas. The value of public courses is in the wide cross-section of participants in a course, which adds to the real world of problem solving. An on-site or private course is better able to focus on issues as they relate to the corporate operation.

Whether a customer chooses to attend and participate in a public course or organise a private course, they can be assured of a flexible solution to their quality training needs.

IIT Training will provide independent practical and objective information on products, protocols, methodologies and concepts.

About Your Instructor

Richard Sharpe has more than 18 years experience in the IT industry. His career started with the Department of Defence in Canberra, as a UNIVAC systems programmer. He subsequently moved to Prime Computer where he worked on X.25 networks.

After a few years in Canberra, Richard moved to Austek Microsystems in Adelaide, where he was responsible for Austek's worldwide network, using both DECnet and TCP/IP. While there, he connected Austek to the Internet and developed considerable experience with TCP/IP.

He subsequently joined Digital Equipment, where he developed and taught a variety of courses involving UNIX and TCP/IP, as well as installing networks of UNIX systems for customers. Richard is currently an independent consultant and spends his time connecting people to the Internet, developing Web pages and developing CGI scripting based applications.

He has experience in the areas of data communications, UNIX system and network programming (he is the author of SMBlib, a freely available SMB library for UNIX systems), troubleshooting, network design and implementation, Web server management, Web page authoring and CGI scripting.

Hands-on TCP/IP Networks

Presented by
Richard Sharpe

On behalf of
IIT Training Pty Ltd

Table of Contents

1

TCP/IP Network Fundamentals

iit TRAINING

TCP/IP Network Fundamentals

Introduction

As the most widely implemented non-proprietary protocol Transmission Control Protocol/Internet Protocol (TCP/IP) holds a prominent position in networking technology.

From its foundations in the Defence Advanced Research Projects Agency (DARPA) multi-vendor networks, to its prominence on the National Science Foundation (NSF) managed Internet and its emergence in 1983/84 to Australia, TCP/IP holds pride of place.

During the past few years it has become normal practice to define communication architectures in the context of the ISO OSI Reference Model. As TCP/IP and OSI developed at about the same time there are some similarities as well as differences.

This section examines the fundamentals of TCP/IP and related networking concepts and identifies the main components of OSI with respect to TCP/IP.

Objectives

At the conclusion of this section delegates will be able to describe the concepts of:

- Connection oriented and connectionless services
- Protocol layers
- Protocol multiplexing and demultiplexing
- OSI/RM
- The TCP/IP and OSI/RM relationship.

Notes:

Some Basic Concepts

Before we start to look at TCP/IP, we need to establish a common vocabulary of networking concepts and terminology that is used in reference to TCP/IP. These include:

- Protocol Layering
- Peer-to-peer Networking
- Encapsulation
- Fragmentation
- Multiplexing/Demultiplexing
- Connection-Oriented Services
- Connectionless Services
- Delivery
- Windows

Notes:

Protocol Layering

Protocol layering is an important networking concept as it has become common to describe the functionality of networks in terms of layers. The international standard for protocol layering is called the International Standards Organisation (ISO) Open Systems Interconnection (OSI) Reference Model.

For the moment, consider a simple four-layer network. At the bottom we have a layer which provides us access to the physical network - the sort of thing which Ethernet or X.25 provides. The function of this layer is to exchange *frames* with the same layer on the remote host.

The next layer is the internet or network layer, which provides an addressing and routing structure to allow us to exchange data across multiple, interconnected networks. The unit of data here is the *datagram*.

Above this we have the transport mechanism, which provides us with a reliable, error free path between the two hosts. It looks to the layer above like a point-to-point circuit connecting the two together. The unit of data here is the *packet*.

Finally we have the application, which needs to exchange *messages* with its opposite number on the remote host.

Each layer is only visible to the ones on either side. Each layer contains a definable set of functionality and provides a suitable interface to that functionality alone.

Notes:

Layer *n* at destination receives same object as layer *n* at source.

Makes designing, describing and building communications systems easier.

Each layer provides a specific subset of functions and definitions of the interface to the layer.

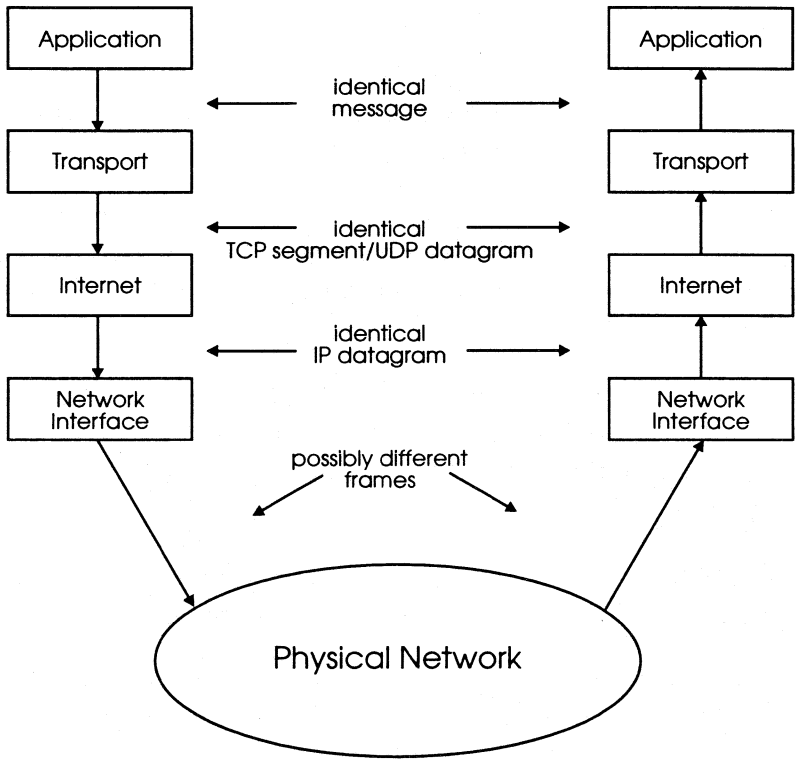


Figure 1.1

Notes:

Encapsulation

Each protocol layer takes the *packet* from the level above and *encapsulates* it into the data area of its own packet. This is a very important concept which underpins the working of all layered protocol stacks. When the frame reaches the remote host, each layer de-encapsulates the packet for the next layer up.

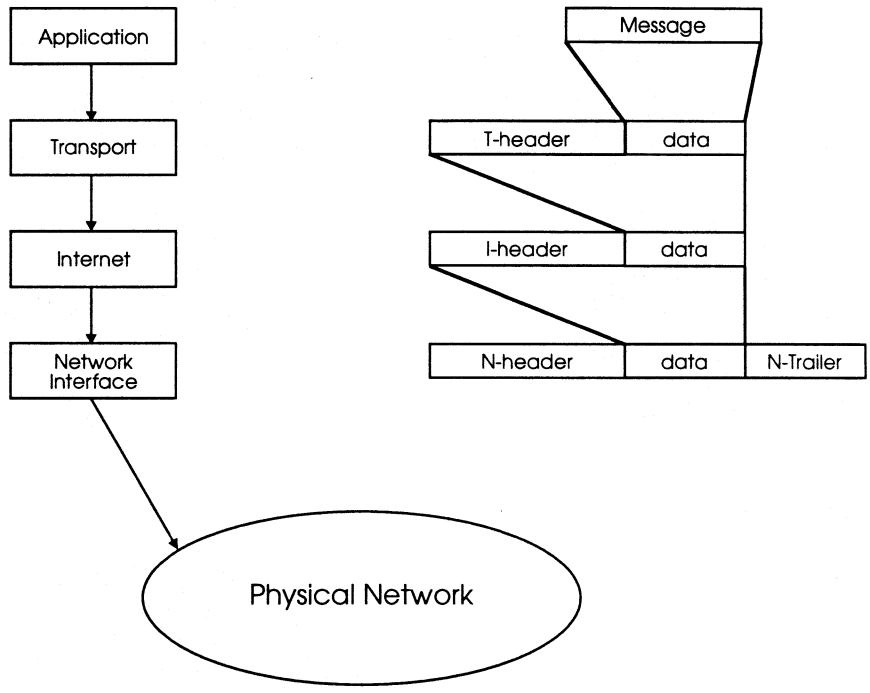


Figure 1.2

Notes:

Fragmentation

The internet or network layer and above are *software* layers - and so the size of their packets is limited only by memory availability. The physical network imposes a *hard* limit on frame size. This limit is known as the *MTU* - Maximum Transmission Unit. This could be as low as 128 octets (some X.25 systems).

It may be necessary to break, or *fragment*, a transport packet into several bits before reassembly on the other side when MTU across the network are different. However, this is to be avoided under most circumstances because of the impact of packet losses (see later).

The remote internet layer is responsible for reassembling the fragments before sending them up to its transport layer.

Hyperchannel	65535 octets
16Mbps Token Ring (IBM)	17914 octets
4Mbps Token Ring (802.5)	4464 octets
FDDI	4352 octets
Ethernet V2	1500 octets
IEEE 802.3/802.2	1492 octets
X.25	576 octets
PPP (low delay)	296 octets

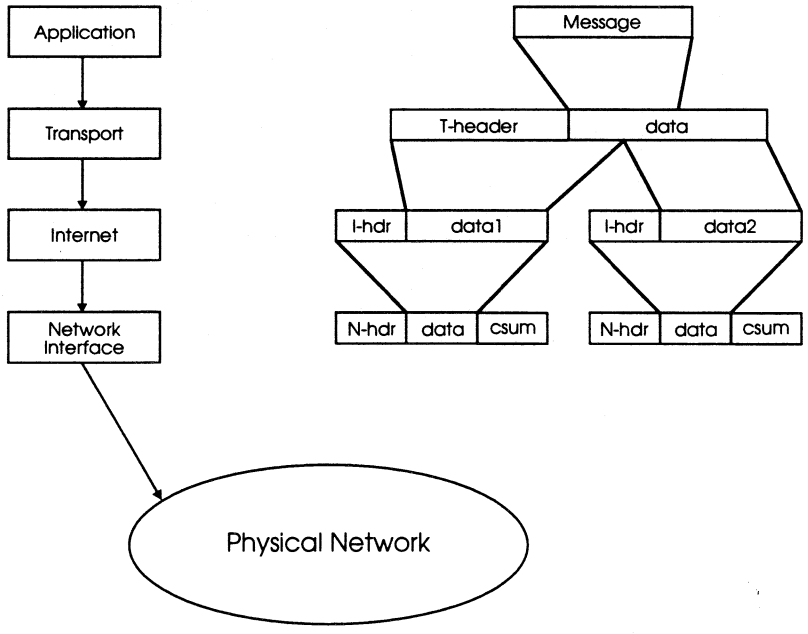


Figure 1.3

Multiplexing/Demultiplexing

We use the term *multiplexing* to define many activities in communications but it usually refers to the combining of a number of datastreams into a single stream. Packet switching is an example of multiplexing as each packet could come from a completely different source or be heading for a different destination and yet they still travel down the same route.

The reverse of multiplexing is *demultiplexing*. This term has a special meaning within the TCP/IP world as it is used to describe the mechanism by which a protocol layer is able to deliver messages to multiple upper layers. This means, for example, that more than one application can use the communications system at a time.

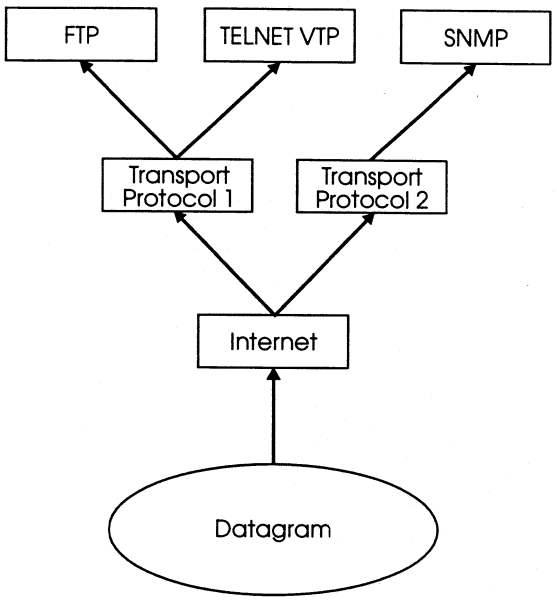


Figure 1.4

Notes:

Connectionless Service

A connectionless service is like the telegram system. Each piece of data must be individually addressed and delivered to its destination. Each *datagram* can hold a finite quantity of data. There is often no guarantee of delivery and no notification that the datagram has arrived. Advantages however are that of broadcast ability network capacity utilisation.

Note that X.25 packet switched networks (e.g. ARPANET) used to operate in this manner. Older X.25 (1980) networks also have a datagram facility. Ethernet and Token Ring networks are connectionless at the lowest level.

Not withstanding the delivery problem associated with a connectionless service, some application protocols like Network File System (NFS) use User Datagram Protocol (UDP) which is connectionless.

Connection-Oriented Services

A connection-oriented service is like the telephone system. Before exchanging data (speaking), we must first create a connection with the other end. For this we need the network address of the remote party (telephone number), we then call the party and request a connection (dial number and called party answers). Having established this connection, we exchange data (have a conversation) and then terminate the connection (put the phone down) on completion. This type of service has limited application in public networks due to difficulties with broadcasting.

X.25 networks are inherently connection-oriented and provide us with a *virtual circuit*, or point-to-point link, between the two DTEs.

Sliding Windows

This is a more complex form of positive acknowledgement which makes better use of the available network bandwidth. *Sliding Window* techniques allow multiple packets to be sent before waiting for acknowledgement.

For example initially there are 5 packets to send, and the window's size is set to 3. The packets within the window are transmitted, as they are acknowledged the window moves to the right, revealing more packets to transmit. Timers are maintained for each packet and cancelled on acknowledgement. The window only slides past acknowledged packets.

Performance is related to window size and the rate at which the network can accept packets. A well tuned sliding window protocol will keep the network saturated with packets obtaining better throughput.

Acknowledgements can also be "piggybacked" on data carried in the opposite direction.

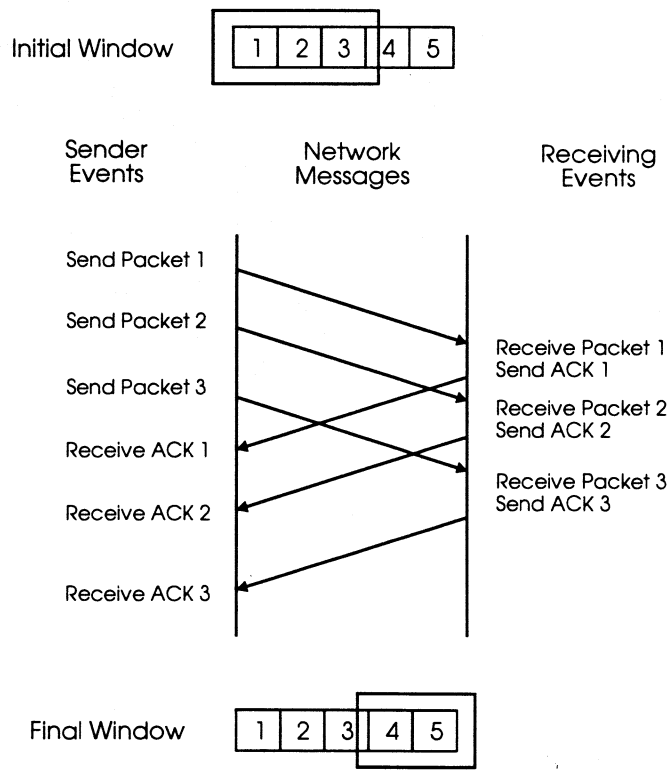


Figure 1.5

OSI Reference Model

The Open Systems Interconnection (OSI) committee was established (under the control of the International Standards Organisation) during the late 1970s with a broadly similar brief to that of the Internet Activities Board (IAB) - to bring some order to the chaos of multi-vendor networking.

The first task undertaken by the OSI committees was to define an architecture which could be used as the basis of their communications system. This architecture became the OSI Reference Model (OSI/RM) and is widely used as ***the*** reference model for all communications.

OSI ≠ Open Systems

OSI means Open (i.e. publicly debated/owned) method of Interconnecting Systems (for systems read *proprietary* systems). It is not, *per se*, a method of interconnection Open Systems. In practice, TCP/IP is the *de facto* method of interconnecting Open Systems due to wide deployment of the protocol. When referring to ISO standards, TCP/IP is ***not*** an OSI protocol as such, though we typically apply the TCP/IP protocols to the functional layers of the OSI/RM.

Notes:

Reference Model

The model has seven layers - the significance in this is that it is a number not too large to cause confusion and not too small to over simplify the model's functions.

The model shows the mechanisms that are involved in communicating between two *end systems*, possibly via one or more *intermediate systems*. The model defines a layered, peer-to-peer networking architecture.

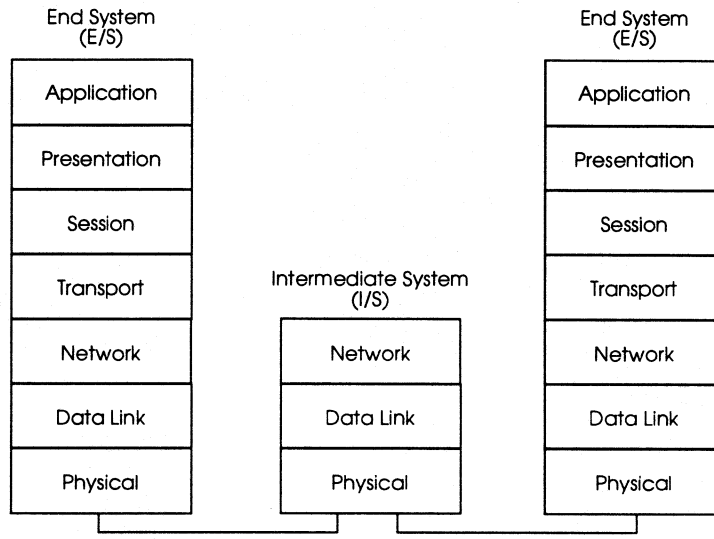


Figure 1.6

The layers of the model are:

1. Physical Provides the transparent transmission of bit streams between systems including relaying. This layer is directly connected to the physical medium which connects the systems. It sends and receives a stream of bits across the medium, be it wire or radio link, providing a physical connection between the two systems which are communicating.

2. Data Link Provides the transfer of data between directly connected systems and detects any errors in the transfer. The data link layer controls the flow of data, the correction and detection of errors, and sequencing. This may include deciding which system can transmit data at which time (access control). This layer can also detect errors in the received data and correct them.

3. Network Provides routing and relaying through intermediate systems. The network layer controls routing and other functions, including flow control and relaying. If there is no direct connection between the two systems that wish to communicate, the network layer finds out what intermediate systems can relay the messages to their destination, or if necessary, what other networks there are which can carry the connection part of the way.

4. Transport Provides the transparent transfer of data between end processes. The transport layer is the hinge between the upper and lower part of the model. It provides and monitors the quality of service and error rate and can send several messages down the same network connection at the same time. This is known as multiplexing data streams.

5. Session Provides the means to organise and synchronise the dialogue between application processes and manage their data. The session layer sets up a framework for dialogue between systems. When a connection is established, the session layer arranges the way data should be sent - in both directions at once, in alternate directions, or in one direction - for as long as the connection is required.

Notes:

6. Presentation

Provides for the representation of information that is communicated between or referred to by application processes. This layer is concerned with the way data is represented. The communicating systems may use character sets and different codes to represent particular symbols. The presentation layers of the two systems agree on a common representation. They then translate the data which is sent to them by the application layer into that common representation. For example:

Little Endian byte order, low to high (Intel, VAX, Alpha)
Big Endian byte order, high to low (SPARC, Motorola, IBM, ...)

7. Application

Provides a window between application processes in order to exchange meaningful information. The application layer does not contain application programs. It is the layer beneath the application programs, which receives their instructions when they want to send information to other programs on other systems. For example:

FTAM - File Transfer Access Method
VTP - Virtual Terminal Protocol
X.400 - Mail

Notes:

TCP/IP and OSI/RM

When considering applying the TCP/IP protocol suite to the OSI/RM, the key factor is what functionality is being deployed by the protocol.

For example File Transfer Protocol (FTP) is referred to as an Application level protocol because it implements Application level functions. The Internet Protocol (IP) is a Network layer protocol because it implements Network level functions.

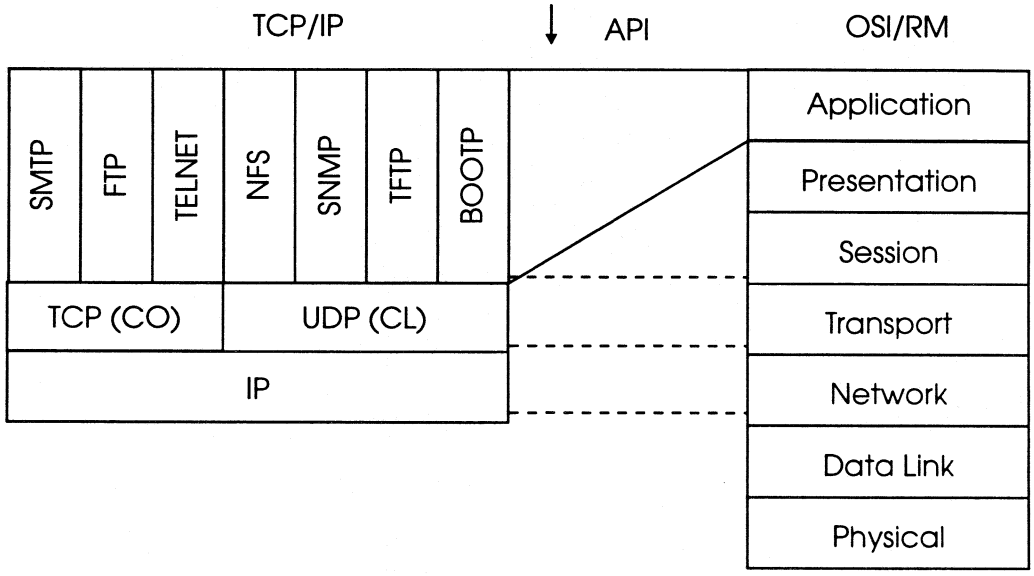


Figure 1.7

Notes:

TCP/IP Layers

Some of the major protocols within TCP/IP and the relationship between them are noted below. We will be discussing all of the protocols shown during the next few chapters.

ARP	Address Resolution Protocol
RARP	Reverse Address Resolution Protocol
IP	Internet Protocol
ICMP	Internet Control Message Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
SMTP	Simple Mail Transfer Protocol
FTP	File Transfer Protocol
TFTP	Trivial File Transfer Protocol
TELNET	Terminal Emulation
BOOTP	Remote Boot Protocol
SNMP	Simple Network Management Protocol
NFS	Network File System
DNS	Domain Name System
DHCP	Dynamic Host Configuration Protocol
SMB	Server Message Block
SSH	Secure Shell

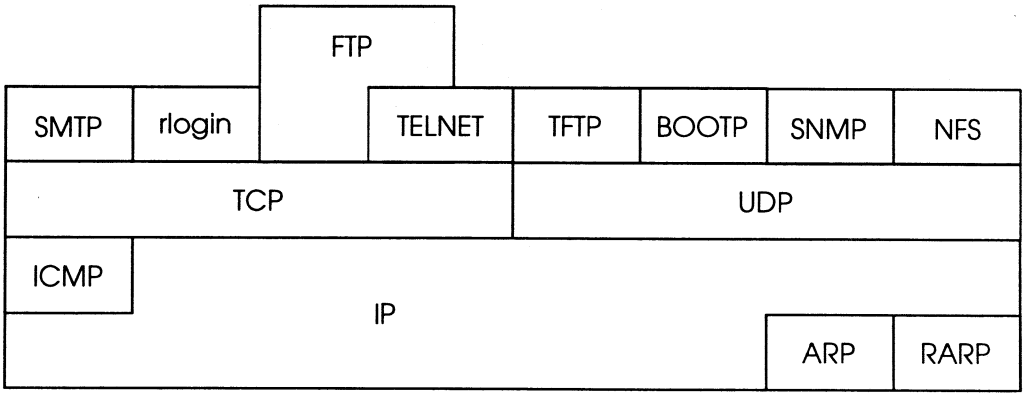


Figure 1.8

The Internet

The Internet is a collective term used to describe a number of connected networks, centered on the DARPA ARPANET. The networks link together academic institutions, commercial organisations, government bodies, private organisations and even individuals with interests in the defence field. The shape of the current Internet was more or less established by January 1983 when all computers connecting to ARPANET where required to use TCP/IP. Since that time the Internet has grown at an amazing rate, and along with it TCP/IP has increased in popularity and functionality. This has extended to the point where it is the *de facto* networking standard for most organisations.

The Internet Architecture Board (IAB) is responsible for the development of TCP/IP and related matters. The IAB coordinates research and development activities as well as underwriting the standards which are generated by this activity. See Appendix A which contains the text of RFC1160 - an outline of the activities of the IAB.

Details of the standards, as well as a wealth of other material, appears in documents called RFCs - *Request For Comments*. These are contributed by anybody who is interested in doing so. Appendix B contains a list of some important RFCs.

Notes:

2

Addressing - The Internet Structure

Addressing - The Internet Structure

Introduction

In this section we look at the Internet addressing scheme adopted by TCP/IP. Following that we examine the very important topic of address resolution - the mapping between the Internet address and the actual address of the physical network in use. The reverse address resolution - mapping a network address onto an internet address is also reviewed.

Objective

At the conclusion of this section delegates will be able to:

- Successfully describe the different IP address classes
- Describe the ARP and RARP processes
- Confidently specify IP addresses for an IP network.

Notes:

IP Addressing

Before we can understand how TCP/IP moves data across an Internet, we must know how it identifies connections to the network, the IP Address or Network Layer address.

All TCP/IP communications requires a knowledge of the IP address. Conversations are identified firstly by the Source and Destination IP addresses.

TCP/IP addressing helps to hide the physical network details, making the Internet appear a uniform entity.

The Internet provides a *universal communication service* because it permits any hosts to talk to any other host. To achieve this we need a universally adopted method of addressing, or identifying, the host computers.

TCP/IP uses a 32-bit addressing mechanism which encodes the location of the host (in terms of a network identifier) which may also be used to route data to the host. Routing is described in detail in section 6.

A host may have multiple connections to a physical network, or to many physical networks. This means that a single host may have multiple addresses - hence we say that TCP/IP addresses specify a *network connection* rather than a host address.

Notes:

Address Classes

All addresses are based on a 32-bit integer which contains a network id and the id number of the host on that network.

Networks on the Internet vary between NSFNET, a very large network with many hosts and gateways attached - there are only a few networks this size. At the other extreme is the small site LAN, based on Ethernet. There are a large number of these small networks that typically have < 255 host connections.

The three network classes cater for these two extremes:

Class A For very large networks with more than 16 million hosts.

Class B For intermediate networks with between 256 and 65,000 hosts.

Class C For small networks with less than 255 hosts.

Dotted Decimal Notation

32-bit integers are a bit difficult for the average human to deal with when represented in binary form. For this reason we use a *decimal dotted* syntax for all internet addresses which represents the address as four integers, separated by decimal points.

10000000 00001010 00000010 00011110
becomes
128.10.2.30

Each integer gives the value of 1 octet.

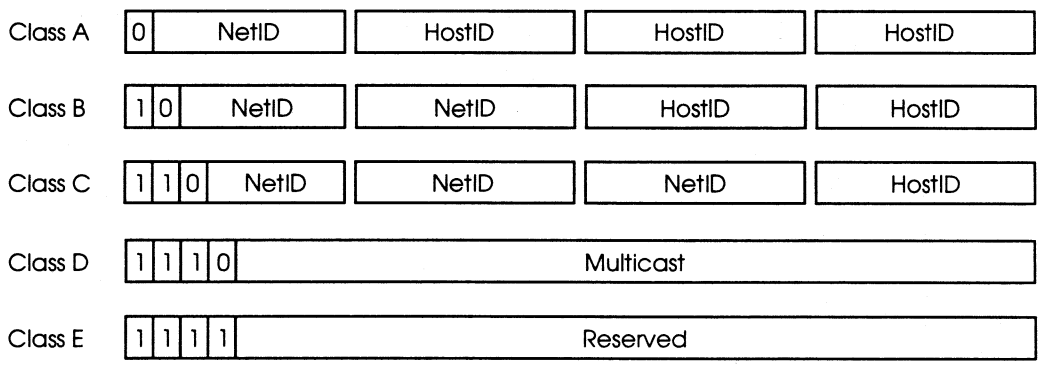


Figure 2.1

Special Addresses

The structure of internet addressing permits the use of a *broadcast* mechanism to address all hosts connected to a network. The host id is set to all 1's to indicate a broadcast. Properly implemented software will map this to a hardware broadcast facility, if available. Ethernet broadcast is efficient and imposes no overhead, X.25 does not support broadcast and so broadcast messages on X.25 will have a major impact on network load.

Internet addresses can refer to networks as well as hosts, simply by setting the host id to all 0's. This is a useful convention as quite often we need to know about networks specifically - e.g. for routing and error reporting.

By convention if an address contains all 0's in any component, it is interpreted to mean *this network* or *this host*. This can be very useful in situations such as the booting of a diskless workstation, which may not know its internet address initially. Such a device could use an address of all 0's until the correct internet address has been determined.

In addition, an address with all 1's in any component stands for a broadcast address:

Net ID	Subnet ID	Host ID	Description
-1 255		-1	Limited broadcast (never forwarded)
netid		-1	Net-directed broadcast to netid
netid	subnetid	-1	Subnet-directed broadcast to netid, subnetid
netid	-1	-1	All subnets directed broadcast to netid

Note: Address 127.0.0.1 is reserved for an internal loopback mechanism which in practice means that the whole class A network 127 is reserved.
Some old UNIX hosts used a host portion of all 0's to indicate the broadcast address.

Subnetwork Addressing

For routing to work, the IP Addresses on the respective network connections must have different network IDs. Since registered network IDs are issued by the IAB, there has to be a mechanism for network planners to allocate their own (Sub) network IDs to make routing work. Otherwise the planner must allocate many non-registered IP addresses just to get different network numbers.

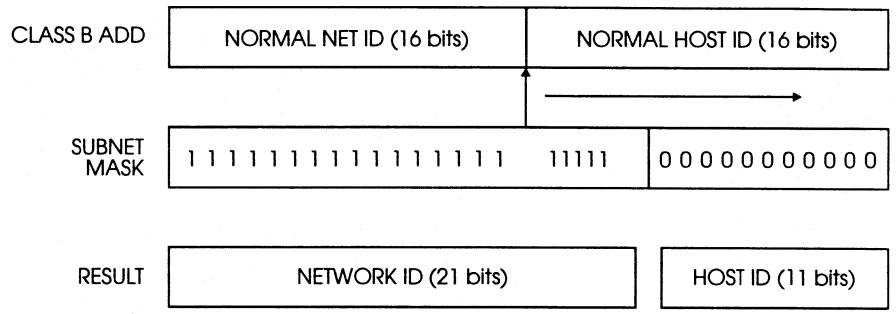


Figure 2.2

The mechanism that has been adopted is the sub network mask. This 32-bit number when placed over the IP address, selects out those bits which are to be regarded as the Network ID and those which are the Host ID.

Subnet addressing and the subnet mask puts network numbering in the control of local management.

1 in the Mask = the corresponding bit in the IP address is part of the Network/Subnet ID

0 in the Mask = the corresponding bit in the IP address is part of the Host ID

It is recommended that no gaps are left in the mask - 1s should be added from the most significant bit of the address classes network ID portion.

The subnet mask can be used with any class of address and provides a planner, the tool to trade the total number of hosts available on a subnetwork for the total number of subnetworks.

A subnetwork mask must be loaded into a host or router at the same time as the IP address is allocated. If it is not, the equipment generates the default mask for the class of address in use.

Class A	255.0.0.0
Class B	255.255.0.0
Class C	255.255.255.0

A host specific route can be allocated with the mask 255.255.255.255. The default route can be allocated with the mask 0.0.0.0.

Networks larger than about 200 hosts should always be planned with subnet addressing in mind as changes later can be labour intensive.

Notes:

Issues with Internet Addressing

Internet addresses represent network connections and are associated with interfaces (Ethernet, PPP, SLIP, etc.). If a host is moved it will receive a new connection point and may receive a new address, which can create problems.

If your Class C network grows beyond 255 connections, you will have to upgrade it to a Class B. There is no easy way of doing this as most software is not able to handle multiple addresses for the same physical network, therefore you cannot phase it in gently over a period of weeks. All changes have to be completed on the same day!

The IP address determines the route taken to reach the host, hence it also has the effect of restricting access to the host to a single point. Consider a simple network where users on host A send messages to host B via I_4. An alternative route would be via gateway G and I_5. If link I_4 fails for some reason, messages addressed to host B will not be delivered unless users use I_5. Even though host B is operational and connected to the internet, it will be unreachable from host A unless the users are aware of the alternative address.

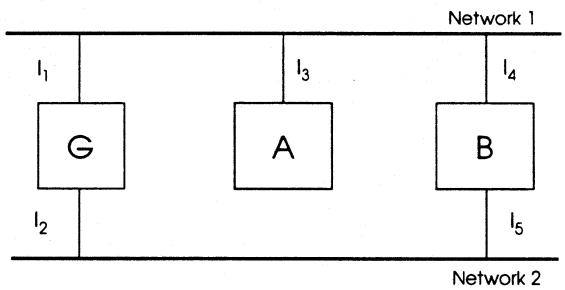


Figure 2.3

Notes:

IP Version 6 Addressing Architecture

As at May 1995 the new version of IP known as V6.0 was still in a draft format before the Internet Engineering Task Force (IETF). As such the finalisation of the new addressing scheme to increase the available network addressing options is still in a state of 'Work in Progress'. The comments here reflect the current state of the new addressing scheme.

IPv6 Addressing

IPv6 addresses are 128-bit identifiers for interfaces and sets of interfaces. There are three types of addresses:

- **Unicast:** An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.
- **Anycast:** An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to one of the interfaces identified by that address (the "nearest" one, according to the routing protocols' measure of distance).
- **Multicast:** An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

There are no broadcast addresses in IPv6, their function being superseded by multicast addresses.

Notes:

Addressing Model

Under IPv6 addresses of all types are assigned to interfaces, not nodes. Since each interface belongs to a single node, any of that node's interfaces' unicast addresses may be used as an identifier for the node. An IPv6 unicast address refers to a single interface. A single interface may be assigned multiple IPv6 addresses of any type (unicast, anycast, and multicast). There are two exceptions to this model. These are:

- A single address may be assigned to multiple physical interfaces if the implementation treats the multiple physical interfaces as one interface when presenting it to the Internet layer. This is useful for load-sharing over multiple physical interfaces.
- Routers may have unnumbered interfaces (i.e., no IPv6 address assigned to the interface) on point-to-point links to eliminate the necessity to manually configure and advertise the addresses.

There are three conventional forms for representing IPv6 addresses as text strings:

- The preferred form is x:x:x:x:x:x:x:x, where the 'x's are the hexadecimal values of the eight 16-bit pieces of the address. Examples:
  - FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
  - 1080:0:0:0:8:800:200C:417A
- Due to the method of allocating certain styles of IPv6 addresses, it will be common for addresses to contain long strings of zero bits. In order to make writing addresses containing zero bits easier a special syntax is available to compress the zeros. The use of two "::" indicate multiple groups of 16-bits of zeros. For example the multicast address: FF01:0:0:0:0:0:0:43 may be represented as: FF01::43. Note - The "::" can only appear once in an address.
- An alternative form that is sometimes more convenient when dealing with a mixed environment of IPv4 and IPv6 nodes is x:x:x:x:x:x:d.d.d.d, where the 'x's are the hexadecimal values of the six high-order 16-bit pieces of the address, and the 'd's are the decimal values of the four low-order 8-bit pieces of the address (standard IPv4 representation). Examples: 0:0:0:0:0:0:13.1.68.3 and 0:0:0:0:0:FFFF:129.144.52.38, or the same addresses in compressed form: ::13.1.68.3 ::FFFF:129.144.52.38

Internet Addressing Authority

Control of addresses for networks attached to the Internet is controlled by the Network Information Centre (NIC). They assign the Netid portion of the address and delegate control of the Hostid portion to the local administrator.

If you are not going to be connected to the Internet (right away) and you don't want to apply for a network number, you should give consideration to using one of the address ranges allocated to private networks.

Class	Start	End	Net/Bits
A	10.0.0.0	10.255.255.255	10/8
B	172.16.0.0	172.31.255.155	172.16/12
C	192.168.0.0	192.168.255.255	192.168/16

You should not just grab some network number out of the air, as more organisations are moving to connect to the Internet. If you do, you will be up for a large re-numbering exercise. By using one of the above you can install a masquerading gateway and avoid much pain.

In Australia Telstra Internet is responsible for the IP address distribution.

Notes:

Mapping Internet Addresses to Physical Addresses

The internet behaves like a virtual network, with a packet delivery system and addressing mechanism. However, in order for two hosts on the same network to exchange messages they have to know their *physical addresses*. We need a means of mapping an internet address onto a physical address. There are two techniques used to achieve this:

Direct Mapping

This is the simplest mechanism. We arrange for there to be a common element between the internet address and physical address. For a Token Ring network with station numbers in the range 1-255, we can establish the network as Class C and assign the same hostid and physical number to each host, e.g.

Stn 27 <-> 192.5.48.27

X.25 and Frame Relay networks using X.121 addressing schemes are difficult to manage like this so it is more common to use a lookup table with pairs of internet and X.25 addresses. Access via a hashing algorithm is reasonably efficient.

Dynamic Mapping

When dealing with networks such as Ethernet, it is impractical to encode the 48-bit physical address within the 32-bit internet address space. It is also desirable that hosts can be attached and removed from the network without disrupting the software, or other hosts. A special protocol, *Address Resolution Protocol* (ARP), is used to do this. ARP dynamically maps the address and therefore avoids the problems inherent in fixed tables of mappings.

Notes:

Address Resolution Protocol - ARP

The ARP protocol has been specifically designed to work in an Ethernet environment and is implemented on all Ethernet-based TCP/IP software.

Host A wishes to send a packet to host B, it does not know the physical address of host B. The first stage is to broadcast a packet (using the Ethernet broadcasting mechanism) which contains the Internet address of host B. All hosts on the network receive the broadcast message, but only host B recognises its own internet address.

Host B replies with a packet that contains both its internet and Ethernet addresses. Host A saves this information in its cache for subsequent use and forwards its data to host B using the Ethernet address.

The ARP mechanism is slightly more sophisticated than this. When host A makes its initial broadcast it also includes its own Internet/Ethernet binding. Some more sophisticated hosts on the network (who received the broadcast) use this information to update their caches, increasing the chance of a successful mapping without the need to broadcast. In addition to this, it is normal to broadcast as the host starts up and also to flush the address cache at intervals of about 20 minutes (can be changed by the system manager on some implementations).

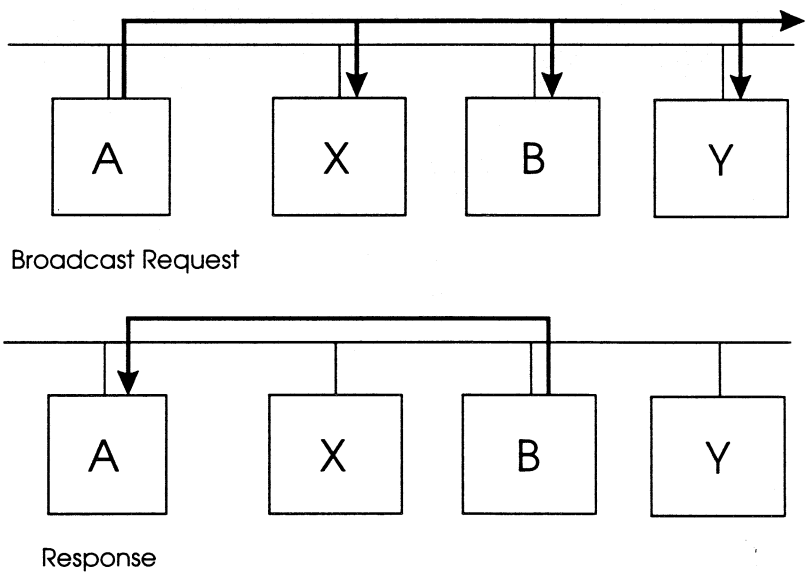


Figure 2.4

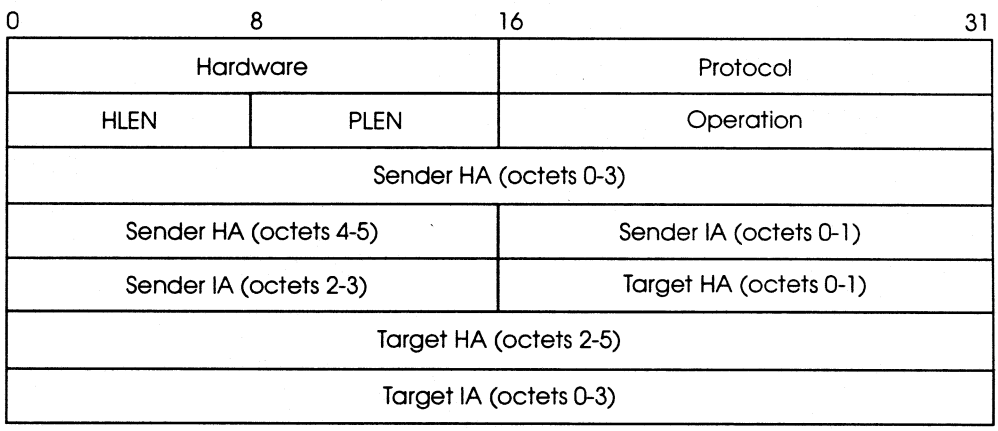
ARP Format

ARP has been generalised for uses on LAN networks other than Ethernet hence the format of the protocol is variable length. The example shown is for Ethernet.

Hardware Protocol	Hardware interface type (Ethernet = 1)
Operation	1 - ARP request, 2 - ARP response, 3 - RARP request, 4 - RARP response
HLEN	Length of hardware address
PLEN	Length of protocol address
Sender HA/IA	Sender hardware and internet addresses
Target HA/IA	Target hardware and internet addresses

The 28 octet ARP message is encapsulated within the Ethernet frame as data.

N.B. ARP Ether type = 0x0806, ARP reply type = 0x8035



* Sender Header Address - 48-bit Ethernet Address
* Sender Internet Address - 32-bit IP Address

RFC-826

Figure 2.5

Potential Problems

Potential problems of the dynamic ARP process are:

- It can take a significant time if the target machine is down or overloaded
- ARP's can flood the network if a lot of machines simultaneously need to send.

ARP Configuration Parameters

When building ARP software, the programmer configures the system by choosing values for parameters such as:

- The size of the ARP cache
- Time-out interval the client waits for an ARP response
- Number of times a client re-tries a request
- Time interval between re-tries
- Timeout (Time To Live) for a cache entry
- Size of the packet retransmission queue.

Notes:

Reverse Address Resolution

Diskless hosts present a particular challenge to the addressing system because a number of hosts share a single system image. This image cannot contain any embedded network information (for obvious reasons).

All the diskless host has at startup as its LAN MAC address (permanently held in the NIC controller hardware) - this does, however, uniquely identify the host. The solution to the problem involves contacting a server and requesting the Internet address - the protocol for doing this is called the Reverse Address Resolution Protocol - RARP.

One problem with this is that in a TCP/IP environment diskless workstations require a lot more configuration information than just the IP address. BOOTP can provide all this and is therefore becoming more popular for these workstations. Another issue is that often programmers do not have a defined interface into RARP.

Notes:

Reverse Address Resolution Protocol - RARP

RARP relies on the existence of *RARP Servers* on the network, which contain tables of internet and LAN MAC addresses.

Host A starts up and broadcasts a RARP request, specifying itself as the target by providing its Ethernet address. All hosts receive this, but only RARP servers respond by sending back the RARP message with host A's internet address included. There may be more than one server, in which case there will be several replies.

The message format used is the same as that used by ARP. It should be noted that the message format allows a host to ask about any target.

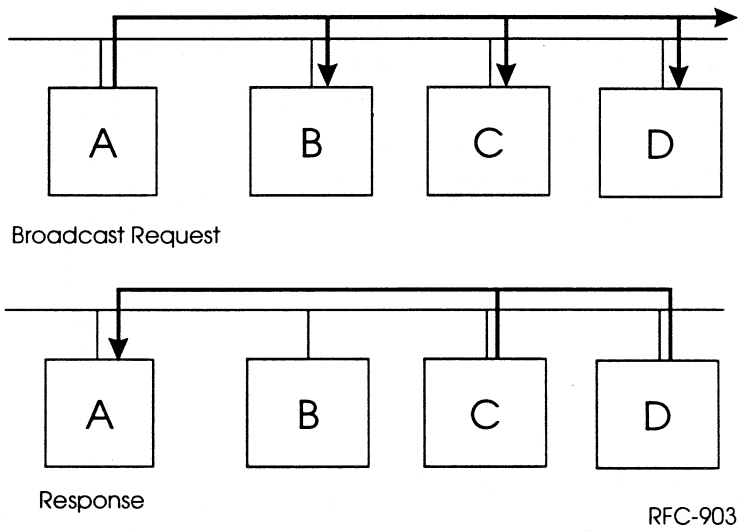


Figure 2.6

Notes:

TCP/IP Implementations

TCP/IP can be implemented across virtually any network type, the most common types are:

- Internet (HDLC)
- Ethernet and Token Ring LANs
- X.25 WANs
- PPP
- ATM (of late)

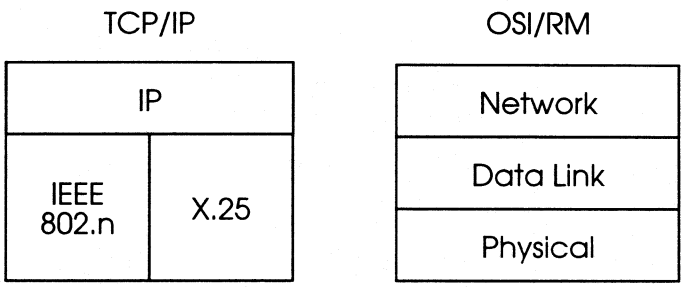


Figure 2.7

Notes:

TCP/IP over Ethernet

Issues that need to be addressed by a TCP/IP over Ethernet are:

- Which "Ethernet"
  - DIX Ethernet vs. 802.n
  - SNAP Protocol
- Address resolution and mapping
  - ARP
- Diskless workstation support
  - RARP

Network Layer Interface

The network layer interface has to support IP datagram encapsulation and a mechanism for demultiplexing arriving datagrams (between IP and any other protocols). The differences between DIX Ethernet and the IEEE 802.3 or 802.5, with IEEE 802.2 LLC headers have an impact in this area.

Trailer encapsulation is an Ethernet-specific protocol that modifies the encapsulation mechanism in order to optimise efficiency for hosts with limited memory page size.

Trailer encapsulation is however totally deprecated and should not be used.

Address Resolution and Mapping

Ethernet makes use of a dynamic address resolution protocol - *Address Resolution Protocol (ARP)* - to match (bind) an IP address with an

Notes:

Ethernet address. The implementation of this protocol provides a number of optimisations that provide an efficient solution to the requirements.

BOOTP, DHCP and TFTP

In modern environments it is often necessary to support a diskless workstation or to configure a workstation's TCP/IP environment from a central location. While RARP provides a mechanism to give a workstation its TCP/IP address, other, better protocols include:

BOOTP	Allows a workstation to pick up almost all of its TCP/IP configuration and the name of its boot file.
DHCP	Allows a workstatioin to pick up more TCP/IP configuration information than BOOTP.
TFTP	Used by diskless workstations and remote boot PROMs to download operating systems or other configuration information.

Notes:

Ethernet Network Interface

Interfacing TCP/IP to CSMA/CD networks is complicated by the existence of two network protocols - DIX Ethernet and the IEEE LAN protocols (802.n). Definition of DIX Ethernet (V.2) and IEEE 802.3 is basically compatible except that ESPEC2 contains a *Packet Type* field and IEEE 802.3 contains a *Data Length* field of the same size (16 bits) and at the same location within the frame header. This means that the two systems may coexist on the same LAN, but will not interwork.

Subnetwork Access Protocol (SNAP) available for LLC but not widely used. SNAP protocols allow the Ethernet v2.0 packet type (0800 for IP) to be preserved in standard compliant 802.2 LLC headers.

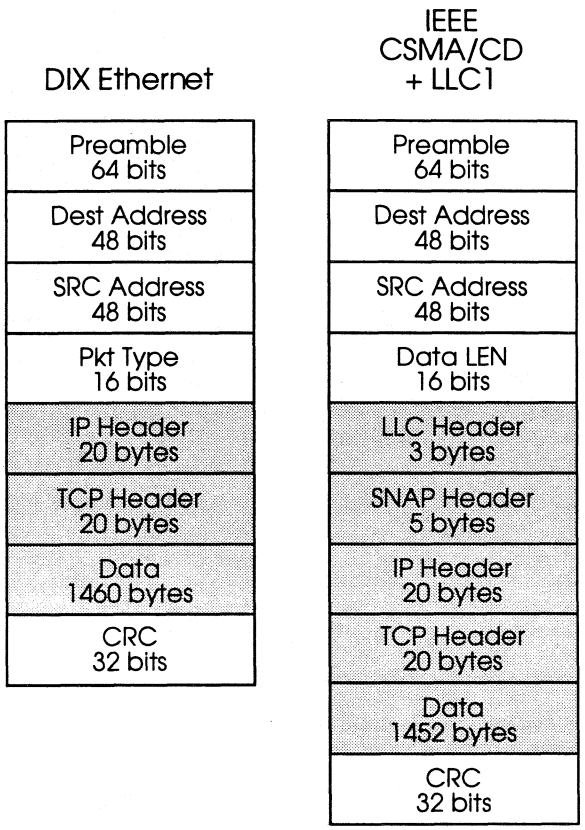


Figure 2.8

Notes:

There are two mechanisms defined for interfacing TCP/IP to Ethernet-type networks - RFC 894 *(Standard for transmission of IP datagrams over Ethernet networks)* and RFC 1042 *(Standard for transmission of IP datagrams over IEEE 802 networks).*

RFC 894

RFC 894 defines the use of the Ethernet type field for distinguishing between protocols - 0x0800 is reserved for use by IP, 0x0806 for ARP. This can also coexist with 802.3 implementations, therefore check the length field - if it is greater than 1500 then RFC 894 addressing is assumed, otherwise the data is passed to the LLC sub-layer for processing.

RFC 1042

RFC 1042 defines the use of the *Subnetwork Access Protocol (SNAP)* that is used to interface IP to the LLC, as well as the use of LLC 1 and 2 as defined by IEEE 802.2. The combined LLC/SNAP headers look like this:

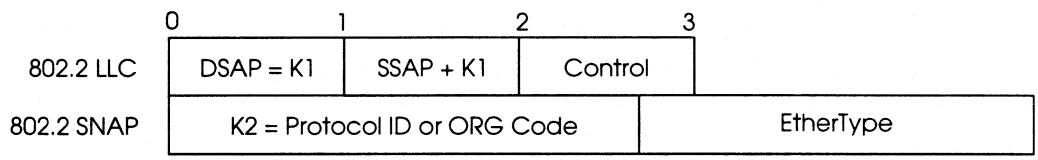


Figure 2.9

K1 = 0xAA, Control = 3 (UI - Unnumbered Information), K2 = 0 and the EtherType is 0x0800 (IP) or 0x0806 (ARP). SAP 0xAA is reserved for use by SNAP.

Whilst the majority of Ethernet hardware equipment sold today conforms to IEEE 802.3 it is still unusual to find LLC implemented for operating systems such as UNIX or VMS. RFC 894 is the de facto standard for interfacing to both DIX and IEEE 802.3 **hardware.**

TCP/IP Over X.25

An implementation of TCP/IP over X.25 addresses similar issues to the Ethernet and LAN MAC implementation example.

Network Layer Interface

The network layer interface has to support IP datagram encapsulation and a mechanism for demultiplexing arriving datagrams (between IP, ICMP and any other protocols). The connection-orientated implementation of modern X.25 networks complicates the implementation to a degree.

Address Resolution and Mapping

Address resolution adopted for use in X.25 networks is invariably via static tables - X.25 hosts tend not to appear and disappear with the same regularity as Ethernet hosts.

Connection between Ethernet (IP-E or IP-IEEE) and X.25 networks (IP-X25) is usually implemented within a dedicated router *(IP router)*, strictly a *gateway* in TCP/IP terminology. Hosts connected onto both networks may also implement a *gateway* service.

Notes:

X.25 Network Interface

X.25 provides a connection-orientated network service, based on virtual circuits. X.25 (1980) did in fact provide a connectionless service based on *datagrams* but this was dropped in X.25 (1984) and X.25 (1988).

IP-X.25 works by opening a Switched Virtual Circuit (SVC) to the destination host when an IP datagram arrives at the network interface. If the circuit subsequently becomes inactive for a period then the circuit may be closed; if the interface runs out of circuits then existing circuits may also be closed. Clearly, this mechanism does not affect higher level protocols such as TCP - the TCP streams remain *open* even if the SVC is closed, just as over a connectionless network.

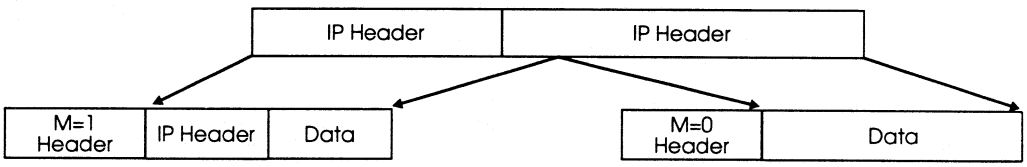


Figure 2.10

Protocol demultiplexing (to IP or ICMP etc.) occurs during call setup - the first octet of the Call User Data Field is set to 204 for IP. The remote host then passes packets from this SVC to an *IP server* that waits for incoming calls with this CUDF.

Use is made of X.25 packet fragmentation facilities - IP does not fragment packets that exceed the MTU. X.25 Packet Layer treats each datagram as *complete packet sequence* - this means that the IP datagram is fragmented to fill a number of packets. Each packet in the sequence

Notes:

(except the last) has the M-bit set to indicate that there is more data to complete the sequence. This mechanism has the advantage of reducing the overhead of placing a copy of the IP header with each fragment when IP handles fragmentation.

Datagrams may be lost if either end closes the SVC or a reset is experienced.

IP datagram size is restricted to 576 octets, unless individual sites negotiate sizes up to 1024 octets. This is independent of negotiated X.25 packet and window sizes.

Since services that use IP must expect it to deliver out of sequence datagrams, multiple X.25 connections can be safely opened to the same hosts to overcome throughput limitations of X.25 window and packet sizes.

Notes:

X.25 Address Mapping

It is not feasible to encode IP addresses - there is not enough room and when using public X.25 networks the host does not usually have control of the address assignment.

IP Address	X.121 Address
128.10.2.1	234219957411
128.10.2.2	234219887422
128.10.2.4	234225586432

Figure 2.11

For this reason address resolution is normally provided by a lookup table that relates host IP addresses to their X.25 NUA. If a hashing algorithm is used, access speed is usually quite adequate.

The big problem with fixed tables is maintenance - imagine the work involved with a network of 200 hosts when you add a new one! When table based routing is used it is essential to have some form of *Network Information Service* - such as Yellow Pages from Sun Microsystems.

X.25 addresses *(Network User Addresses - NUAs)* are defined according to the ITU-T X.121 standard. They consist of:

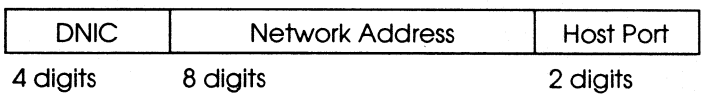


Figure 2.12

Notes:

TCP/IP and Other Networks

The IAB list the following *official* protocols for supporting IP across different network types:

IP-ARPA	Internet Protocol on ARPANET	BBN 1822
IP-WB	Internet Protocol on Wideband Network	RFC 907
IP-X25	Internet Protocol on X.25 Networks	RFC 877
IP-E	Internet Protocol on Ethernet Networks	RFC 894
IP-EE	Internet Protocol on Exp. Ethernet Networks	RFC 895
IP-EEE	Internet Protocol on IEEE 802	RFC 1042
IP-DC	Internet Protocol on DC Networks	RFC 891
IP-HC	Internet Protocol on Hyperchannel	RFC 1044
IP-ARC	Internet Protocol on ARCNET	RFC 1051
IP-SLIP	Transmission of IP over Serial Lines	RFC 1055
IP-PPP	Transmission of IP over Serial Point to Point lines	RFC 1171
IP-NETBIOS	Transmission of IP over NetBIOS	RFC 1088

There are also a large number of experimental protocols that are used for specific research projects e.g. running IP over ISO Transport and similar.

All of these protocols have to address the problems illustrated for Ethernet and X.25 which are data encapsulation and address resolution. Some protocols (e.g. PPP) are able to offer some optimisations because of the way they work. Some are very inefficient but can be used as a temporary migratory route (IP over NetBIOS).

3

Internet Protocol Suite

Internet Protocol Suite

Introduction

This section investigates the role of the Connectionless Packet delivery layer. This is the layer where the functions of the IP protocol are performed.

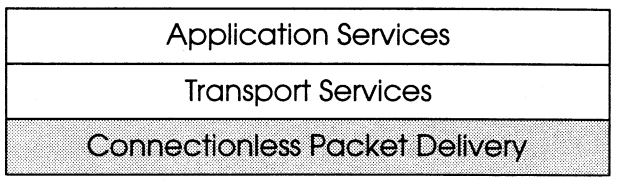


Figure 3.1

Objectives

At the conclusion of this section delegates will be able to:

- Describe the IP datagram in detail
- Understand the issues of packet fragmentation
- Describe the functions of internetworking.

Notes:

Connectionless Delivery System

Connectionless Delivery Service is defined by the Internet Protocol (IP) as follows. It:

- Defines the basic unit of data transfer - datagram.
- Defines data formats.
- Specifies how packets are processed.
- Specifies error handling.
- Allows a routing mechanism.

Other factors associated with this delivery system are:

Unreliable — Delivery is ***not*** guaranteed. The packet may be lost, duplicated, delayed or delivered out of sequence.

Connectionless — Each packet treated as an independent entity. As such, each packet sent (although in a sequence) may travel different paths. That is - *no fixed route.*

Best Effort — The internet software makes every effort to deliver the packet. No error correction or retransmission at this level.

Notes:

IP Datagram

The logical network defined by the IP connectionless delivery service is analogous to a physical network in many ways. A physical network uses a unit of transfer (usually called a frame) that consists of a header (containing source and destination addresses) and data. The basic transfer unit used by TCP/IP is called an IP *datagram* (sometimes an *Internet datagram, datagram* or *packet*).

The IP datagram has header and data areas, the header contains addresses (which are Internet addresses).

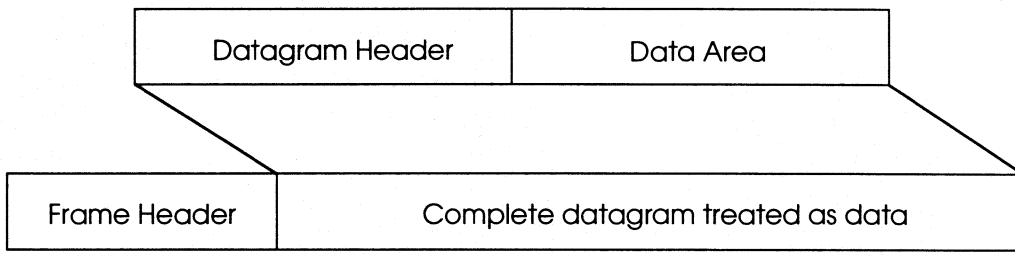


Figure 3.2

It is encapsulated into the frame of the physical network. Largest frame of the network is the Maximum Transfer Unit (MTU).

The IP datagram could have been any size the protocol designer chose because unlike physical networks, a logical network is handled by software, not hardware. The IP datagram is carried in the physical network frame as data. The most efficient way of handling this is to ensure that one IP datagram is the maximum data size of the physical network frame. This maximum data size of a physical network is called its **Maximum Transmission Unit (MTU)**. The MTU of Ethernet is 1500 bytes and of IEEE 802.3 with LLC & SNAP is 1492 bytes. The MTU of Token Ring is 4000 bytes approximately, but is dependent on ring speed.

Note: Currently, the maximum length of a datagram, including header, is 65,535 octets (16-bit total length field in datagram format).

IP Datagram Fragmentation

Every physical network has a *Maximum Transmission Unit (MTU)* that defines the maximum amount of data that can be transferred in a single frame. This means that choosing and IP datagram size for efficiency is not completely straightforward. Of course, in many cases TCP/IP is only used across a single network type, but the protocol has to be able to cope with an Internet consisting of multiple interconnected networks. For this reason IP datagram size is independent of the underlying network's MTU - this is much easier to manage than a lowest common denominator system.

A convenient MTU size is selected and a *fragmentation* mechanism is provided to break up an IP datagram into chunks that can be handled by the physical network. The remote host reassembles the fragments to recreate the original IP datagram. This means that the IP datagram is not normally reassembled until it reaches its final destination - the fragments may even be fragmented further to cross networks with smaller MTU sizes.

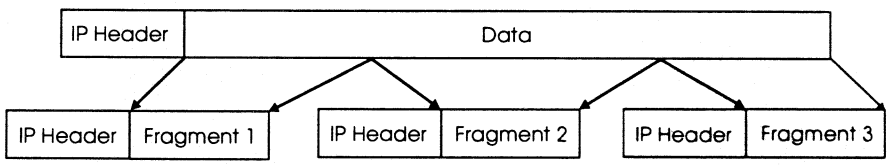


Figure 3.3

Notes:

Each fragment contains a copy of the original header plus a piece of data. The fragment data size has to be a multiple of 8 octets (because of the granularity of the length field).

On the Internet hosts and gateways are expected to handle IP datagrams of up to 576 octets without fragmentation (512 data bytes + header). In practice much larger IP datagrams are commonly found - particularly on Ethernet networks, where there is frequently either 1024 or 1500 bytes of protocol data.

If a fragment is lost in transmission then the *whole* datagram has to be retransmitted. This can have a large effect on the throughput of TCP connections as mechanisms like slow start and congestion avoidance can come into play.

The IP specification states that gateways must accept datagrams up to maximum of the MTUs of networks to which they attach.

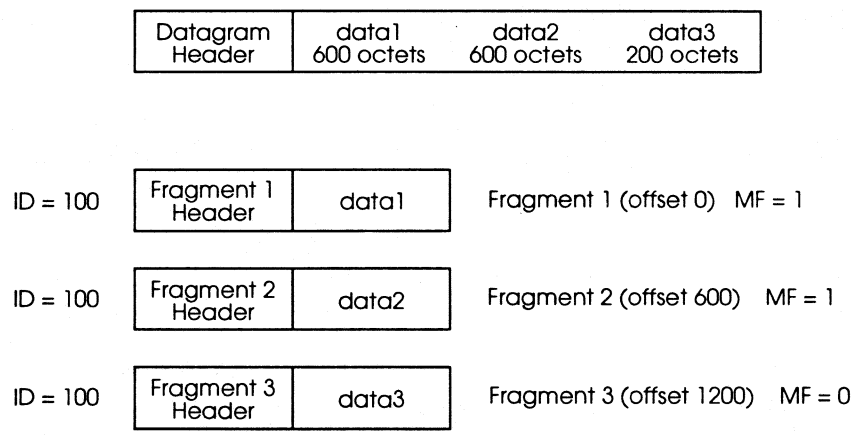


Figure 3.4

Example:

Datagram with 1,400 octets of data.

MTU of 620 octets.

Headers 1 and 2 have the *more fragments* bit set.

Note: Offsets shown are decimal octets and must be divided by 8 to get actual value stored in the fragment header.

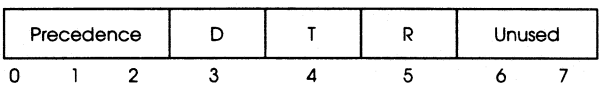
IP Datagram Format

VERS 4 bits representing the value of the version of IP protocol used to create this datagram (version 4).

HLEN 4 bits representing the length of the IP header in 32-bit words.

All fields within the header are fixed length with the exception of IP Options and Padding. The most common header has ***no*** options and ***no*** padding, hence a length of 20 octets - HLEN value 5.

Type of Service 8 bits representing:



Precedence allows sender to indicate importance of datagram

0 = Normal
7 = Network Control

DTR Type of transport required (as a hint to the routing algorithms)
D - requests low delay
T - high throughput
R - high reliability

Total length Gives total length of IP datagram measured in octets (including header). Maximum size 2^{16} or 65,535 octets.

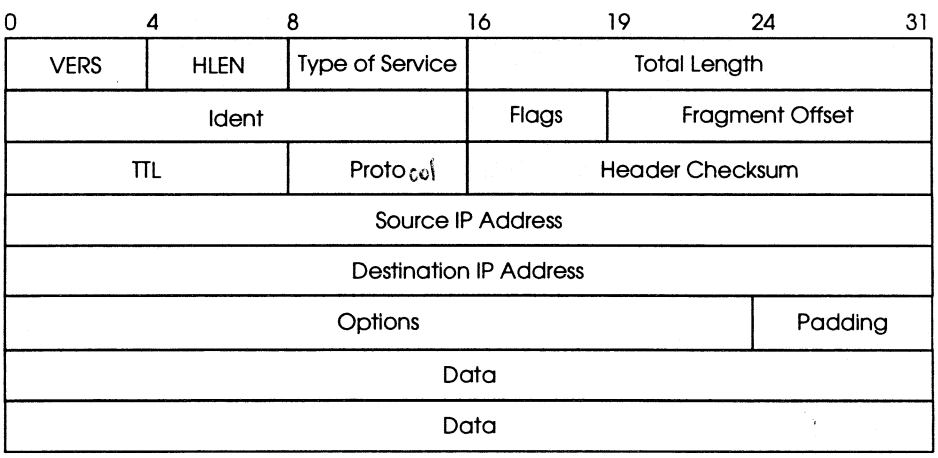


Figure 3.5

Ident A unique integer that identifies the datagram. The destination uses the IDENT field along with the source address to identify the datagram.

Flags 3 bits representing fragmentation
0 reserved must be 0
DF - Do not fragment bit

0 - may fragment
1 - do not fragment

MF - More fragments bit
0 - last fragment
1 - more fragments

Fragment Offset 13 bits specifying the offset in the original datagram of the data being carried in the fragment. Measured in units of 8 octets starting at 0.

TTL Time To Live - this indicates the maximum time in seconds that the datagram is allowed to exist within the internet. TTL is decreased by 1 each time it is processed. Whenever a TTL reaches 0, the datagram is destroyed and an error message is processed back to source.

Notes:

Proto The value in the protocol field specifies which high level was used to create the message being carried in the data portion of the datagram i.e.

1 - ICMP
6 - TCP
8 - EGP
17 - UDP
29 - TP4

Header Checksum A checksum on the **header only**. As some fields change i.e. TTL, the checksum is recomputed and verified each time the IP header is processed.

Header treated as a sequence of 16-bit integers in network byte order and using ones complement.

Source Address 32 bits representing source IP address.

Destination Address 32 bits representing destination IP address.

Options This field is not required in every datagram and is primarily used for testing and delivering. The length of this field is variable depending on what options are chosen.

Padding Field of zeros added to ensure datagram header ends on a 32-bit word boundary.

Notes:

Internetworking

The term *internetworking* is used to describe a number of discrete physical networks that are connected together to form a single network. A characteristic of such an "internet" is that the underlying physical network structure should be invisible to network users.

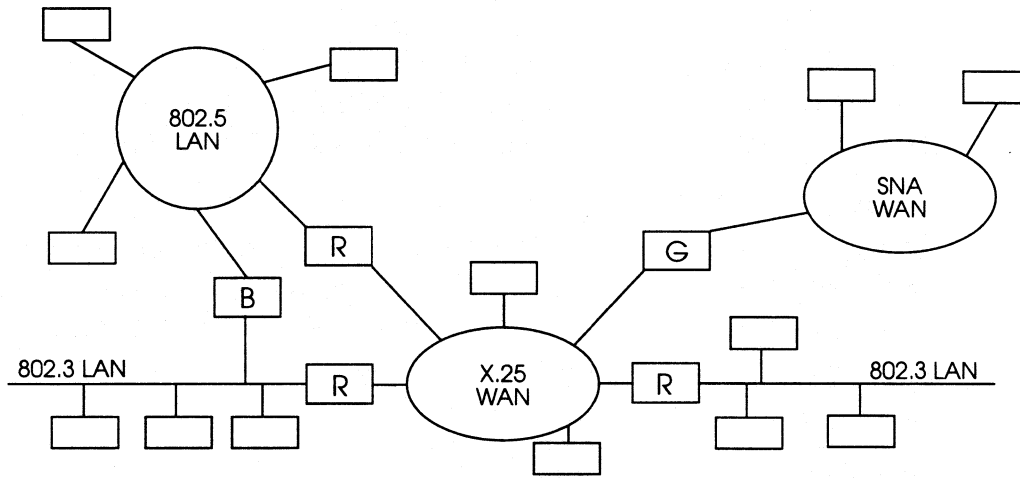


Figure 3.6

Figure 3.6 shows an "internet", consisting of physical networks of the four main types linked together by a device - generically known as a relay - that is responsible for handling the necessary conversions as packets move between networks. Examples of relay devices are:

Repeaters - amplifying devices for LANs
Bridges - used to connect LANS together
Router - used to connect between networks
Gateway - provide conversion between protocols.

If all of the networks in Figure 3.6 are using the same protocols then all of the relay devices could be routers; if the SNA network was not using the same protocols then a gateway would be offered to convert SNA and non-SNA protocols; the LANs could be linked by a bridge.

In practise there are a number of interpretations between the different terms that are used. In the TCP/IP world a *gateway* is used to connect between two networks (e.g. a router by the definition above). Often we use the term *gateway* and *router* interchangeably.

Internetworking Components

Bridges

Bridges are able to link together two physically separate segments of a LAN and create a single logical LAN. They operate over the first one and a half layers of the OSI/RM, and are often referred to as *MAC bridges*. The two LAN segments could be geographically remote from each other, with the bridge operating via a lease line, fibre-optic cabling or X.25. A feature of IEEE 802 LANs is that, as they share a common Logical Link Control (IEEE 802.2), dissimilar LAN types can be connected via a bridge (e.g. a CSMA/CD - TRN bridge).

Bridges are often thought of as protocol independent so far as higher level protocols such as TCP/IP, Decnet or Netware are concerned. As such they play no part in TCP/IP internetworking. But because of the representation of MAC addresses in these two technologies, the Bridge must understand and convert ARP request and responses.

They are, however, very widely used for building LANs and their presence can have a marked effect on network performance - we will examine this in more detail later.

Notes:

Routers/Gateways

A router is used to connect together two physically separate networks; unlike a bridge it does not create a single logical network. The function of a router is to forward packets between the networks in accordance with some routing algorithm. The term gateway is often used within the TCP/IP world to describe a device with these characteristics. For consistency we will continue to use that term, but note that in most (OSI-like) network environments a gateway is used to provide a protocol conversion.

Routers operate over the lower three layers of the OSI/RM, and so they are not protocol independent, as layer three (IP in TCP/IP) is used to define the routing method. Routers may either be implemented on general-purpose hosts or in dedicated gateway devices - there are a number of so called *IP Routers* available that implement a standalone *gateway*.

In practice, every implementation of IP has to have some functionality as a router. We need to look at the way in which routing is handled, and how routing information is disseminated amongst routers.

Notes:

IP Routing

The Internet consists of a large number of networks (in excess of 2000) that are connected together by gateways (routers). A function of any Internet is that there must be some mechanism to forward a packet to its eventual destination - possibly across a number of intermediate networks and gateways. This function is carried out by the IP layer within TCP/IP, the work is actually done within the gateways which have a sufficient knowledge of the topology to forward the packet either to another gateway or to the final destination.

There are two routing methods available within TCP/IP - *Direct Routing* and *Indirect Routing*.

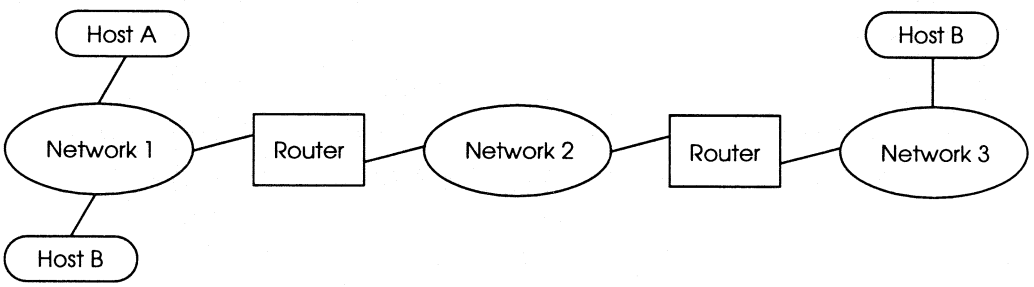


Figure 3.7

Notes:

Direct Routing

Direct routing is used to provide delivery of IP datagrams when both hosts lie on the same physical network. The Internet addressing scheme described in section 2 divides an address into two components - the network ID and the host ID. If the network ID portion of the IP address is the same for the source and destination then direct routing is possible.

The activity of IP is then to encapsulate the IP datagram and perform address mapping using the method defined by the physical network interface (as described in section 2).

If two stations are on the same logical LAN (separated only by **bridges** and **repeaters**), it is best that they have the same **Network ID**, however, by employing a router (or a host that can route between subnets) they can be on different nets.

If two stations are on different logical LANs (separated by **routers**) it is best if they have different **Network IDs** (the router's tables would be smaller) but they don't have to.

There is an implication for network addressing and for network growth. If a Bridge is replaced by a Router, **all IP addresses on one side must be changed.**

Notes:

Indirect Routing

If the network ID portion of the source and destination addresses is not the same then the destination host lies on a different physical network. It is necessary to forward the IP datagram to a gateway, which will route the datagram either to another gateway or to the final destination.

The IP layer needs to identify an appropriate gateway. There are three methods available for determining the correct gateway:

- Table driven routing
- Default routing
- Host-specific routing

Having determined an appropriate gateway (that will be on the same physical network for obvious reasons [see recursion] the IP datagram is forwarded to the gateway by direct routing.

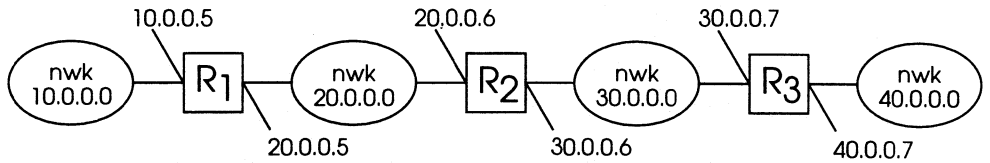
On receipt the gateway will now inspect the destination address to see if it lies on a network to which it is connected. If it does then the packet is forwarded to its destination by direct routing, otherwise a new gateway for onward routing must be determined using one of the three methods.

Note. This process continues until the datagram is delivered - if there is a fault in the routing, or the destination does not exist, the packet could continue forever like the Flying Dutchman. Remember the *Time To Live* (TTL) field of the IP header - this is decremented by one each time the datagram is forwarded by a gateway; when it reaches zero the datagram is discarded and an error message is returned to the source. This is an effective cure for the Flying Dutchman problem.

Notes:

Table-driven Routing

Each host holds an *Internet Routing Table* - this holds information on destination **networks.**



Routing table for router R_1

Reach hosts on network	Route to this address
10.0.0.0	Deliver Direct
20.0.0.0	Deliver Direct
30.0.0.0	20.0.0.6
40.0.0.0	20.0.0.6

Routing table for router R_2

Reach hosts on network	Route to this address
10.0.0.0	20.0.0.5
20.0.0.0	Deliver Direct
30.0.0.0	Deliver Direct
40.0.0.0	30.0.0.7

Routing table for router R_3

Reach hosts on network	Route to this address
10.0.0.0	30.0.0.6
20.0.0.0	30.0.0.6
30.0.0.0	Deliver Direct
40.0.0.0	Deliver Direct

Figure 3.8

For hosts which are also acting as routers, table-driven routing provides each host and router with an *Internet Routing Table* (*/etc/gateways* on UNIX systems) that contains information on destination networks. If data were held on destination hosts (rather than network ids) the tables would become impossibly large to handle. However, a route to a host is possible and desirable in some situations.

Notes:

The table holds network addresses and a routing instruction. The routing instruction either identifies a router device, or identifies that the datagram may be delivered by direct routing. The example shown above for router R2 illustrates that a datagram arriving for network 40.0.0.0 would be passed to router R3, at 30.0.0.7, for onward transmission. Whereas a datagram for network 30.0.0.0 would be delivered directly to the host using direct routing.

This technique is very widely used, but it does have a number of consequences:

1. All traffic heading for a given network takes the same path, multiple paths may not be used concurrently.
2. Only the final router along the delivery route is able to determine if the host exists, or is operational.
3. Each router routes traffic independently, traffic from host A to host B may follow a different path from traffic from host B to host A.

These shortcomings mean that a two-way communication cannot always be guaranteed!

Default Routing

Default routing is a very useful technique for keeping routing tables small (and manageable). Each host establishes a default router for handling traffic to networks that do not appear in the routing table. This is particularly useful for networks that consist of a large volume of local traffic and a single router connection to the outside world. In this case no routing table would be required, as if the datagram could not be routed directly it would be sent to the default (the router).

Notes:

Host-specific Routes

Most IP routing software will allow per-host routes and it can be more convenient to have specific routes for individual hosts for the following reasons:

- Better control for admin
- Could provide security
- Very useful for debugging

Notes:

Routing Algorithm

Both direct and indirect routing are commonly used together. The algorithm illustrated is typical of the routing algorithm normally implemented as a part of IP software by gateways and hosts.

```
Route_IP_Datagram (datagram, routing_table)
        Extract destination IP address, ID, from datagram
        Compute IP address of destination network, In
        if In matches any directly connected network address
                send datagram to destination over that network;
                (This involves resolving ID to a physical address,
                encapsulating datagram, and sending the frame.)
    else if ID appears as a host-specific route
                route datagram as specified in the table;
    else if IN appears in routing table
                route datagram as specified in the table;
    else if a default route has been specified
                route datagram to the default gateway;
    else declare a routing error;
```

Note that we have been looking at the transmission of IP datagrams, and their subsequent routing. When the datagram arrives the host must decide what to do with the datagram - this is because not all gateways are dedicated to that task, there is no reason why a host should not act as a gateway (most UNIX TCP/IP implementations are able to do so). The host has to decide if the datagram is for itself, or for forwarding.

Internet Control Message Protocol

ICMP is an error *reporting*/control message mechanism. It allows hosts or gateways to send error or control messages to other gateways or hosts.

ICMP only reports error conditions to the original source; the source must relate these errors to the individual application programs and take action to correct the problem.

As with all other traffic, ICMP messages travel across the internet in the data portion of IP datagrams. Thus error messages themselves may be lost or discarded.

To avoid infinite regress of messages about messages etc, no ICMP messages are generated about ICMP messages.

Furthermore, ICMP messages are only sent about error handling with fragment zero of fragmented datagrams.

Note that ICMP messages are handled by the IP software, they are not passed to the user application that may have caused the problem. Often the only way you may know that ICMP messages have been received is in an error log audit trail or similar. ICMP was originally designed to allow gateways to notify hosts of non-delivery of datagrams but has since been expanded to provide communication between hosts and between gateways.

Notes:

ICMP Message Types

Type Field	ICMP Message Type
0	Echo reply
3	Destination unreachable
4	Source quench
5	Redirect (change a route)
8	Echo request
11	Time exceeded for datagram
12	Parameter problem on datagram
13	Timestamp request
14	Timestamp reply
15	Information request
16	Information reply
17	Address mask request
18	Address mask reply

ICMP checksum uses the same method as the standard IP checksum, but only applies to the ICMP message.

The IP datagram header plus the first 64 bits of the datagram are always included in error messages. This allows the receiver to determine more precisely the protocols used, the application program responsible for originating datagram etc. Note later that higher level protocols invariably pack the really important information in the 1st 64 bits for this very reason.

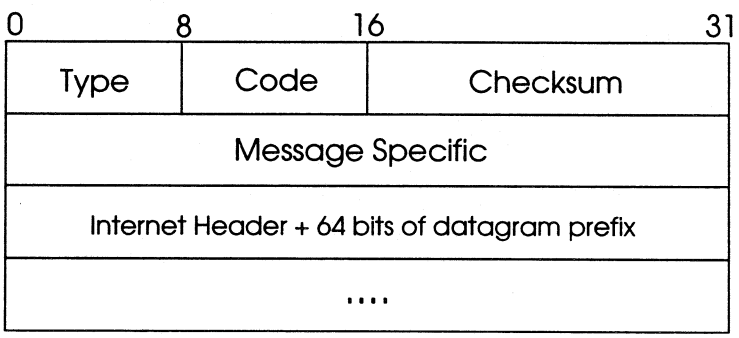


Figure 3.9

ICMP Functions

Some of the main ICMP functions are noted below:

1. A debugging tool most frequently used is the ICMP echo request and echo reply. This allows a host or gateway to send an ICMP echo request message to a specified destination. Any machine that receives an echo request formulates an echo response and returns it to the original source. On most systems this is known as a **PING** or Packet Internet Net Groper.

2. Although gateways send ICMP destination unreachable messages if they cannot route or deliver datagrams, not all such errors can be detected.

3. A source quench message is a request for the source to reduce its current rate of datagram transmission. Usually congested gateways send one source quench message for every datagram they discard.

4. When a gateway detects a host using a non-optimal route, it sends the host an ICMP message called a redirect, requesting that the host change its routes.

5. When the Time To Live (TTL) counter has reached zero, or because a timeout occurred while waiting for fragments of a datagram, it sends an *ICMP time exceeded* message back to the source.

6. Whenever there is a problem with a datagram, not covered by other ICMP error codes, a parameter problem ICMP message is sent to source.

7. A requesting machine sends an ICMP timestamp request message to another machine asking that the machine return its current Time Of Day (TOD) value.

8. To participate in subnet addressing, hosts need to know which bits of the 32-bit IP address correspond to the physical network and which correspond to host identifiers. An ICMP address mask request and reply are used to resolve this.

4

Transport Layer Services

TRAINING

Transport Layer Services

Introduction

In this section we will look at two protocols which provide a transport service across the top of the IP unreliable datagram delivery service. These are User Datagram Protocol and Transmission Control Protocol. Although they both provide a transmission service they are dissimilar, although complementary, protocols.

As well as examining the data formats in some detail, we will examine how TCP provides efficient use of the available network, whilst responding dynamically to variable delays and network congestion.

Objectives

At the conclusion of this section delegates will be able to:

- Describe in detail the UDP and TCP protocol formats
- Describe the operation and use of UDP and TCP.

Notes:

User Datagram Protocol - UDP

So far we have looked at how data can be moved between hosts in the form of IP datagrams. In practice it is not hosts which exchange data but individual application programs executing on the hosts. The term *process* is used to describe an executing application program and most operating systems support multiple processes (e.g. UNIX, VMS).

Processes have some restrictions associated with them:

i) Processes are created and destroyed dynamically, we rarely know the process ID of the remote system.

ii) We do not want to impact other systems if we reboot, and replace all the processes with new ones.

iii) We want to be able to access *functions* on the remote system without necessarily being aware of the underlying process.

iv) A single process may implement a number of functions, we need a mechanism for identifying the required function.

UDP gets around these problems by providing a set of abstract destination points called *protocol ports*. The local operating system is responsible for providing access between the ports and the processes which use them. Arriving messages are queued until a process extracts them. With each message sent, the sender specifies the remote port number and a local port number on which replies will be received.

UDP relies on IP to transport messages between hosts, it therefore provides the same unreliable, connectionless delivery service as IP.

Other limitations of UDP are:

- it does not use acknowledgements to confirm that messages have arrived
- it does not provide a flow control mechanism
- UDP messages can be duplicated, arrive out of order or not at all.

UDP Message Format

A UDP message is called a *user datagram* and contains a header plus user data. The minimum size is 8 octets.

Source Port 16-bit UDP protocol port number for replies from the remote process.

Destination Port 16-bit UDP protocol port for delivery of this message on the remote system.

Length Count of octets in UDP datagram (header and data).

UDP Checksum Optional UDP checksum (0 if not calculated).

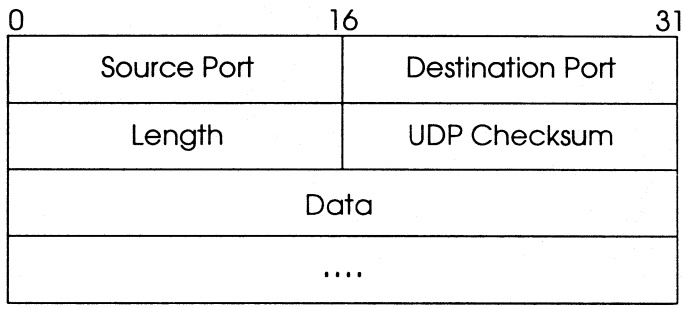


Figure 4.1

The **source port** is often allocated from above 1024 by the OS, while the **destination port** is specified by the protocol the application is using.

Notes:

UDP Demultiplexing

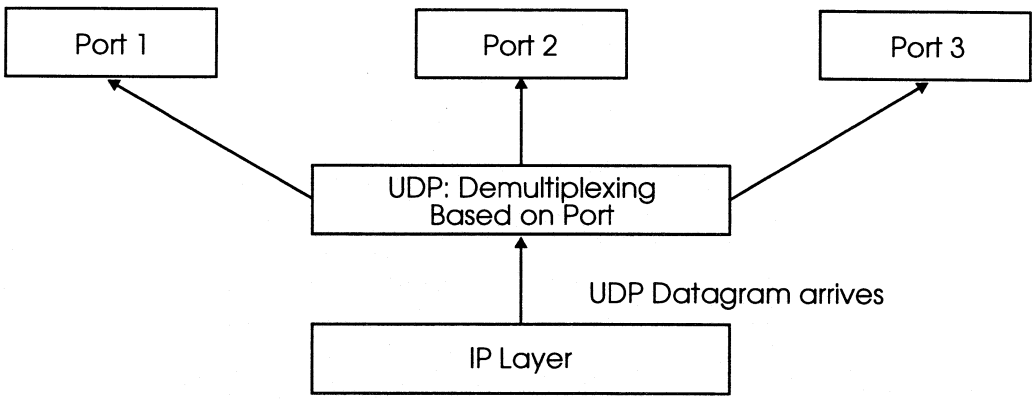


Figure 4.2

The IP layer delivers the complete UDP datagram, as sent by the remote system, to the UDP layer. UDP then arranges for the data portion of the UDP datagram to be delivered to the appropriate port number. The local operating system enables the process to receive the data from the port queue.

Notes:

UDP Well-known Port Numbers

Some UDP protocol ports are pre-assigned for use by specific services:

Port	Keyword	UNIX Keyword	Function
0	-	-	Reserved
7	ECHO	echo	Echo
9	DISCARD	discard	Discard
11	USERS	systat	Active Users
13	DAYTIME	daytime	Daytime
15	-	netstat	Who is up
17	QUOTE	qotd	Quote of the Day
19	CHARGEN	chargen	Character Generator
37	TIME	time	Time
42	NAMESERVER	name	Host Name Server
43	NICNAME	Whois	Who is
53	DOMAIN	nameserver	Domain Name Server
67	BOOTPS	bootps	Bootstrap Protocol Server
68	BOOTPC	bootpc	Bootstrap Protocol Client
69	TFTP	tftp	Trivial File Transfer
111	SUNRPC	sunrpc	SUN Microsystems RPC
123	NTP	ntp	Network Time Protocol
161	-	snmp	SNMP NetMonitor
162	-	snmp-trap	SNMP Traps
512	-	biff	UNIX Comsat
513	-	who	UNIX rwho daemon
514	-	syslog	system log
525	-	timed	Time daemon

Notes:

UDP Checksum Calculation

The IP checksum calculation only includes the IP header, not the data stored in the data segment of the IP datagram. So therefore to validate the data portion a UDP checksum needs to be calculated.

The UDP checksum calculation builds a 12 octet *pseudo header* consisting of:

Source IP	IP address of local system
Destination IP	IP address of remote system
Proto	IP protocol type code for UDP (17)
UDP Length	Length of the UDP datagram (not including pseudo header)

The pseudo header is used to verify that the UDP datagram reaches the correct destination (which is defined as the correct IP address/UDP port combination).

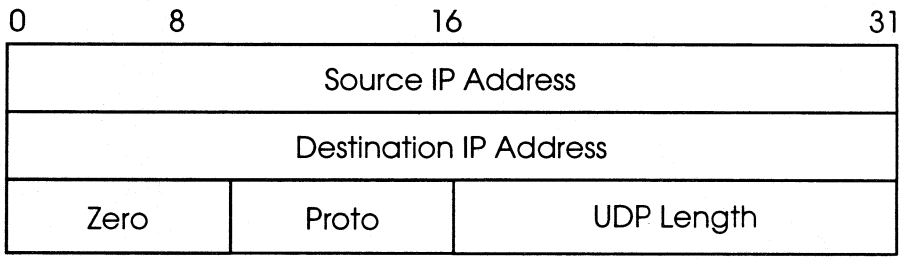


Figure 4.3

It must be noted that the pseudo header and padding is ***not*** transmitted with the UDP datagram, nor are they included in the length.

The degree of interaction required to achieve this means that UDP/IP is rarely implemented as two distinct layers. Often the UDP layer constructs the IP datagram (including addresses and the UDP datagram) before passing this to the IP layer for transmission. This would not be possible in a truly layered environment - it helps to make UDP/IP a fast and efficient protocol in many implementations.

In the final analysis, given the unreliable, connectionless nature of UDP, it is best to regard it as merely a user front end to IP.

Transmission Control Protocol - TCP

The objective of networking is to transfer data between computers, more specifically it is to transfer data between applications which lie on different computers. When the size of the transfer becomes large, it is tedious for the application programmer to build in all the necessary error correction required for use on unreliable, connectionless transport protocols such as UDP. Ideally we want to insulate the application programmer from the underlying network totally, i.e. provide a *pipe*, down which data is passed and where error correction and delivery functions are handled.

TCP is a protocol which seeks to achieve this objective. It provides a uniform interface to a stream transfer service which has the following characteristics:

- Stream orientation
- Virtual circuit connection
- Buffered transfer
- Unstructured data stream
- Full duplex connection

Stream Orientation

Applications transfer data as a stream of bits, with the remote stream delivery service passing the application the same sequence of octets that was passed to the source delivery service.

Notes:

Virtual Circuit Connection

TCP is connection-oriented. A *call* is established to the remote system before a transfer commences and the protocol maintains the link during transfer. This produces the illusion of a *virtual circuit* between the two applications.

Buffered Transfer

Applications may pass data to the protocol software in whatever sizes are most convenient. The protocol collects sufficient data to fill a segment before sending it (to maintain efficiency), there is a *push* mechanism to force transmission. Delivery is at a rate demanded by the remote application, *pushed* data is presented without delay.

Unstructured Data Stream

TCP does not honour structured data streams, applications are responsible for defining and agreeing on the stream format.

Full Duplex Connection

Transfer may take place in both directions concurrently - from an application viewpoint this provides two parallel streams. TCP makes use of this mechanism to *piggyback* control information from one stream back to the source in datagrams carrying data in the opposite direction.

Notes:

TCP Segment Format

The unit of transfer used by TCP is called a *segment*. Segments are used for establishing connections, data transfer, sending acknowledgements, advertising window sizes and terminating connections. Piggybacking of acknowledgements onto packets travelling back to the source reduces traffic and increases efficiency.

A segment consists of a header plus data. The component parts of the header are:

SOURCE PORT	TCP port number for reply
DEST PORT	TCP port number to deliver segment to
SEQUENCE NUM	Position in *sender's* byte stream of the data in the segment
ACK NUMBER	Position of highest byte that sender has received +1
OFF	Offset of data portion of segment
RES	Reserved for future use
FLAGS	Purpose and contents of segment
URG	Urgent
ACK	ACK field valid
PSH	This segment requests push
RST	Reset the connection
SYN	Synchronise sequence numbers
FIN	Sender has reached end of byte stream
WINDOW	Sender's current buffer size
CHECKSUM	16-bit integer checksum of header + data
URGENT PTR	Position in byte stream where urgent data ends
OPTIONS	Data for remote TCP software, esp. MaxSegmentSize

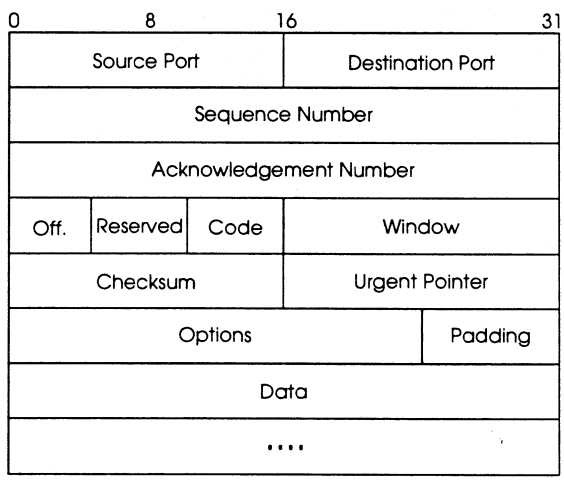
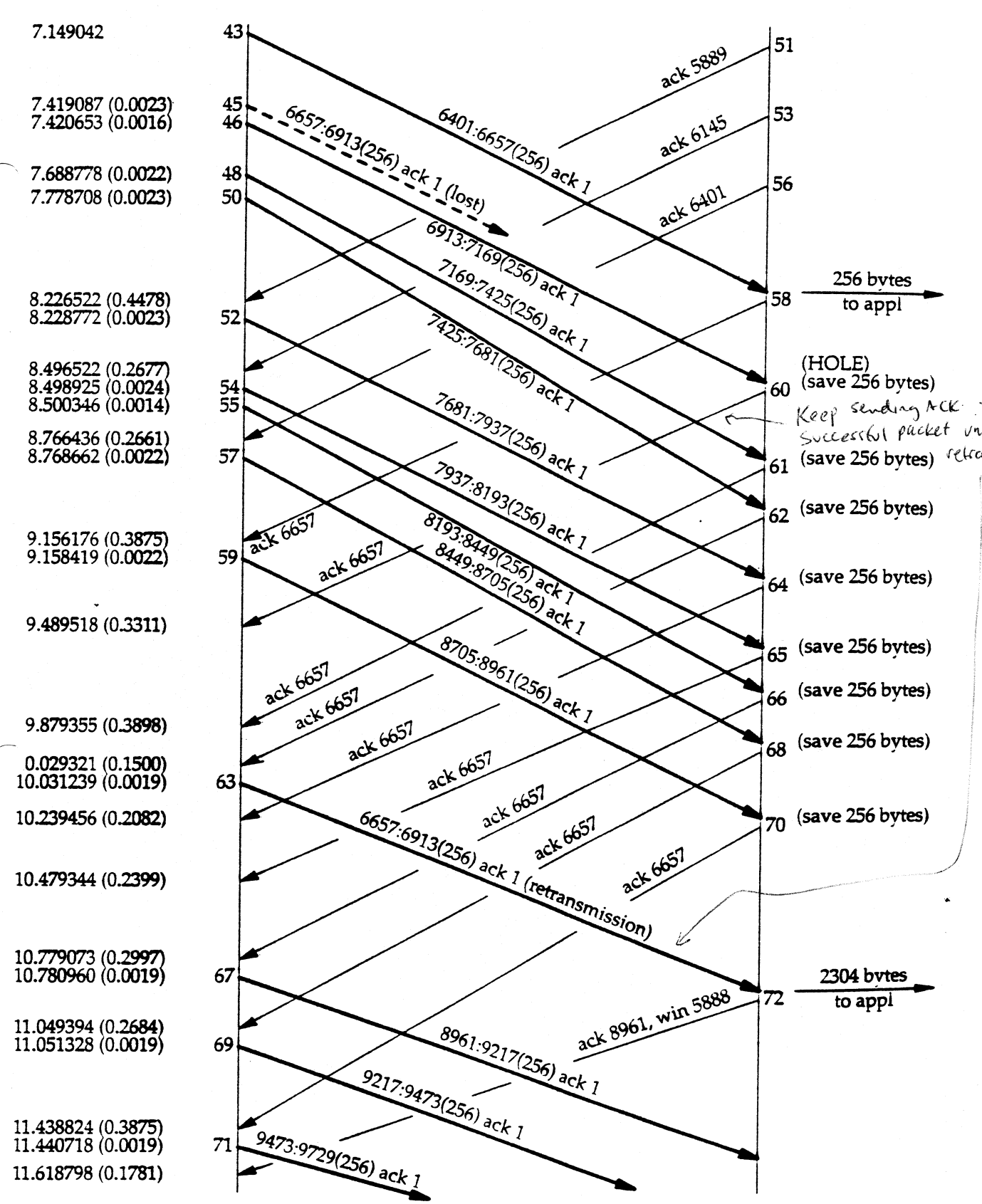


Figure 4.4

TCP Ports

TCP like UDP uses a *port* mechanism to deliver messages to multiple application processes.

Port	Keyword	UNIX Keyword	Function
0	-	-	Reserved
1	TCPMUX	-	TCP Multiplexor
5	RJE	-	Remote Job Entry
7	ECHO	echo	Echo
9	DISCARD	discard	Discard
11	USERS	systat	Active Users
13	DAYTIME	daytime	Daytime
15	-	netstat	Who is up
17	QUOTE	qotd	Quote of the Day
19	CHARGEN	chargen	Character Generator
20	FTP-DATA	ftp-data	File Transfer Protocol
21	FTP	FTP	File Transfer Protocol
23	TELNET	telnet	Terminal Emulation
25	SMTP	smtp	Simple Mail Transport Prot.
37	TIME	time	Time
42	NAMESERVER	name	Host Name Server
43	NICNAME	whois	Who is
53	DOMAIN	nameserver	Domain Name Server
77	-	rje	Any private RJE Application
79	FINGER	finger	Finger
93	DCP	-	Device Control Protocol
95	SUPDUP	supdup	SUPDUP Protocol
101	HOSTNAME	hostnames	NIC Host Name Server
102	ISO-TSAP	isp-tsap	ISO-TSAP
103	X400	X400	X.400 Mail Service
104	X400-SND	x400-snd	X.400 Mail Sending
111	SUNRPC	sunrpc	SUN Microsystems RPC
113	AUTH	auth	Authentication Service
117	UUCP-PATH	uucp-path	UUCP Path Service
119	NNTP	nntp	USENET News Transfer Protocol
129	PWDGEN	-	Password Generation Protocol
139	NETBIOS-SSN	-	NetSBIOS Session Service
160-1023		Reserved	



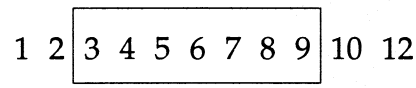
Packet Exchange For Retransmission Around 10 Second Mark

Source: TCP/IP Illustrated - Vol1, Stevens, W. Richard, Addison Wesley, p308

Data Transfer and Flow Control

TCP uses a *sliding window* mechanism to provide efficient data transfer by enabling a number of outstanding packets before acknowledgement is required and to provide end-to-end flow control.

The TCP sliding window mechanism operates at the byte level. Think of the window as a view of the buffer containing the byte stream:



There are three pointers, the first marks the left of the window separating bytes that have been sent and acknowledged. A second pointer marks the right of the window - it defines the highest byte that can be sent before more acknowledgments have been received. The third pointer, inside the window, marks the boundary between bytes which have been sent and those that have not.

The receiver maintains a similar window to put the stream back together again. Remember that TCP is full duplex, so both ends need receive and transmit buffers and windows, although the two are completely separate.

Acknowledgements refer to a position in the byte stream (in fact highest byte received + 1). This is a *cumulative* scheme (reports how much of the stream has been accumulated) which can be inefficient under some circumstances but in general the advantages outweigh the disadvantages.

Window size varies over time providing a flow control mechanism. Each acknowledgement carries a *window advertisement* which specifies how many more bytes can be accepted by the receiver (in effect the receiver buffer size). The sender varies the window size to match the buffer size and can speed up or slow down the transfer rate according to the receiver's capacity to handle the data.

Note that this is an end-to-end flow control mechanism. It does not allow a gateway to slow down the transfer if it becomes overloaded - ICMP Source Quench is used for this.

TCP Connections

TCP is connection-oriented and a connection has to be established before data transfer can take place. A three-way handshake mechanism is used to ensure that the byte streams are correctly synchronised and to guard against possible problems if a connection-establishing packet is lost or duplicated.

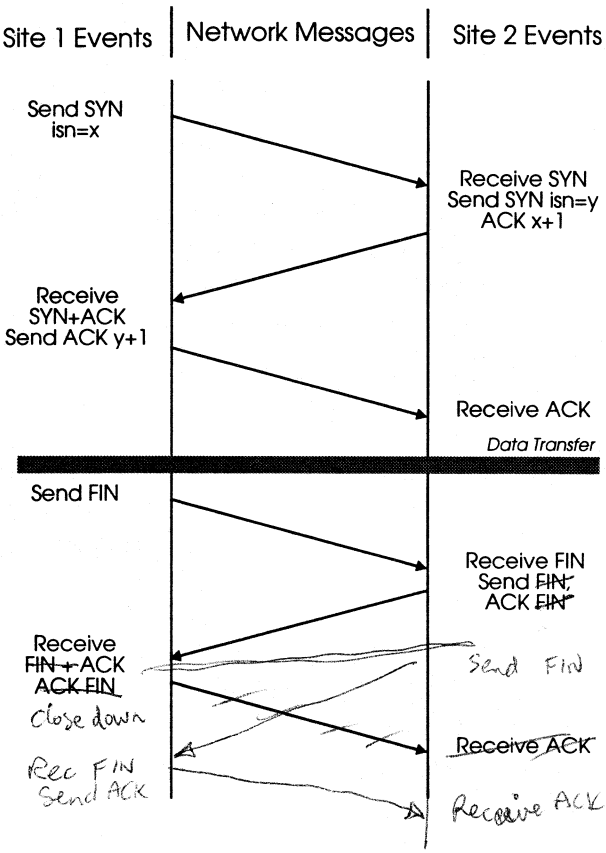


Figure 4.5

Opening

The first segment has the SYN (synchronise) bit set and offers a starting sequence number. The second segment has both SYN and ACK set, confirms the sequence number (+1) and establishes a sequence number for the return data stream. The third segment acknowledges the sequence number. Note that initial sequence numbers do not

Notes: End and FIN's separately ~~not as above~~

normally start at 1. Note also that data can be passed during the handshake, it is not delivered until the handshake is complete. Both parties can advertise their buffer sizes.

Closing

Each direction must be closed independently. The application informs TCP that it has no more data to send, TCP sends the remaining data in a segment with the FIN bit set, the receiver informs its application that no more data is available (EOF) and sends and ACK. Data can continue to flow in the opposite direction until the sender closes the stream.

Reset

If abnormal conditions arise, one side can send a RST segment. This causes the other end to abort the connection, releasing all resources. Applications need to be able to cope with this eventually.

Forcing Delivery

TCP is free to divide the byte stream into segments in the most efficient manner to improve network throughput. This can be a problem for some applications (e.g. Telnet) and so a *push* operation is provided to force delivery. This forces immediate transmission (window permitting) and also sets the PSH bit which gives immediate delivery at the receiver.

An *urgent pointer* can be used to advise the receiver that data further along the stream is urgent and should be dealt with quickly (e.g. the keyboard sequences ctl-S, ctl_Q for Telnet).

Notes:

Time-out and Retransmission

TCP uses a timer for each segment which is transmitted. If an acknowledgement is not received by the time the timer has expired, the original segment is retransmitted. Unlike other positive acknowledgment schemes, the TCP time-out period is varied according to the current *round trip time.*

This facility is provided because of the complex nature of the Internet, with its many gateways. The loading of each gateway is uncertain and so there may be considerable round trip delays, which may vary widely on a per-segment basis. TCP uses an adaptive algorithm which is designed to respond rapidly to changes in round trip times:

$$R = \alpha R + (1 - \alpha)M$$

Where R is the Round Trip Time (RTT) estimator and M is the measured RTT. α is a smoothing factor with a value of 0.9, so it smoothes out changes to the actual RTT. This algorithm has been replaced by a better one in modern TCP implementations.

In addition to variable delays, congestion is also a problem. This occurs when a gateway becomes saturated with packets and is unable to handle them rapidly, the queue of packets at the gateway grows and the delays begin to increase dramatically. At some point (when the gateway runs out of storage space), packets will be dumped. In these circumstances, it is essential that TCP responds by reducing the transmission rate and increasing time-out values - a well designed TCP implementation should be able to do this.

Notes:

TCP Timers

In all protocol implementations, timers are used to monitor the different operations. There are several common timers implemented within TCP.

Persistence Timer

Theoretically, during data exchange, the receive window may become zero. If the next TCP segment sent (intended to reopen the window) is lost at the same time, both TCPs could enter an infinite wait for each other.

The persistence timer is used to send 1 byte TCP segments at set time intervals to test if the receiving party is ready again.

Quiet Timer

After TCP connection shutdown, port numbers are only released after a fixed time interval (quiet time).

This is usually defaulted to 2 x Maximum Segment Lifetime (MSL) where MSL corresponds to the time in the TTL field. The logic here is to prevent possible confusion of connections owing to out of date segments (usually between the same 2 portions).

Keep Alive and Idle Timers

An empty packet is sent at regular intervals to confirm that the connection to the other TCP host still exists. If the partner TCP does not respond, the connection is broken off after execution of the quiet timer.

Notes:

Path MTU discovery. TCP/IP asks routers largest size it can handle. Router cannot say but can say with ICPM if ~~s~~ chosen ~~was~~ caused fragmentation. TCP/IP can then try a smaller size until an ok size is found.

TCP Checksum Calculation

The TCP checksum performs the same functions as the UDP version, i.e. it is used to detect errors in the data and also to determine that the segment has been delivered to the correct IP address/TCP port combination.

The TCP checksum calculation builds a 12 octet *pseudo header.*

SOURCE.IP	IP address of source local system
DESTIN.IP	IP address of remote system
PROTO	IP protocol type code for TCP (6)
TCP LENGTH	Length of the TCP datagram

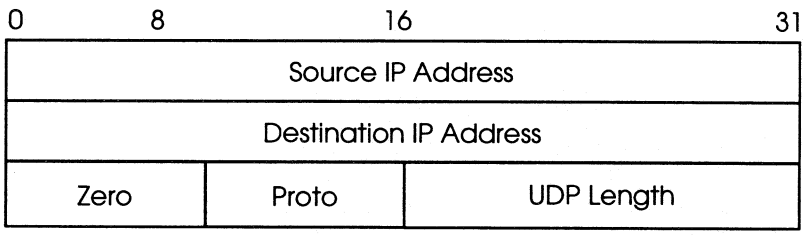


Figure 4.6

The degree of interaction required to achieve this means that TCP/IP is rarely implemented as two distinct layers. There has to be a mechanism to enable TCP to extract data from the delivered IP datagram, and recreate the checksum.

Notes:

5

Routing with TCP/IP

iit TRAINING

Routing with TCP/IP

Introduction

The Internet developed with a wide area network, the ARPANET, already in place. When the Internet experiments began, designers looked at the ARPANET as the main backbone on which to build. Hence the motivation for a series of core gateway systems came from a desire to connect local networks to the ARPANET.

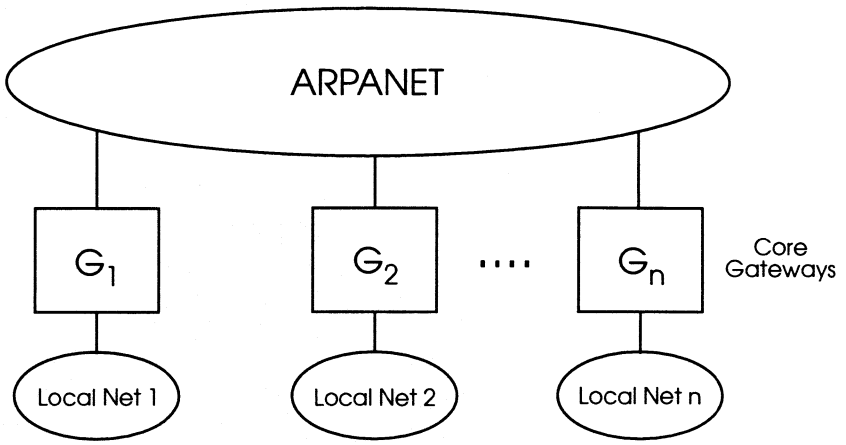


Figure 5.1

This connection process relies on the concept of routing the IP packets between the disparate gateways.

This section addresses the issues of TCP/IP routing in detail.

Objectives

At the conclusion of this section delegates will be able to:

- Describe the difference between an exterior gateway protocol and an interior gateway protocol
- Describe the EGP routing protocol
- Describe the RIP routing protocol
- Describe the OSPF routing protocol
- Describe the implementation issues of choosing a routing protocol.

Core and Non-core Gateways

Core Gateways

- Controlled centrally by INOC
- Relatively small number
- Provide reliable, consistent and authoritative routes for all destinations
- Exchange routing information with other core gateways.

Non-core Gateways

- Controlled and administered by individual groups
- Do not hold routes to all destinations.

Notes:

Gateway to Gateway Protocol - GGP

Proprietary routing protocol used by core gateways. GGP designed to be encapsulated within IP datagrams the same way as UDP and TCP.

GGP is a true vector distance protocol where a gateway using GGP advertises the networks it can reach and the cost for reaching them.

GGP measures this distance in hops where a gateway is defined as zero hops from directly connected networks, one hop away from networks reached through one other gateway etc.

Some points to note are as follows:

- GGP used to exchange routing information between core gateways using GGP messages which travel in IP datagrams.
- Protocol is private to core gateways.
- New gateways may be added without disturbing existing gateways. Neighbours propagate routing information to the new gateway, also accept routing information from new gateway.
- Routing information consists of data pairs:

  Network + Distance to network (cost)

- Cost is measured in *hops* – number of intervening gateways to the network.
- Note: The Internet now uses SPREAD. This is a link state protocol. As it is proprietary to the core system and hence core gateways, no authoritative public information exists regarding its structure or implementation.

Notes:

GGP Message Formats

There are four types of GGP messages, each with its own format. The first octet contains a code that identifies the message type.

The format of a GGP routing update message. A gateway sends such a message to advertise destination networks it knows how to reach. Network numbers contain either 1, 2 or 3 octets, depending on whether the network is class A, B or C.

Type — A value of 12 means that this is a Routing Update Message.

Sequence Number — This is used to validate a GGP message. Both sender and receiver must agree on the sequence number before the receiver will accept the message.

Update — A binary value that specifies whether the sender needs an update from the receiver.

Num. Distances — Because GGP groups networks by distance, this field specifies how many distance groups are present in this update.

The last part of a GGP routing update message contains sets of networks grouped by distance. Each set starts with two 8-bit fields that specify a distance value and a count of networks at that distance.

Notes:

If the count specifies *n* networks at a given distance, exact *n* network IP addresses must occur before the next set header. To conserve space, only the network portion of the IP address is included, so network numbers may be 1, 2 or 3 octets long. The receiver must look at the first bits of the network identifier to determine its length.

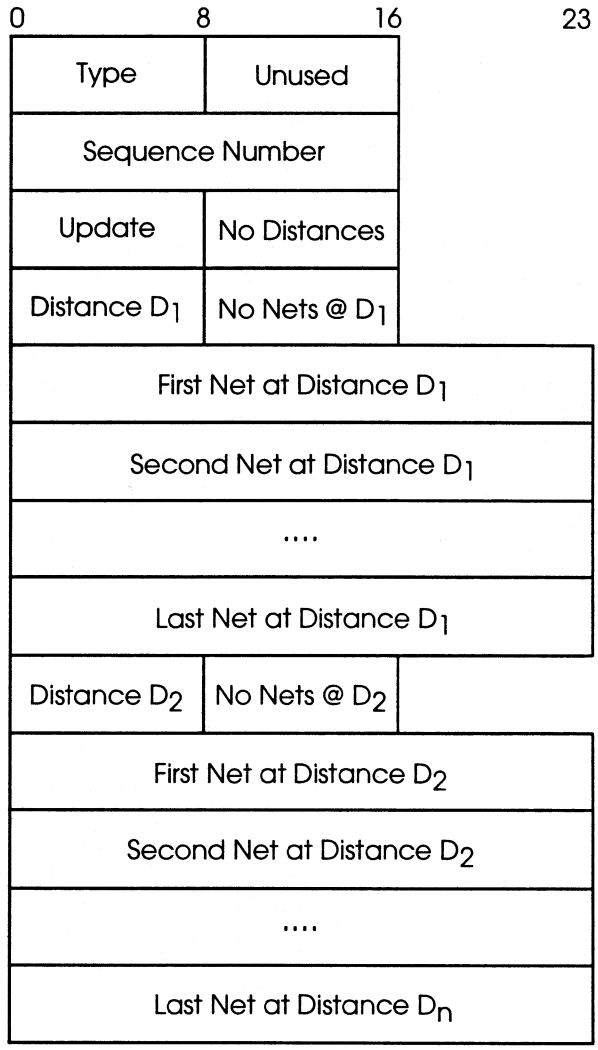


Figure 5.2

GGP Problems

When a gateway receives a GGP routing update message, it sends a GGP acknowledgment back to the sender, using a positive acknowledgement if the routing was acceptable, and a negative acknowledgement if an error was detected.

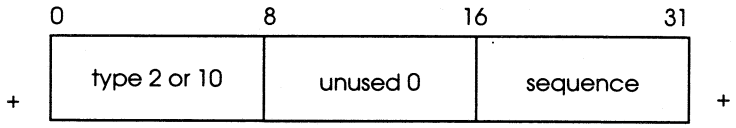


Figure 5.3

The format of a GGP acknowledgement message.
2 = positive, 10 = negative

Sequence - specifies the sequence number that the receiver is acknowledging. In negative acknowledgements, the sequence field gives the last sequence number received correctly.

In addition, the GGP protocol includes messages that allow one gateway to test whether another is responding. A gateway sends an echo request message to a neighbour, which requests that the recipient respond by sending back an echo reply message.

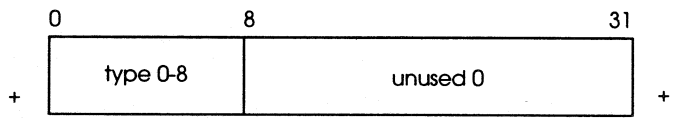


Figure 5.4

Type 8 is echo request while Type 0 is a reply.

Hinden and Sheltzer	RFC-823
Braden and Postel	RFC-1009

Non-core Gateways

Treating a core system as a central router introduces an extra hop for most traffic. A mechanism is needed that allows non-core gateways to learn routes from core gateways so they can choose optimal backbone routes.

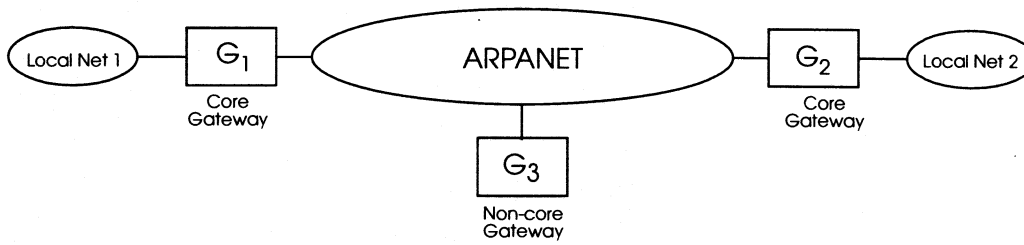


Figure 5.5

Allowing sites to have multiple networks and gateways means that the core does not attach to all networks directly, hence an additional mechanism is needed to allow core systems to learn about them.

Notes:

The points to note about the diagram below are:

- Local network 4 is *hidden* behind local network 1 – the core gateway is not aware of it.
- We need a mechanism for noncore gateways that can update the core gateway with information about local networks and get routing information from the core.
- Need to agree who is responsible for advising the core.

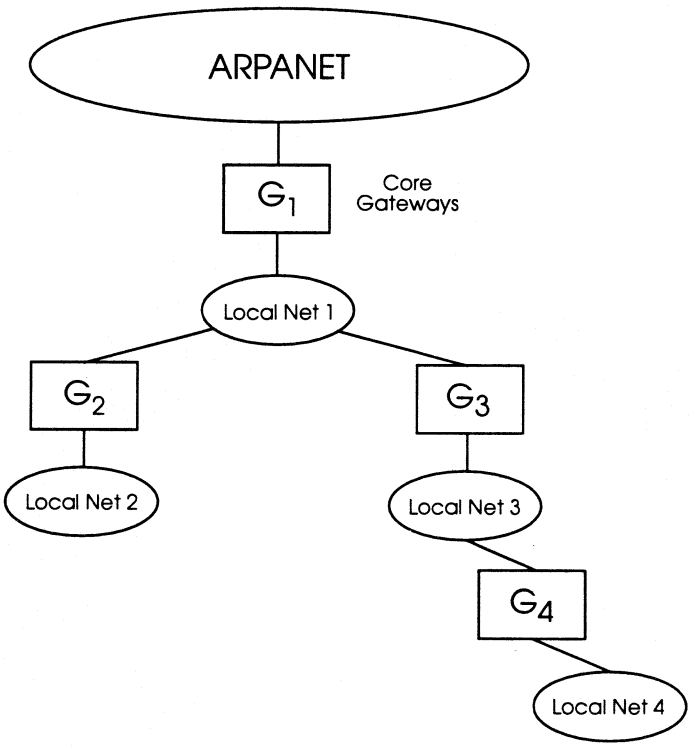


Figure 5.6

Notes:

Autonomous Systems

A large TCP/IP internet has additional structure to accommodate administrative boundaries: each collection of networks and gateways managed by one administrative authority is considered to be a single autonomous system.

An autonomous system is free to choose an internal routing architecture, but must collect information about all its networks and designate one or more gateways that will pass the reachability information to other autonomous systems.

Because the connected Internet uses a core architecture, every autonomous system must pass reachability information to Internet core gateways.

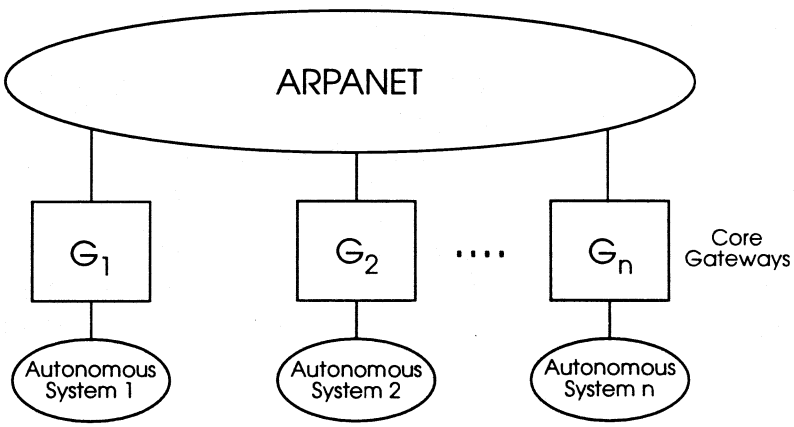


Figure 5.7

Notes:

Figure 5.7 shows that instead of multiple, independent networks connected to the Internet, we have multiple sites with local networks under local control of a single administrative authority.

A group of networks and gateways under control of a single administrative authority is therefore called an *autonomous system.*

Two gateways that exchange routing information are said to be exterior neighbours if they belong to two different autonomous systems.

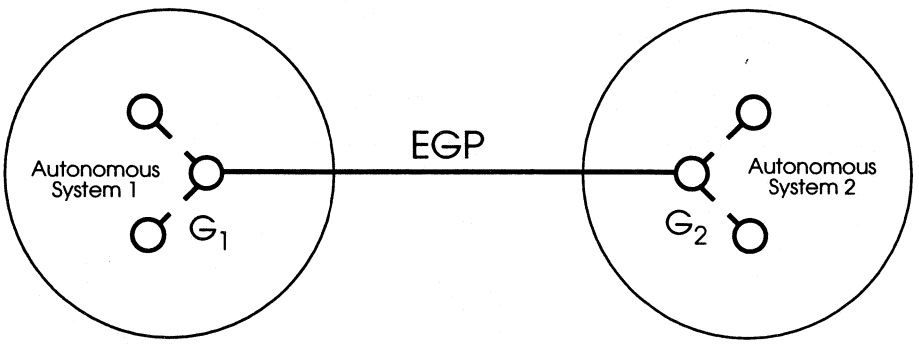


Figure 5.8

Conceptual illustration of two exterior gateways, G_1 and G_2, using EGP to advertise networks in their autonomous systems after collecting the information. As the name implies, *exterior* gateways are usually close to the outer *edge* of an autonomous system.

Notes:

Exterior Gateway Protocol (EGP)

EGP is not a routing algorithm but instead allows external routers in different autonomous systems to exchange network reachability information.

Key RFCs that define EGP are:

Mills	RFC - 904	Formal EGP specification
Rosen	RFC - 827	Early version of EGP
Seamonson & Rosen	RFC - 888	Gateway document
Mills	RFC - 975	Gateway document
Braden & Postel	RFC - 1009	Gateway requirements

Core Gateways

The original core gateway system evolved at a time when the Internet had a single backbone (the ARPANET), surrounded by a core gateway system. The original core architecture was designed to provide connections between local area networks and the backbone. As the Internet grew in complexity, core gateways needed a protocol mechanism to learn about the multiple networks within ASs.

AS Numbers

ASs are assigned 16-bit identification numbers (in much the same way that network and protocol numbers are assigned), and every EGP message contains one word for this number.

Notes:

Configuring EGP
- An autonomous number
- A list of EGP neighbours
- The exchange of routing information between EGP and IGP

Popular protocols
RIP - Routing Information Protocol
Hello Protocol
Gated combinations of RIP, HELLO, EGP
OSPF Open shortest Path First protocol

Neighbours

If two neighbours are not part of the same AS, we call them EXTERIOR neighbours. In order for one system to use another as a transport medium, routers which are exterior neighbours of each other must be able to find out which networks can be reached through the other. EGP enables this information to be passed between exterior neighbours.

Information Exchange

Each AS must implement its own routing algorithm within that system. EGP is *NOT* a routing algorithm. It enables exterior neighbours to exchange information which is likely to be needed by any routing algorithm, but does NOT specify what the routers are to do with this information.

Metrics

When a router learns of a network in another AS, it does not obtain a universally accepted measure of distance.

EGP does not interpret any of the distance metrics that appear in routing update messages. The rules specify that a value of 255 means that network is unreachable, but other values are only comparable if they refer to routers in the same AS.

Notes:

Three Parts of EGP

The EGP protocol is broken down into three parts:

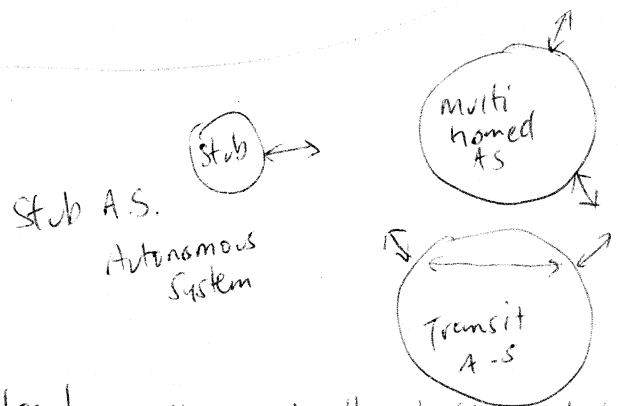
- Neighbour Acquisition
- Neighbour Reachability
- Network Reachability

Note that all messages defined by EGP are intended to travel only a single "hop".

Neighbour Acquisition

Before it is possible to obtain network reachability information from another external router it is necessary to acquire it as a neighbour. The Neighbour Acquisition Protocol uses the following messages:

- Neighbour Acquisition Request
- Neighbour Acquisition Reply
- Neighbour Acquisition Refusal
- Neighbour Cease Message
- Neighbour Cease Acknowledgement



Notes:

Border Gateway Protocol - Allow or disallow traffic into/out of restricted regions
- Policy Based Routing
- Three types of Autonomous systems
  - Stub
  - Multihomed
  - Transit

Neighbour Reachability

It is important for a router to keep real time information as to the reachability of its neighbours. If a particular router cannot be reached, it should cease forwarding traffic to it. The Neighbour Reachability Protocol uses the following messages:

- Hello Messages
- I Heard You Messages

Network Reachability

An exterior neighbour sends Network Reachability messages to convey information about reachable networks to its EGP neighbour.

EGP does not interpret any of the distance metrics that appear in Network Reachability messages.

Notes:

EGP Message Formats

All EGP messages begin with a fixed header that identifies the message type.

Version Contains an integer that identifies the version of EGP used to format the message. Receivers check the version number to verify that their software is using the same version of the protocol.

Type/Code Identifies the type of message, with the code field to distinguish among the subtypes.

Status Contains message dependent status information.

Checksum EGP uses the same checksum algorithm as IP, treating the entire EGP message as a sequence of 16-bit integers, and taking the one's complement of the one's complement sum. When performing the computation, field *checksum* is assumed to contain zeros, and the message is padded to a multiple of 16 bits by adding zeros.

Autonomous Systems Num Contains the assigned number of the autonomous system of the gateway sending the message.

Sequence Number Contains a number that the sender uses to synchronise messages and replies. The neighbour replies with the last sequence number it received, allowing the sender to match responses to transmissions.

EGP Routing Update Message Format All routes are given relative to a specified network. The message lists gateways on that network and the distance of destinations through each. Network addresses contain 1, 2 or 3 octets.

#Int. Gwys Specifies the number of interior gateways appearing in the message.

#Ext. Gwys Specifies the number of exterior gateways appearing in the message.

IP Source Network Gives the network from which all reachability is measured.

EGP routing update messages are a generalisation of GGP routing update messages because they accommodate multiple gateways instead of a single gateway. Thus, the fields of the routing update message following the IP Source Network form a sequence of blocks, where each block gives reachability information for one gateway.

The networks reachable from that gateway are listed along with their distance. Like GGP, EGP groups networks into sets based on *distance*. For each distance, there is a count of networks at that distance followed by the list of network addresses. After the list of all networks at a given distance, the pattern is repeated for all distance values.

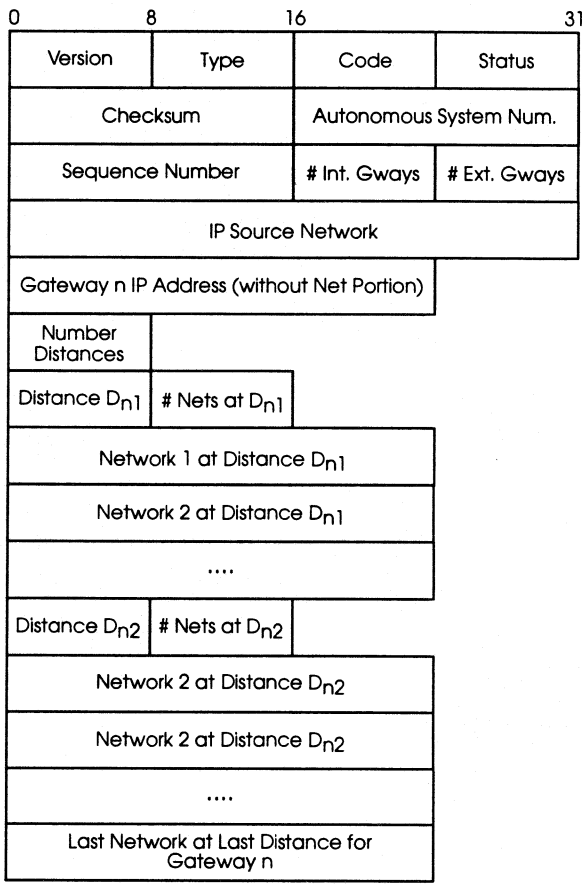


Figure 5.9

EGP Limitations

There are several limitations to EGP:

- EGP only allows noncore gateways to advertise destination networks within their autonomous system.
- EGP does not interpret any distances that appear in routing messages. The distance field indicates that a path exists, not the length of route.
- EGP does not interpret any of the distance metrics that appear in routing update messages.
- Because EGP only propagates reachability information, it restricts the topology of any internet using EGP to a tree structure in which a core system forms the root; there are no loops among other autonomous systems connected to it.

Gateway G_2 has been designated to run EGP on behalf of the autonomous system. It must report reachability to networks 1 through 4. It reports network 1 as reachable through gateway G_1, networks 3 and 4 as reachable through gateway G_3, and network 2 as reachable through G_2. From G_2's perspective, network 2 lies at distance 0. However, it reports network 2 at distance 1, its distance from the source network.

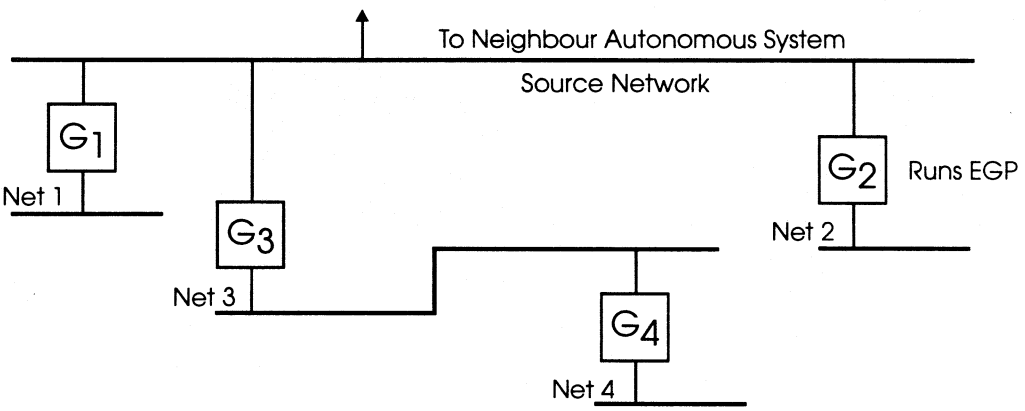


Figure 5.10

Interior Gateway Protocols

EGP propagates information between core gateways and other *external* autonomous systems. *Interior Gateway Protocols* are used for propagating information within an autonomous system.

Two gateways within an autonomous system are said to be interior to one another.

Unlike exterior gateway communication, for which EGP provides a widely accepted standard, no single protocol has emerged for use within an autonomous system. The main reason comes from the varied topologies and technologies used in these autonomous systems. As a result, a number of protocols have become popular, these include:

- Routing Information Protocol (RIP)
- Hello Protocol
- Gated combines RIP, HELLO and EGP
- OSPF (open Shortest Path First protocol)

Because there is no standard, we use the term Interior Gateway Protocol (IGP) as a generic description.

Other Routing Protocols

Cisco developed a proprietary routing protocol - IGRP - for use in its own equipment. This has many advantages over RIP with some features in common with OSPF i.e. the use of link values.

Dual IS-IS is a DEC proposal to the IAB which modifies the information passed by the OSI routing protocol IS-IS to carry TCP/IP routing information as well. This attractive concept would allow IS-IS to work in OSI or TCP/IP or Dual environments. This proposal appears to have been overtaken by the popularity of OSPF in the USA. Those who wish to have dual IS-IS will find it almost unavailable commercially.

Routing Information Protocol - RIP

RIP, Routing Information Protocol, is an Internal Gateway Protocol (IGP), intended for use within an IP-based internet. However, the specific ancestry of this protocol is within the Xerox Network protocols.

Distance Vector Algorithm

RIP advertises the distance to a network or host as the number of "hops" or routers that a datagram must pass through to reach the destination network. A table is built with an entry for every possible destination in the internetwork. The entry contains the distance to the destination as well as the next router on the route to that network.

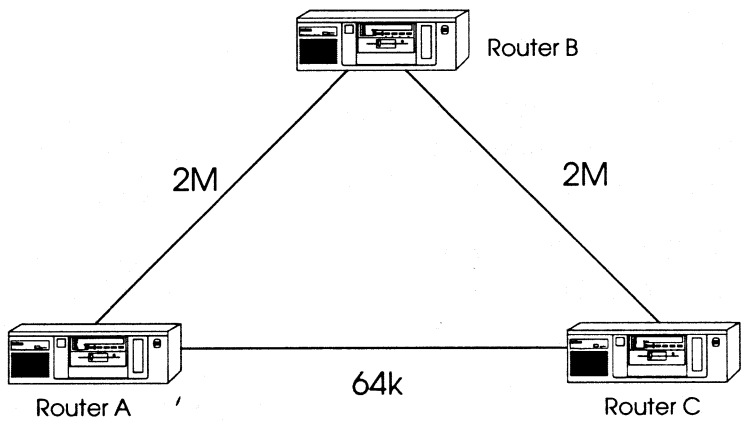


Figure 5.11

Notes:

Hop counts do not calculate optimal results. The best path from Router A to Router C calculated via hop count is over a slow speed line when a more optimal route exists. RIP implementations generally use an artificially high hop count for slow speed lines.

Every thirty seconds participating routers send routing updates to every neighbour. The update is a set of messages that contain pairs, an IP network address and a distance to that network. If we do not hear from a neighbour within 180 seconds the route is marked as invalid.

Active and Passive Modes

RIP partitions participants into *active* and *passive* (silent) machines. Active routers advertise their routes to others. Passive machines listen and update their routes based on advertisements, but do not advertise. Routers run RIP in active mode and hosts run RIP in passive mode.

Address Masks and RIP

When running the RIP protocol, only the subnet masks for directly connected networks are known. Therefore, routes to a subnet must not be sent outside the network of which the subnet is a part.

This filtering is carried out by the gateways at the "border" of the subnetted network. These are gateways that connect that network with some other network. Within the subnetted network, each subnet is treated as an individual network. Routing entries for each subnet are circulated by RIP. However, border gateways send only a single entry for the network as a whole to hosts in other networks.

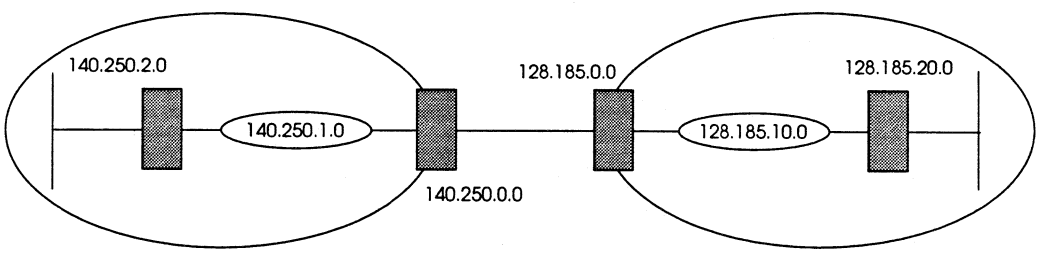


Figure 5.12 Subnet routes are only sent when the destination subnet is a member of the same IP network as the sending address.

RIP Format

RIP was originally implemented by UCB, along with the *routed* daemon, for 4.2BSD which has resulted in it being very widely used in private networks with internal gateways. RIP messages are broadcast into the network at regular intervals, each message consists of a fixed header followed by a (optional) list of reachable networks - the message format looks like this:

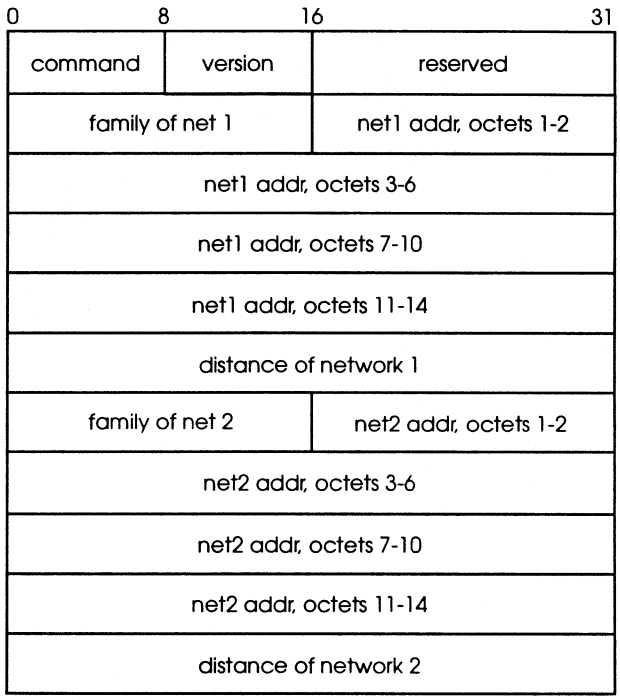


Figure 5.13

Notes:

The *command* field specifies if the packet is a request for routing information (1) or a response (2). Gateways broadcast unsolicited response packets at regular intervals. The RIP message format is not restricted to TCP/IP - *family of net i* identifies the protocol family for which the address should be interpreted. Likewise, the address field will hold up to 14 octets of address data, IP only uses 4 octets and the remainder are unused. *The distance of network* i is expressed in gateway hops. By convention 16 represents infinity, and so RIP is only suitable for relatively small internets.

RIP does not notify of loops, relying instead on the gateways to detect a loop and guard against it. It also takes a long time for routing information to propagate through the network - this means that datagrams can be incorrectly routed.

Notes:

Split Horizon and Poison Reverse

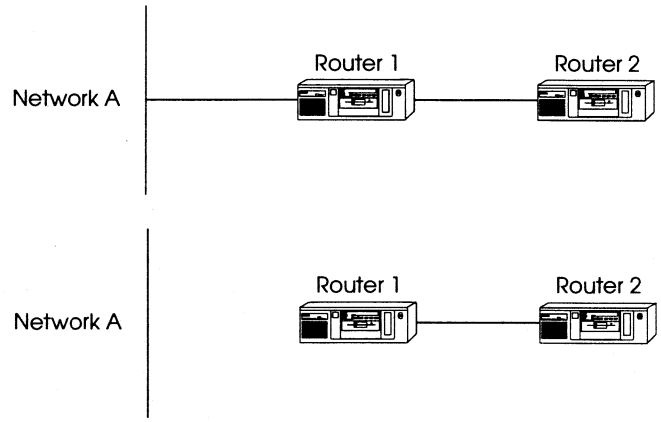


Figure 5.14

RIP does not explicitly detect routing loops. Because of this RIP must take precautions to prevent routing loops. RIP uses a hop count of 16 to indicate an unreachable net or host. This low value reduces the time taken by the whole network to learn of routing changes. However, it also means that RIP cannot be used in large networks.

Split Horizon is a scheme for avoiding routing loops caused by including routes in updates sent to the router from which they were learned. This simple scheme omits routes learned from one neighbour in updates sent to that neighbour.

Split Horizon with Poison Reverse includes such routes in updates but sets their metrics to infinity (16).

Notes:

Limitations

Scalability: RIP was not designed to operate in large internetworks. RIP supports a maximum of 15 hops between destinations. A destination that requires more than 15 consecutive hops is classified as "unreachable". This restricts the maximum size of an Autonomous System to 16 consecutively connected networks.

Hop Count Metric: The use of a hop count metric can cause RIP to fail to select the most efficient and economical path. A routing protocol that uses a hop count metric cannot take into account line speed, reliability, or the delay for any network link.

Wasted Bandwidth: RIP's update mechanism requires frequent broadcasts of a router's entire routing table which can consume a considerable amount of network bandwidth.

Slow Convergence: When a routing link fails it may take up to several minutes for the new routing information to propagate throughout the autonomous system. During this period of time routing inconsistencies can result in the formation of routing loops. Routing loops can cause packets to circulate continuously until the time to live field is exceeded or the routing loop is corrected.

Default Routes

The special address 0.0.0.0 is used to describe a default route. A default route is used when it is not convenient to list every possible network in the RIP updates, and when one or more closely connected routers in the system are prepared to handle traffic to the networks that are not listed explicitly.

The entries for 0.0.0.0 are handled by RIP in exactly the same manner as if there were an actual network with this address. However, the entry is used to route any datagram whose destination address does not match any other network in the table.

Hello Protocol

- HELLO protocol in limited use.
- Unusual in that it does not use hops as a measure of distance. Instead uses delay time.
- Before sending, message is timestamped. On receipt remote gateway checks delay.
- Synchronisation of clocks is crucial.
- Propagation of new routing information must be carefully controlled to avoid instability.

The hello protocol is quite widely used within The Internet, but relatively uncommon outside it in private networks. (It was developed originally with NSFNET backbone.) HELLO uses a routing metric that is based on network delay, rather than hop count. This provides a more dynamic approach to routing that provides a more rapid response to congestion than the static approach adopted by RIP and EGP. A HELLO message looks like this:

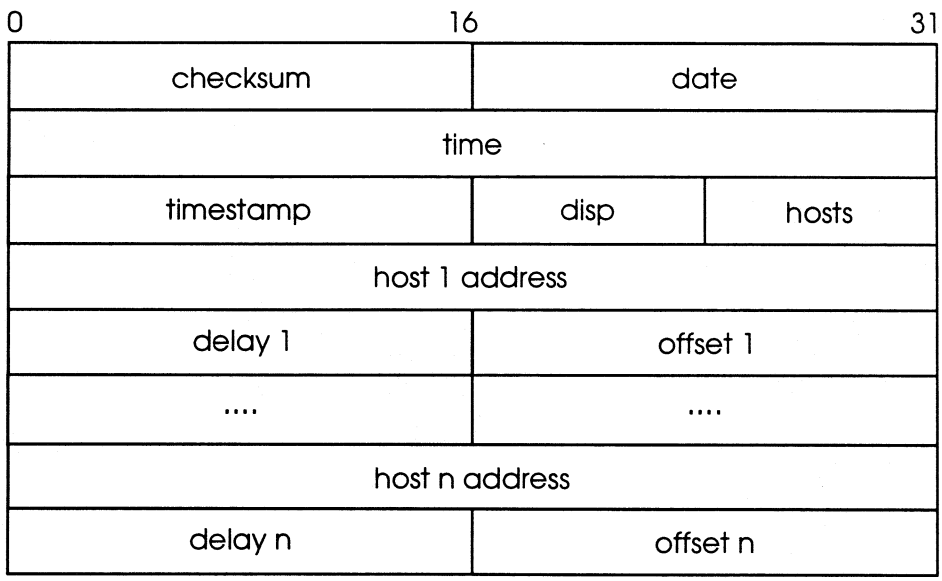


Figure 5.15

Date contains the local date of the sender, *time* is local time according to the sender's clock - the *timestamp* field is used in the computation of the round trip time. *Host* is a count of the number of entries, *disp* indicates the number of entries that are used for the local network. Each entry gives the *delay* to a host and the *offset* between the host and sender's clocks.

Gateways broadcast HELLO messages at regular intervals to their neighbours, who are able to compute the delay by reference to the timestamp field and calculation of the sender's clock error. Gateways can also broadcast their estimates of the delays to reach all known hosts - hence a best fit route can be derived. Note that if a new, quicker route is broadcast then this will become congested and therefore slower. This does create a possibility of instability (**i.e. no load balancing**).

Notes:

route, routed and *gated*

All TCP/IP systems have the capability of specifying routes into other networks - even if they are not connected to other networks. UNIX implementations usually provide the *route* command that allows entries to be made in the system's internal tables to specify new routes, one at a time.

The daemon *routed* reads static route entries (contained in */etc/gateways*) at startup and thereafter maintains the internal routing tables by communicating with other gateways using the RIP protocol.

The daemon *gated* performs a similar function to *routed*, except that its static entries are contained in */etc/gated.cf*. *gated* is able to use RIP, HELLO and EGP protocols to converse with other neighbours and establish routing information. It also provides a facility for the network administrator to identify *trusted* gateways - ones which are unlikely to fail - and hence provide some measure of reliability.

While routed has been the most widely available IGP routing daemon for UNIX systems in the past, gated is more widely available today. If a vendor does not ship it, publicly available versions can be obtained in source code form.

Notes:

Open Shortest Path First

OSPF, Open Shortest Path First, is an internal gateway protocol developed for IP-based internets. It was developed by the OSPFIGP working group of the Internet Engineering Task Force (IETF). This group was co-chaired by John Moy and Mike Petry. The working group was originally created in 1988. In October 1989, John Moy wrote RFC 1131, which defines the proposed specification for the protocol. RFC 1247 was written by John Moy in July 1991 which obsoletes the original RFC 1131.

Flexible Routing Metric

In OSPF, metrics are assigned to outbound router interfaces. The total path cost equals the sum of the path's component interfaces. The actual cost can be assigned by the system administrator to indicate any combination of network characteristics (delay, bandwidth, dollar cost etc.).

Using OSPF in the above example, costs can be assigned so that Router A reaches Router C by travelling through Router B and utilising the two high speed lines rather than the lower 64k line.

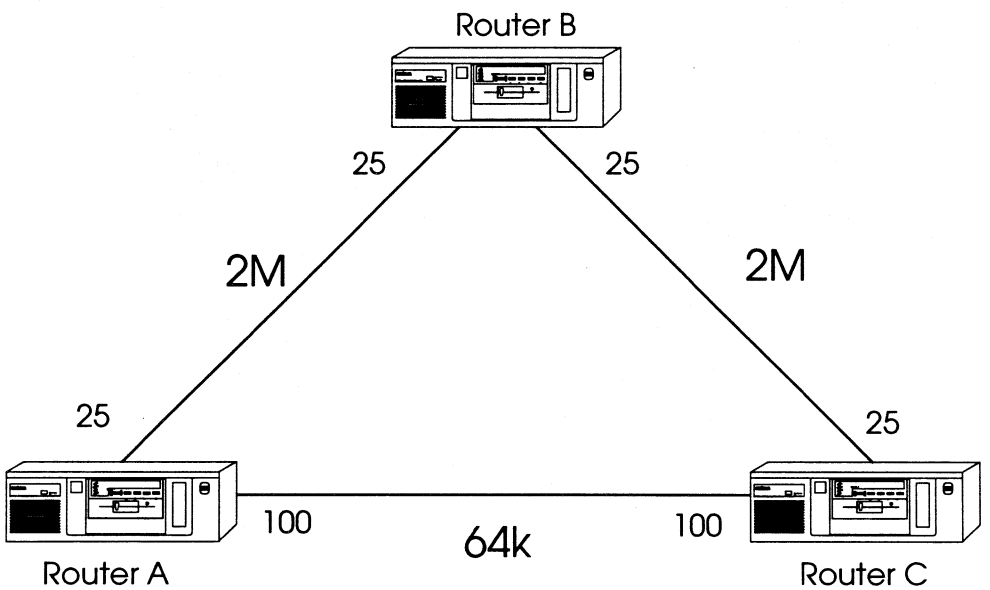


Figure 5.16

SPF Based

In an SPF based routing protocol, each router maintains a database describing the Autonomous System's topology. Each participating router has an identical database. Each router distributes the state of its usable interfaces by flooding.

All routers run the exact same algorithm in parallel. From the topological database, each router constructs a tree of shortest paths with itself as the root. The shortest path tree gives the route to each destination in the autonomous system. The algorithm is attributed to Dijkstra and is called SPF for Shortest Path First.

The routing table is built from the shortest path tree and consists of the collection of best paths to a particular destination. The router will forward to the next hop router based on the routing table.

A cost is configured on the outbound interface of each router. This is configured by the system administrator. The lower the cost, the more likely the interface is used to forward traffic.

Notes:

Internal Routers
Area Border Routers
Backbone Routers
AS Boundary Routers

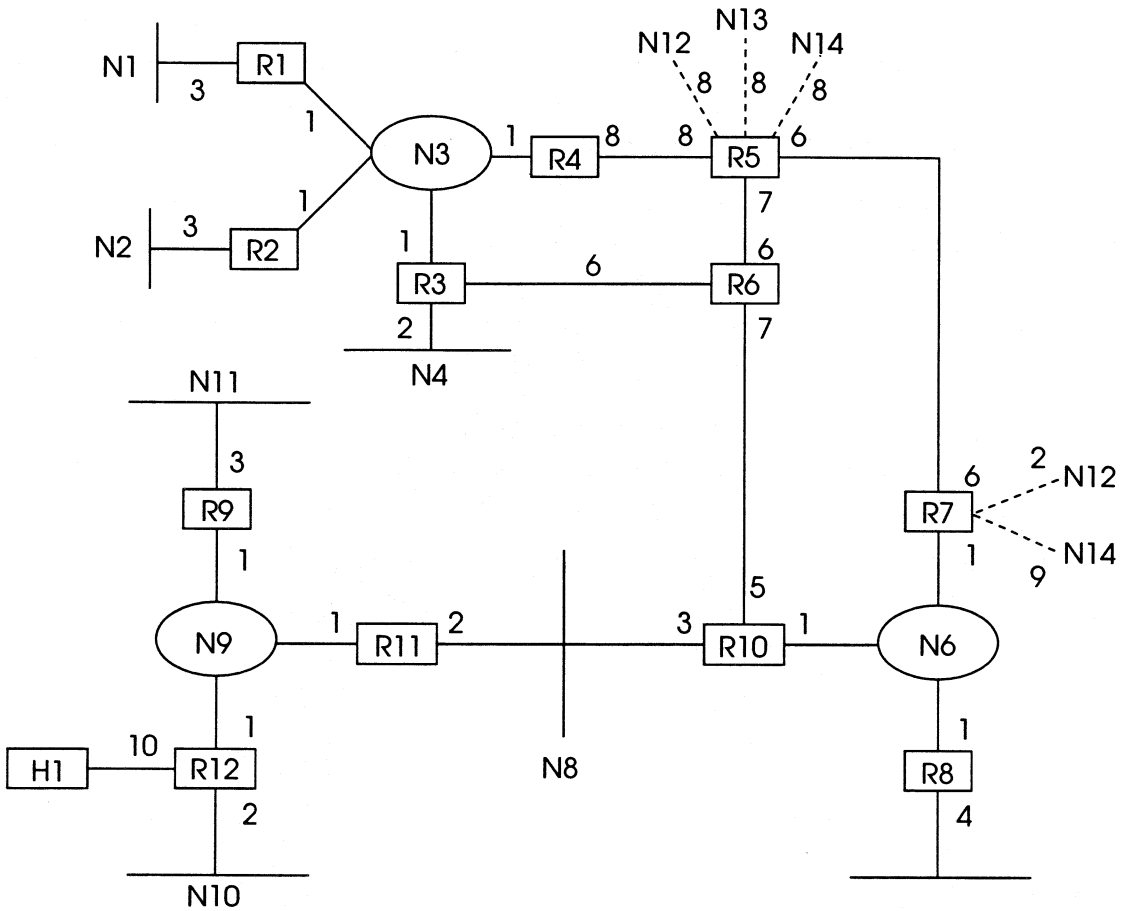


Figure 5.17 A sample Autonomous System

Notes:

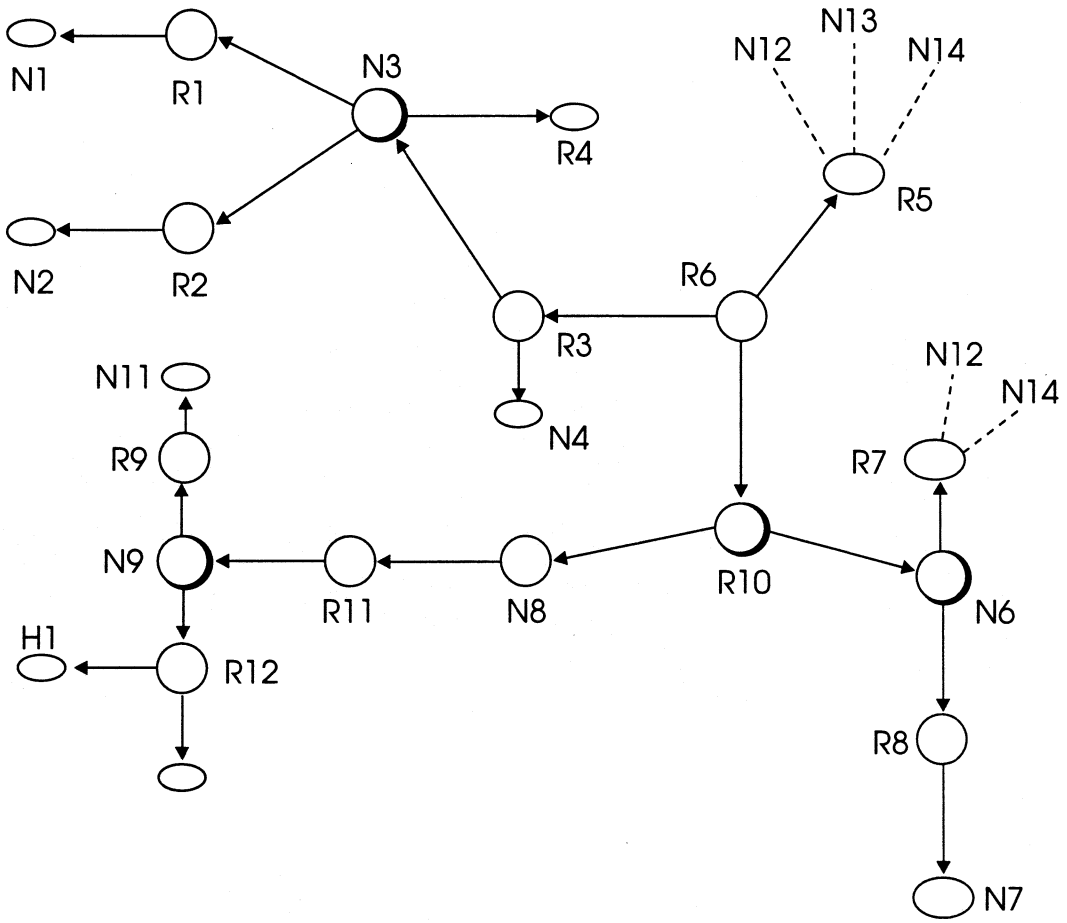


Figure 5.18 The SPF tree for router 6

Notes:

destination	next hop	distance
N1	R3	10
N2	R3	10
N3	R3	7
N4	R3	8
N6	R10	8
N7	R10	12
N8	R10	10
N9	R10	11
N10	R10	13
N11	R10	14
H1	R10	21
R5	R5	6

Figure 5.19

Notes:

Physical Networks

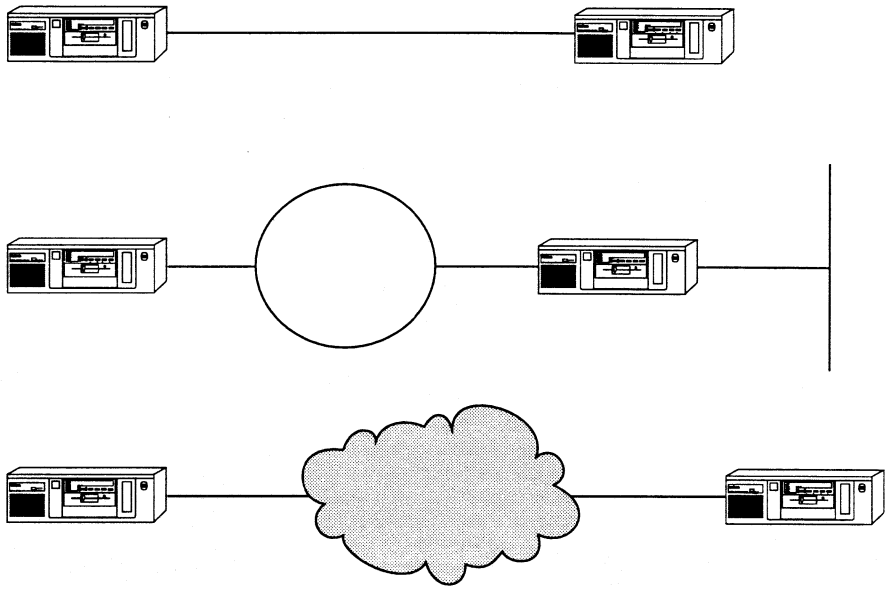


Figure 5.20 Physical Networks

OSPF supports the following types of physical networks:

Point-to-Point Networks: A network that joins a single pair of routers.

Broadcast Networks: Networks supporting multiple routers, together with the capability to address a single physical message to all the attached routers (broadcast).

Non-broadcast Networks: Networks supporting multiple routers but having no broadcast capability.

Notes:

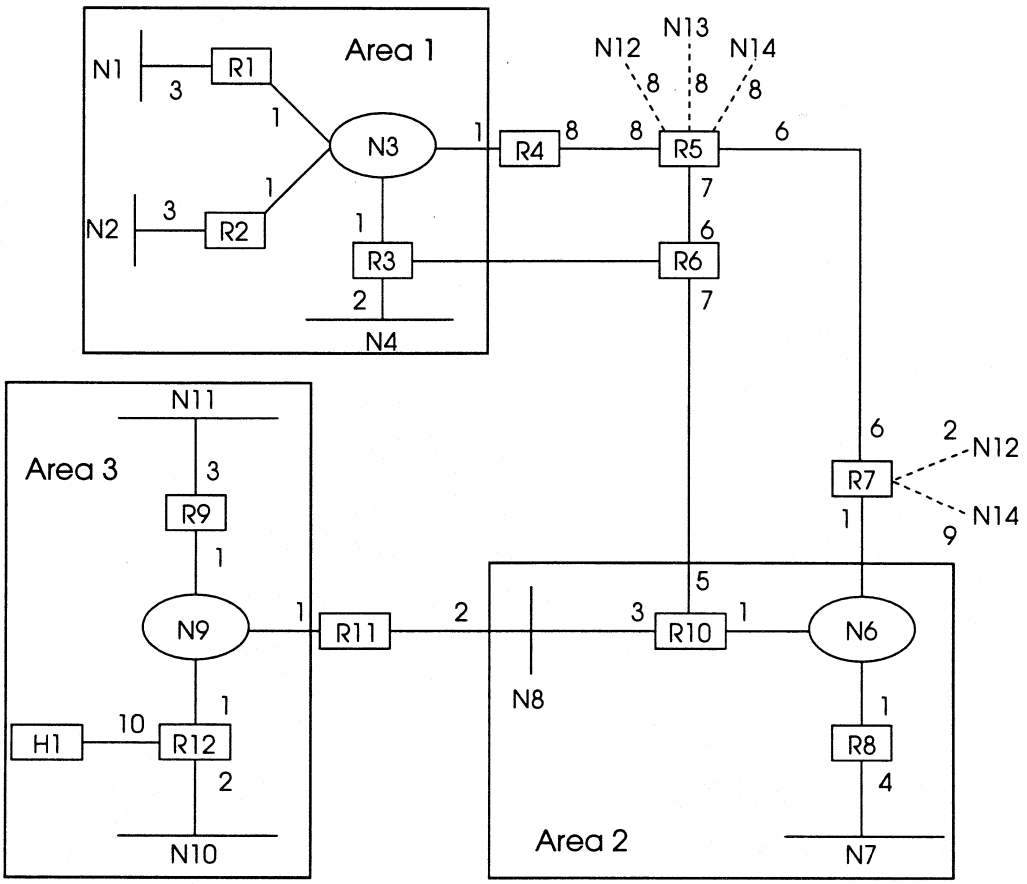


Figure 5.21

OSPF allows collections of contiguous networks and hosts to be grouped together into Areas. Each Area runs a separate copy of the basic SPF algorithm and has its own topological database.

Notes:

The Backbone

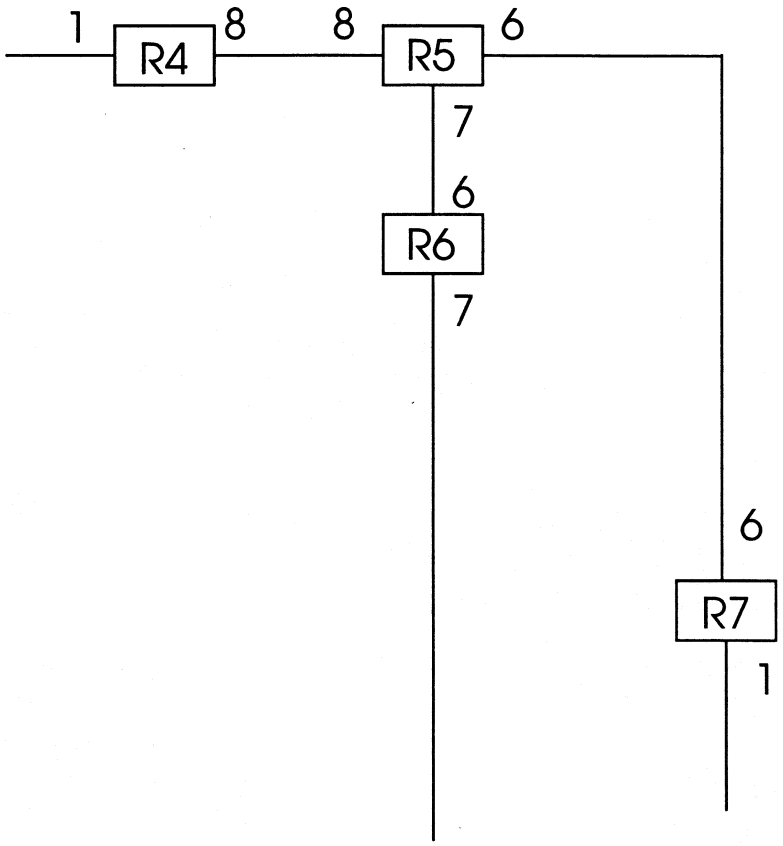


Figure 5.22

The *Backbone* consists of those networks not contained in any area. The backbone is responsible for distributing routing information between areas.

The backbone must be contiguous. It is possible to define areas in such a way that the backbone is no longer contiguous. In this case the system administrator must restore connectivity by configuring virtual links.

Router Classifications

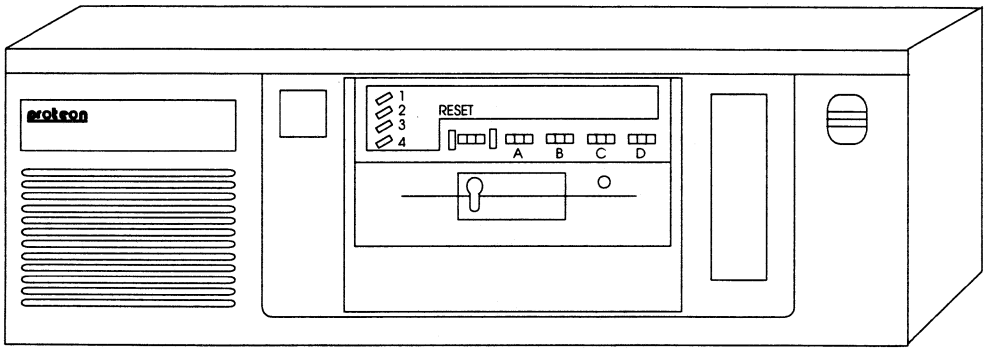


Figure 5.23

Internal Routers: A router with all directly connected networks belonging to the same area. These routers run a single copy of the basic routing algorithm.

Area Border Routers: A router that attaches to multiple areas. Area border routers run multiple copies of the basic routing algorithm, one copy for each attached area. Area border routers condense the topological information for their attached areas and distribute it over the backbone to the other areas.

Backbone Routers: A router that has an interface into the backbone.

AS Boundary Routers: A router that exchanges information with routers from other autonomous systems.

Notes:

Virtual Links

Virtual links serve to connect separate components of the backbone. The two end points of a virtual link are area border routers. Virtual links may be configured between any two area border routers that share a common non-backbone and non-stub-area. The area that they share is referred to as the *transit area.*

The virtual link is treated as if it were a point-to-point network belonging to the backbone.

An attempt is made to establish adjacency over the virtual link. When adjacency is established, the virtual link will be included in backbone router links advertisements and OSPF packets pertaining to the backbone area will flow over the adjacency.

The cost of the virtual link is defined as the cost of the intra-area path between two area border routers.

Virtual links must be configured in each of the links two endpoints.

Notes:

Stub Areas

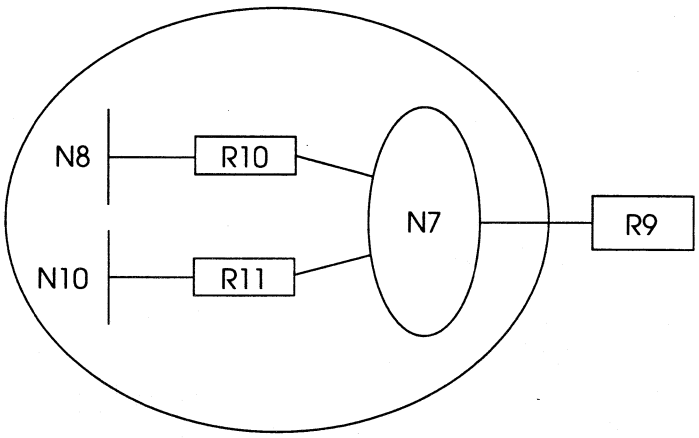


Figure 5.24

An area can be configured as a stub area when there is a single exit point from the area. External advertisements are not flooded into or throughout stub areas. Routing to AS external destinations in these areas are based on a per area default. This reduces the topological database size which reduces the memory requirements for a stub areas internal routers.

The stub area's Area Border Router must advertise a default route for any external destination. All routers belonging to an area must agree when the area has been configured as a stub.

Virtual links cannot be configured through stub areas. AS Boundary routers cannot be placed internal to stub areas.

Notes:

Variable Length Subnet Masks

OSPF attaches an IP address mask to each advertised route. Including the mask with each advertised destination enables the implementation of variable length subnet masks.

The network 128.185.0.0 needs to be broken up into variable sized subnets:

- 14 subnets of size 4k 16 x 4096
- 10 subnets of size 200 256 x 256
- 500 subnets of size 100 512 x 128

Define the resulting network masks:

- 255.255.240.0
- 255.255.255.0
- 255.255.255.128

Notes:

Functional Summary

OSPF routers use the *Hello protocol* to acquire neighbours. A router sends Hello packets to its neighbours and receives their Hello packets. The router will attempt to form adjacencies with its newly acquired neighbours. Topological databases are synchronised between pairs of adjacent routers.

Adjacencies control the distribution of routing protocol packets. Routing protocol packets are sent and received only on adjacencies.

A router periodically advertises its state, or link state. Link state is also advertised when a router's state changes.

Link State Advertisements are flooded throughout the area. The flooding algorithm is reliable, ensuring that all routers in an area have exactly the same topological database. The database consists of the collection of link state advertisements received from each router. From this database each router calculates a shortest path tree with itself as the root. This shortest path tree yields a routing table for the protocol.

Notes:

Routing Protocol Packets

The OSPF protocol runs directly over IP using protocol # 89. OSPF defines five types of routing protocol packets:

Hello Packets: OSPF's Hello protocol uses Hello packets to discover and maintain neighbour relationships.

Database Description Packets: Two routers forming adjacency will summarise and share their database contents using Database Description Packets.

Link State Request Packets: When two routers are synchronising their databases by exchanging database description packets and one needs specific advertisements to be updated, a link state request packet is sent.

Link State Update Packets: Link State Update Packets carry a set of new link state advertisements one hop further away from their point of origination.

Link State Acknowledgements: These packets provide reliability in the flooding procedure.

Notes:

The Hello Protocol

The Hello protocol is responsible for establishing and maintaining neighbour relationships. It also ensures that communication between neighbours is bidirectional. Hello packets are sent out periodically on all interfaces. Bidirectional communication is indicated when the router sees itself in the neighbour's Hello packet.

On multi-access networks, the Hello protocol elects a Designated Router for the network.

Forming Adjacency

Adjacency should be established with a neighbour when one of the following conditions hold:

- The network is point-to-point.
- The network is a virtual link.
- The router is the Designated Router.
- The router is the Backup Designated Router.
- The neighbouring router is the Designated Router.
- The neighbouring router is the Backup Designated Router.

Notes:

The Synchronisation of Databases

Adjacent routers are required to synchronise databases. The synchronisation process begins during adjacency forming. Each router describes its database by sending a sequence of *Database Description Packets* to its neighbour. Each Database Description Packet describes a set of link state advertisements belonging to the database. When a neighbour sees a link state advertisement more recent then its own, it requests the advertisement.

In order to send Database Description Packets, the routers must decide upon an initial sequence number and a Master router. The router with the highest Router ID becomes the Master and provides the initial sequence number. The Master sends Database Description Packets which must be acknowledged by the Slave.

Based on the summary received from its neighbour, each router builds a list of requests for LSAs that it needs to bring its own database up to date. Each router will send this list in a *Link State Request packet* to its neighbour.

Notes:

Neighbour States

Neighbour states convey the state of a conversation being held with a neighbouring router.

Down: This is the initial state of a neighbour conversation. There has been no recent information received from the neighbour *(State 1).*

Attempt: This state is only valid for neighbours attached to non-broadcast networks. No recent information has been received from a neighbour but more effort should be made to contact the neighbour by sending Hello packets *(State 2).*

Init: A Hello packet has been recently seen from the neighbour but bidirectional communication has not been established with the neighbour *(State 4).*

2-Way: Communication between the two routers is bidirectional *(State 8).*

ExStart: The Master and initial sequence number is decided upon in this state. Neighbour conversations in this state or greater are called adjacencies *(State 16).*

Exchange: The router is describing its entire link state database by sending Database Description packets to the neighbour *(State 32).*

Loading: Link State Request packets are sent to the neighbour asking for the more recent advertisements that have been discovered but not yet received *(State 64).*

Full: The neighbouring routers are full adjacent. These adjacencies will now appear in router links and network links advertisements *(State 128).*

Notes:

The Adjacency Forming Process

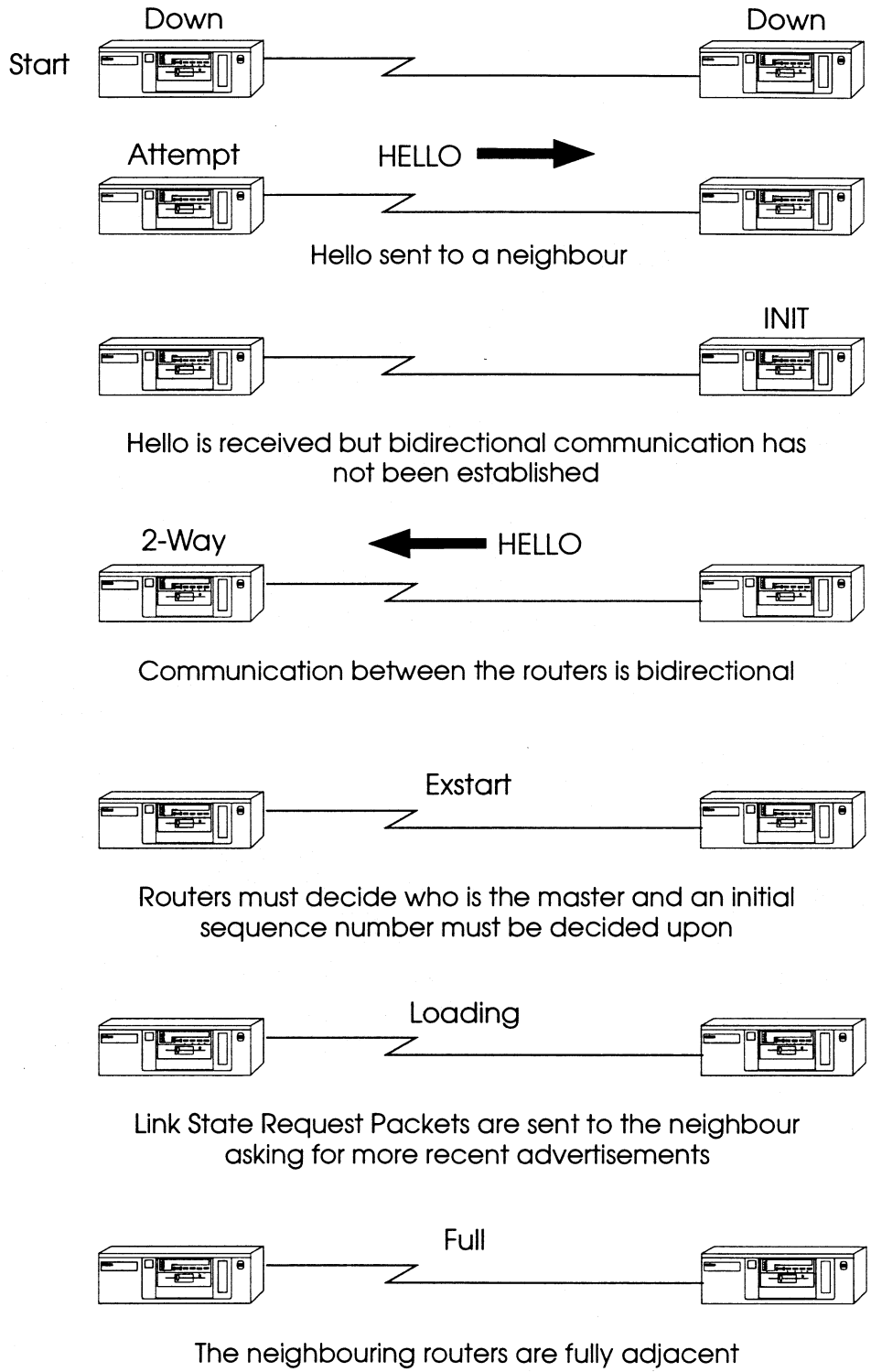


Figure 5.25

The Designated Router

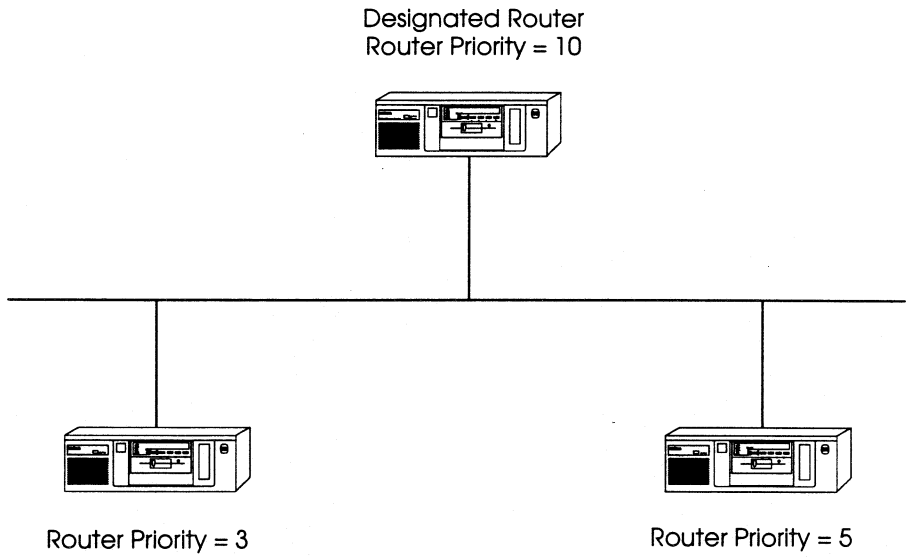


Figure 5.26

Every multi-access network has a Designated Router. The Designated Router performs two main functions:

1. The Designated Router becomes adjacent to all other routers on the network. Since link state databases are synchronised across adjacencies, the Designated Router plays a central part in the synchronisation process.

2. The Designated Router originates a Networks Links Advertisement on behalf of the network listing the set of routers currently attached to the network.

There is also a Backup Designated Router forms adjacencies to all routers on the network and becomes the Designated Router when the present one fails. The Backup Designated Router does *NOT* originate a Network Links advertisement.

The advantage of electing a Designated Router is that it reduces the amount of routing information traffic on the network.

Router Priority

The Designated Router is elected by the Hello protocol. A router's Hello packet contains its Router priority which is configurable on a per interface basis. The router with the highest priority becomes the designated router unless there is already a designated router.

If all priorities are the same, the router with the highest Router ID becomes the Designated Router for the network. If a router is not eligible to become Designated Router his priority is configured to be 0.

Notes:

Processing Hello Packets

Hello packets are sent out periodically on all interfaces (including virtual links) in order to establish and maintain neighbour relationships.

All routers connected to a common network must agree on certain parameters:

Network Mask: The network mask associated with this interface.

Hello Interval: The number of seconds between this router's Hello packets.

Dead Interval: The number of seconds before declaring a silent router down.

Authentication Key: A password which may be configured on a per area basis.

These parameters are included in Hello packets, so that differences can inhibit the forming of neighbour relationships. *When configuring your OSPF Router, care must be taken that these parameters are consistent.*

Notes:

Link State Advertisements

A router periodically advertises its state, or link state. Link state is also advertised when a router's state changes. A router's adjacencies are reflected in the contents of its link state advertisements. The relationship between adjacencies and link state allow the protocol to detect dead routers in a timely fashion.

Link state advertisements are flooded throughout the area. The flooding algorithm is reliable, ensuring that all routers in an area have exactly the same topological database. The database consists of the collection of link state advertisements received from each router belonging to the area. From this database each router calculates a shortest path tree with itself as the root. This shortest path tree yields a routing table for the protocol.

Notes:

LS Type	Advertisement Name	Description
1	Router Links Advertisement	Originated by all routers. Describes the collected states of the routers interface to an area.
2	Network Links Advertisement	Originated for multi-access networks by the Designated Router. Contains the list of routers connected to the network.
3, 4	Summary Link Advertisement	Originated by area border routers and flooded throughout their areas. Describes a route to a destination outside of the area. Type 3 describes routes to networks. Type 4 describes routes to AS Boundary routers.
5	AS External Link Advertisements	Originated by AS Boundary Routers and flooded throughout the AS. Describes a route to a destination in another AS.

Figure 5.27

Notes:

The Flooding Procedure

Link State Update packets provide the mechanism for flooding Link State Advertisements.

A Link State Update packet may contain several distinct advertisements and floods each advertisement one hop further from its point of origination.

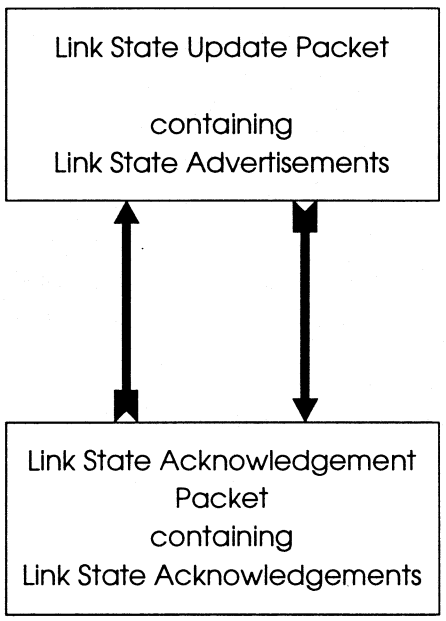


Figure 5.28

Each advertisement must be acknowledged separately. Many acknowledgements may be grouped together into a single packet.

Notes:

IP Multi-cast Addressing

OSPF makes use of IP multi-cast addressing. An IP multi-cast address is implemented using a Class D IP address.

The IP addresses on OSPF protocol packets are set as follows:

- On physical point-to-point networks, the IP destination is always set to the multi-cast address ALLSPFRouters, **224.0.0.5.**
- On broadcast networks, Hello packets are sent to the multi-cast destination ALLSPFRouters, **224.0.0.5**.
- The Designated Router and Backup Designated Router send both Link State Update packets and Link State Acknowledgement packets to the multi-cast address ALLSPFRouters **224.0.0.5**.
- All other routers on the broadcast network send both Link State Update and Link State Acknowledgement packets to the multi-cast address ALLDRouters, **224.0.0.6**.

On all other network types, including virtual links, OSPF packets are sent to the neighbour IP address associated with the other end of the adjacency.

Notes:

Originating Link State Advertisements

Whenever a Link State Advertisement is originated:

- The Link State Sequence Number is incremented.
- The Link State Age is set to 0.
- The Link State Checksum is calculated.
- The advertisement is added to the Link State Database.
- The advertisement is flooded out the appropriate interfaces.

Seven events that originate a Link State Advertisement:

1. The LS refresh timer firing.
2. The interface state changes.
3. The Designated Router changes.
4. A neighbouring router changes to/from the FULL state.
5. An intra-area route has been added/deleted/modified in the routing table.
6. An inter-area route has been added/deleted/modified in the routing table.
7. An external route changes.

Notes:

Receiving Link State Advertisements

Consistency checks are made on the received packet before being handed to the flooding procedure. For each link state advertisement sent out in the packet, the following steps are taken:

- Validate the packet's Link State Checksum.
- Examine the Link State Advertisement's LS type.
- If this is an AS External Advertisement (type 5), and the area has been configured as a stub area, discard the advertisement and get the next one from the LS Update packet.
- Find the instance of this advertisement that is currently contained in the router's LS database. If there is no database copy, or the received one is more recent then the following steps must be performed:
  - If there is already a database copy and it was installed less than MinLSInterval seconds ago, discard the new advertisement.
  - Otherwise immediately flood the new advertisement out the router's interfaces
  - Install the advertisement in the Link State Database (replacing the current copy).
  - Send a Link State Acknowledgement packet out the receiving interface.

Notes:

Calculating the Routing Table

The routing table build process consists of the following steps:

- The present routing table is invalidated.
- Intra-area routes are calculated by building the SPF tree.
- The inter-area routes are calculated through examination of summary link advertisements.
- For routing entries whose next hop is over a virtual link, a real (physical) next hop is calculated.

Notes:

6

TCP/IP and the Application Level Protocols

iit TRAINING

TCP/IP and the Application Level Protocols

Introduction

In this section we look at some of the application protocols which are commonly implemented along with TCP/IP. To many users, these applications ***are*** TCP/IP as this is the part that they most commonly interact with.

In the context of this section, the *applications* that we are referring to are specialised communications applications, file transfer, remote terminal etc. They are capable of providing the Session, Presentation and Application functions of the OSI Reference Model.

Referring back to our simple *3-layer* view of TCP/IP, this section is concerned with applications services.

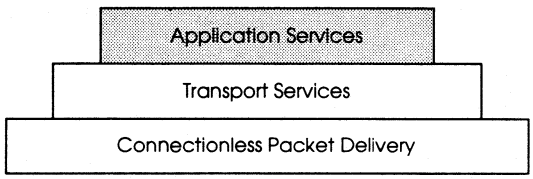


Figure 6.1

Objectives

At the completion of this section delegates will:

- Be familiar with key TCP/IP based applications
- Be able to describe the operation of FTP
- Be able to describe the operation of Telnet
- Understand the key flows between TCP and the application level protocols.

Virtual Terminal Protocol - Telnet

Telnet is a simple virtual terminal protocol which is widely implemented on many operating systems - what it lacks in sophistication is made up for by wide availability. The objective of Telnet is to enable a user, connected to one host, to be able to access a second host, via the network, as if he was locally connected.

Ideally we want our virtual terminal protocol to be able to *pass through* everything we type at our terminal to the remote host, without being upset by funny control codes or issues of the local operating system. The ideal situation would look something like this:

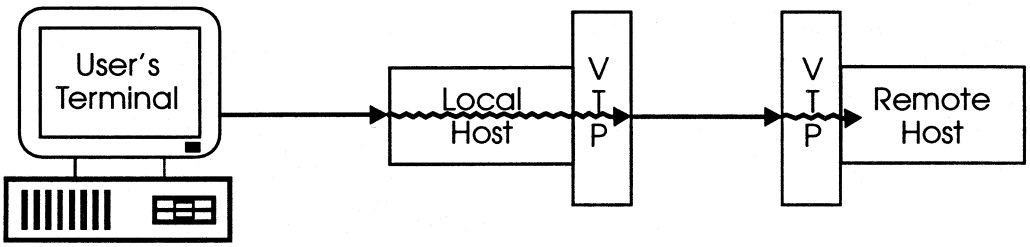


Figure 6.2

In practice there is a lot of work involved in making a virtual terminal protocol work with any operating system, so the simpler the better.

The Telnet protocol is inefficient in its use of bandwidth for the transfer of a single keystroke. It is not unusual to see installations which use remote echo (instead of local) which effectively doubles the load. Another consideration is the TCP and IP protocol overhead is some 40 octets to carry a single character which is fine for a LAN, but on an X.25 network or WAN environment, it could be very expensive.

Telnet Encapsulation

Telnet data is encapsulated within TCP segments and uses port 23. The Telnet servers, typically with the support of a master process, accept connections from clients on TCP port 23 (decimal). The client normally chooses a port number in a random manner. Once connected the master server spawns into a slave process which continues to serve the client's needs. In the normal character mode of operation each keystroke entered by the client is enveloped in its own TCP segment which is in turn encapsulated by an IP datagram. It is possible to put entire lines in a given segment when using the *linemode* option.

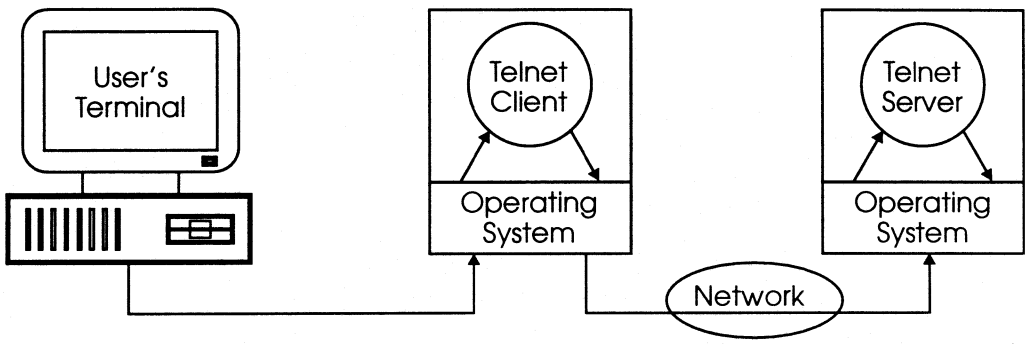


Figure 6.3

Notes:

Network Virtual Terminal (NVT)

To accommodate dissimilarity among machines a Network Virtual Terminal, (NVT), data format is used across the TCP connection between the client and server.

The client process maps user input from the client specific format into a generic format known as the NVT format. As the server receives the NVT formatted data it converts it into a form suitable for its local operating environment. This type of protocol allows for example Macintosh and DOS based machines to adjust to the CR/LF differences.

The NVT format specifies that all communications involve 8-bit bytes. At the beginning of a session all communications involve 7-bit ASCII characters, reserving bytes with the high order bit set, (bit 8), for control purposes.

NVT Control Characters

The NVT protocol interprets certain ASCII control characters as noted in the following table. Whilst ASCII code set (IA5) defines 128 different character combinations, only a small subset is used by NVT for control functions.

Character	Hex	Decimal	Function
NUL	00	0	Null, no operation: No effect on output
BEL	07	7	Sound audible alert
BS	08	8	Back space, move 1 character to left
HT	09	9	Horizontal tab, move right to next tab
LF	0A	10	Line feed, move down 1 line
VT	0B	11	Vertical tab, move down to next tab
FF	0C	12	Form feed, move down to start of page
CR	0D	13	Carriage return. move to left margin
Other	-	-	No effect: no operation performed

Figure 6.4

Passing Commands

To enable a client process to control the remote server process a number of commands are defined in the NVT protocol. To ensure that all possible combinations of control codes can be used without affecting the data channel, a special escape sequence to precede the control codes is used.

The **Interpret As Command** character, (IAC), is inserted into the data stream so that it precedes characters which have NVT defined control functions. The IAC character is a hex FF or a decimal 255, (all ones). If the IAC character is to be sent as part of the user data it is preceded by an additional IAC character. The first is stripped away and the second is allowed to pass through to the active process.

As some terminals can not generate NVT control codes directly, programmers typically bind a predefined key, for example a *Control C*, to an NVT control function. Normally a special key sequence is reserved at the client end to allow interruption of the Telnet process. Problems can occur if this key sequence is also used by the server process as the control sequence will be intercepted and processed locally rather than being passed to the server process.

Notes:

NVT Control Functions

The NVT control functions are noted in the following table.

CMD	Dec	Hex	Meaning
IAC	255	FF	Interpret As Command (Cmd follows)
GA	249	F9	Go Ahead (Line turn around signal, half duplex)
EL	248	F8	Erase Line (Delete last line)
EC	247	F7	Erase Character (Delete last byte)
AYT	246	F6	Are You There (Server reachability test)
AO	245	F5	Abort Output (Dump buffered data)
IP	244	F4	Interrupt Process (Abort active process)
SYNCH	-	-	Synchronise (Out of band data)
BRK	243	F3	Break (Break key)
DMARK	242	F2	Demarcation octet (For out of band data)
NOP	241	F1	No Operation
EOR	239	EF	End Of Record (terminates data records)

Figure 6.5

Of interest in these control functions the Synchronise data command, (SYNCH), is used to pass out of band control codes through the use of the TCP Urgent mechanism. When sending control codes the Telnet client issues a SYNCH command to its TCP entity along with a reserved octet value known as the DATA Mark, (DMARK). The TCP

Notes:

entity transmits the segment using the out of band Urgent flag. Urgent segments are not subjected to flow control and will force the receiving entity to discard all queued data until the DMARK is found. After encountering the DMARK data transfer proceeds as normal.

This out of band communications allows the client to regain control and command applications which may be in a loop and unable to process normal in band data.

Telnet Option Negotiation

The Telnet protocol permits symmetric option negotiation meaning session end can initiate the negotiation process.

The option negotiation process begins with one end sending either a DO X or WILL X command. In these commands the X variable encodes the desired option, i.e. a code of O indicates shift to 8-bit binary data representation. The remote end then accepts or declines the option by returning a WILL X or WON'T X command, (FD or FE hex respectively). By returning a WILL command a station indicates its acceptance of the requested option. A WON'T command indicates the option was refused.

Some options require further negotiation once they take effect. This process, known as Option Sub-negotiation, uses the start and end of option Sub-negotiation commands to synchronise the negotiations between the two stations.

Notes:

Telnet Option Commands

Noted below is a table of the various Telnet option commands. An option is not considered to be in effect until both sides have agreed to its use. To prevent loops from arising a station should send no response to a request to set options which are already in effect.

CMD	Dec	Hex	Meaning
DON'T	254	FE	Denial of requested option
DO	253	FD	Acceptance of requested option
WON'T	252	FC	Refusal to perform specified option
WILL	251	FB	Agreement to perform specified option
SB	250	FA	Start of option negotiation
SE	240	FO	End of option negotiation

Figure 6.6

If desired a station can request that an option be implemented at the remote end by sending the DO X command. The remote end either accepts or refuses this request by returning a WILL X or WON'T X command.

A station can indicate that it desires the use of a given option by sending the WILL X command. Once again the remote station can either accept or decline this assertion by returning a DO X or DON'T X command.

Notes:

Telnet Options

Some of the key Telnet options are noted in the following table.

Option	Code	Meaning
Transmit Binary	0	Shift to 8 bit binary mode
Echo	1	Causes one side to echo received data
Suppress-GA	3	Suppress Go Ahead signal (Full Duplex)
Status	5	Request option status
Timing-Mark	6	Request a timing ark in return stream for synchronisation
Terminal-Type	24	Exchange terminal type information
End-Of-Record	25	Terminates data sent with an EOR code
Line mode	34	Uses local editing. Send lines instead of characters

Figure 6.7

Notes:

File Transfer Protocol - FTP

File Transfer Protocol (FTP) as specified in RFC 959 is the classic use for most large networks - recent statistics for the Internet show that it is the most common application program in use today. Even though a connection oriented protocol like TCP is used for FTP, there are some issues of network load, security, bi-directional transfer and data representation that need to be addressed.

Network Load

File transfers can be small, big or very big indeed. The variability of the load which file transfer can impose on a network, creates lots of headaches for the network designers and administrators with no real solution to the problem.

Security

How to ensure that the host that wants to transfer a file is entitled to do so? Or worse (in these days of virii and worms) is the case where he wants to give you a file! Passing passwords across networks (in clear) is not a good idea and so ideally we need some form of local authentication scheme.

Bi-directional Transfer

Users need to be able to pass files in both directions (i.e. read from and send to remote systems). Apart from the security implications, this can cause other problems - the availability of disk space for example.

Data Representation

There is often a requirement to exchange data between machines which have a different data representation system. For example reading an EBCDIC file onto an ASCII host requires a character translation.

FTP Operation

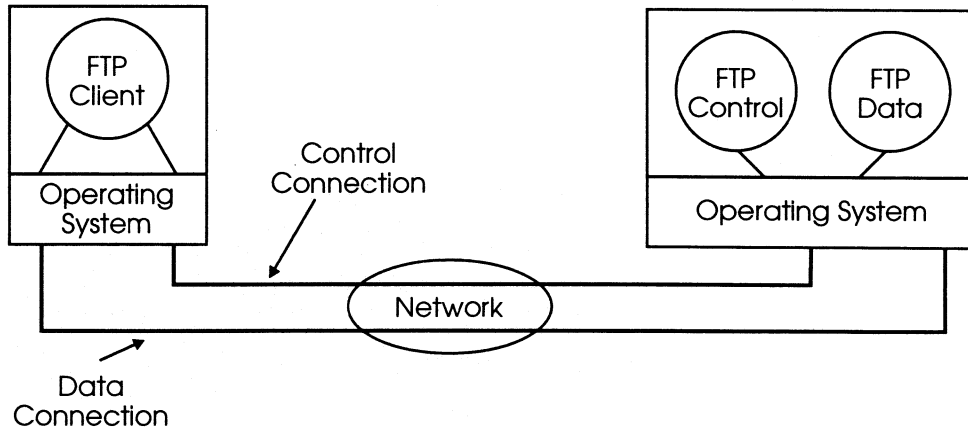


Figure 6.8

FTP provides facilities for *authorised* users to log into remote systems, list directories and copy files between local and remote systems. Some simple data conversion is supported (ASCII-EBCDIC), as well as byte reordering and the use of text record terminators (e.g. nl (UNIX) to crlf (PCDOS, VMS)). FTP may be used in an interactive mode by a human operator, as well as providing a program interface.

FTP uses Telnet to provide basic authorisation facilities, and in fact maintains a Telnet session between local (client) and remote (server) in order to exchange control information. The second TCP connection provides the data transfer path between client and server. Authorisation is dependent upon the operating system interface into Telnet - this usually requires a username/password to obtain system access, with restrictions upon file access and usage.

Notes:

A typical implementation has a server process at the remote site listening on the appropriate TCP port. When a client establishes contact, the server creates a new process to act as the session (control) server. The control server, communicating with the client over the control connection, demands a username/password combination from the client. Having successfully *logged-in*, the client can list directories, select options and specify files for transfer. Once a transfer is initiated, the control server opens a second TCP connection and creates a new process (the data server) to conduct the transfer. This process is destroyed, and the TCP connection closed on completion of the transfer - the client then converses with the control server for the next transfer.

FTP is a sophisticated protocol that permits third party transfers - i.e. a client can arrange a transfer between two other systems, without actually taking part in the data transfer.

Notes:

FTP Encapsulation

FTP data is encapsulated within TCP segments and uses ports 20 and 21 at the server using port 20, for the purpose of data exchange. FTP protocol uses two TP connections between the client and server, the control and data connection.

Control Connection

This is used to facilitate user control of the data transfer process. The control connection allow users to perform login and password exchange, obtain file listings, and specify which file should be transferred.

Data Transfer Connection

This is created whenever data needs to be transferred, as determined through the control connection. The control connection normally remains in effect until the FTP session is terminated. The data transfer connections are normally created and terminated with each file being transferred. When the control connection is broken, the user is considered logged out and all data transfer processes are terminated.

The format of the data exchanged across the control connection follows the Telnet NVT data structure. Unlike ordinary Telnet connections, however, the FTP control connections do not support option negotiation.

Notes:

FTP Port Assignments

As with other client server protocols the server normally awaits connection requests at a well known port. The client software typically assigns ports in a random manner, avoiding those (well known) port numbers which have been preassigned.

The client will normally start by choosing a local port value for the control connection, using it to establish a connection to the server's well known port 21 for the FTP control connection. The client's port value is specified in the TCP SYN segment allowing the server to respond to the client's connection request.

Once the control connection is established the client uses it to communicate the local port value which will be used for the data transfer portion of the connection. The server's control process will then create a separate data transfer process which will await a connection request, (passive open), from the specified client port number. The server's data connection port is always port number 20.

After informing the server of its intended local port value for the data transfer, the client will establish a connection to the server using the specified local port and the server's well known port number 20 as the connection end points. The server will only accept connections on port 20 which originate at the expected remote port assignment.

Notes:

FTP Connection

The following example shows the typical user perspective of the start of an FTP session. The user input is in bold while the standard type indicates responses from the FTP client or server. This particular process has been undertaken with the "verbose" option which allows the full status messages of the transfer to be provided.

%FTP ftp.srv (Name or address can be given)

Connected to 42.0.0.10

220 ftp.xyz.com FTP Server Ready

Name: **Anonymous**

331 Guest login ok, send ident as password

Password: **Guest**

230 Guest login ok, restrictions apply

ftp**Get IIT.EXT IITLOCAL.EXT**

200 Port command ok

150 Opening data connection, (42.0.0.11,1201) number of bytes = XXXXXXXX

226 Data transfer complete

XXXXXXXX bytes received in XXXXX seconds (XXKbps per second)

ftp**Close**

221 Goodbye

ftp**Quit**

FTP Commands

Command	Description
USER	User names - Supplies user name during login.
PASS	Password - Supplies user password.
CWD	Change Working Directory - Changes directory to the specified path.
CDUP	Change To Parent Directory - Changes working directory to the parent directory.
REIN	Reinitialise - All I/O buffers are flushed and resets all parameters to their default settings. The control is left open and a USER command is expected to follow.
QUIT	Logout - Causes the control connection to be broken and the user is logged out.
TYPE	Representation Type - Used to configure the structure of the data. Configured types are: A+ASCII, E=EBCDIC and I=Image.
MODE	Transfer Mode - Used to configure the transfer mode as follows: S=Stream (Default), B=Block, C=Compressed.
RET	Retrieve - Causes the server to send a copy of the file specified by the given path name to the client.
STOR	Store - Accept and store a file with the specified path and filename.
DELE	Delete - Causes the file specified by path name to be deleted at the server site.
RMD	Remove Directory - Causes the specified directory to be removed.
MKD	Make Directory - Causes a directory to be created as specified in the provided path name.
LIST	List - Causes the contents of the default of specified directory to be listed. Can be used to get information on a file.

Command	Description
STAT	Status - Used during a transfer to determine the status of the data connection. It can be used between transfers with a path name given the STAT command effectively has the same effect as the LIST command, (Listing sent over control connection).
HELP	Help - Used to obtain information regarding the various FTP commands by specifying the command in question as the argument.
NOOP	No Operation - Causes the server to return an "ok" reply.

Notes:

FTP Results Codes

When FTP commands are issued results are returned in both a 3-digit numeric format and in a human readable form. The numeric results codes are noted below for each of the digit alignment.

First Digit Result Codes

Result Code	Meaning
1YZ	Positive preliminary reply
2YZ	Positive completion reply
3YZ	Positive intermediate reply
4YZ	Transient negative reply
5YZ	Permanent negative reply

Second Digit Result Codes

Result Code	Meaning
XOZ	If there was a problem, this code indicates a syntax error or unknown command
X1Z	Reply to information request
X2Z	Reply referring to the control of data connections
X3Z	Reply referring to authentication
X4Z	Unspecified
X5Z	Replies indicating the status of the server file system

Third Digit Result Codes

The third digit of the result code is used to specify additional information based on the associated command. Its coding is command specific.

Trivial File Transfer Protocol - TFTP

TFTP is an unsophisticated file transfer protocol intended for simple applications - it is often used by diskless workstations to grab a copy of the system image, for example.

TFTP runs over UDP, so error handling has to be built into the TFTP application. A positive acknowledgement mechanism is used. The sender transmits the file using 512-byte blocks, and awaits an acknowledgement before sending the next block. If the sender's timer expires, the block is retransmitted. The receiver also uses timers, if a new block is not received within the specified time following an ACK, the ACK block is retransmitted. Hence the term symmetric.

TFTP is not necessarily more expensive (in network terms) than FTP, but it is much more restricting.

Notes:

Domain Name Service (DNS)

The *Domain Name Service (DNS)* was established to cope with the tremendous growth of the Internet. Originally the symbolic name of each connected host within the Internet was maintained centrally by the NIC (in a file called HOSTS.TXT). This file was regularly read by all hosts (using FTP), in order to update their internal table (e.g. */etc/hosts*) - with the current size of the Internet this would result in unacceptable loads, it is also difficult to keep this information up to date.

The solution to this problem was to decentralise responsibility for maintaining the information to the administrators of the individual networks that make up the Internet (these are the people who originate the changes). Each area of responsibility is referred to as an *Administrative Domain*, hence the origin of the term *Domain Name Service*. DNS represents a class of service generically known as a directory service, a second example is the *Network Information Service*.

Notes: DNS replaces a IP address with a word eg www.microsoft.com instead of its IP address.

Structure of the Name Space

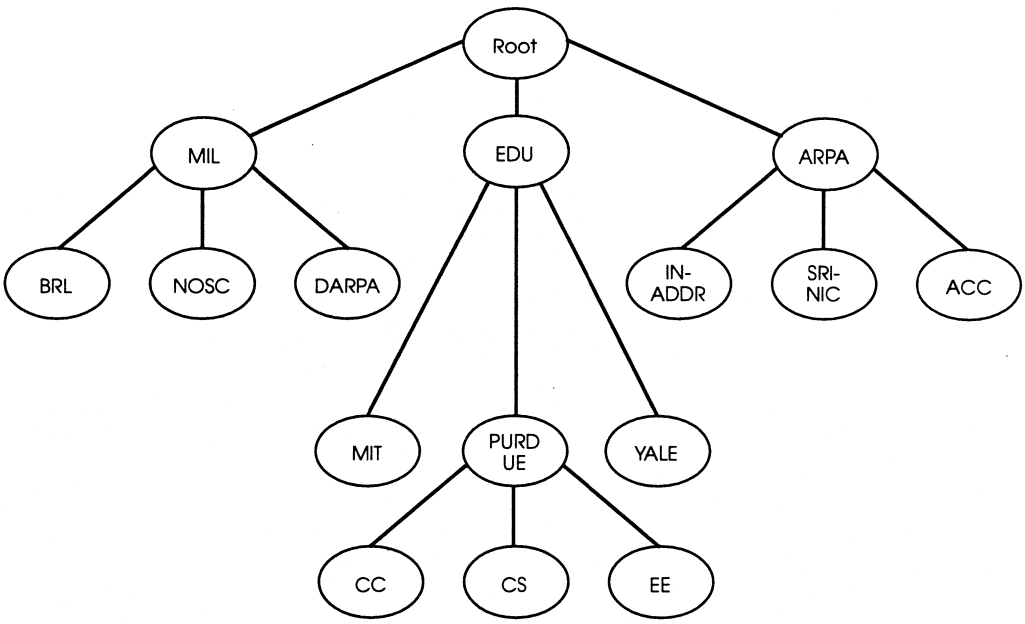


Figure 6.9

The diagram illustrates a portion of the Internet domain name space. It is arranged like a rooted tree: a root is established, below that are *top-level domains*; below that are *sub-domains*; below that are addressable *objects* - hosts, mailboxes etc. The names of objects are always unique within a domain.

A specific object is identified by starting at its name, followed by sub-domain and top-level domain, e.g. CS.PURDUE.EDU.

Notes:

The illustration indicates the top-level domains established in the USA:

- MIL for the US military
- EDU for universities and other educational establishments
- GOV for government organisations
- COM for commercial organisations
- ORG for non-commercial organisations
- ARPA for Internet-specific organisations

There is also a top level domain for each country, AU for us, DE for Germany etc. An example of a mailbox could be barney@iit.oz.au

Notes:

DNS Name Servers

To obtain information for host *barney.acc.arpa*, Fred contacts DNS1.

DNS1 contacts DNS2 to resolve request (if required) and returns this to Fred.

Each DNS name server manages a so called zone, which starts at a node and includes all underlying branches. The name servers all recognise neighbouring servers above and below in the hierarchy - there are usually two servers per zone to provide some redundancy.

Information about addressable objects is stored in *Resource Records (RRs)*, which contain the following data:

- Owner of the information
- Type of information
- Information class (e.g. IN - Internet, ISO - ISO)
- Period of validity
- Data

RR is stored in ASCII format and a number of *types* exist, including:

- Start of Authority (SOA) - Definition of a zone
- Address (A) - Internet address of a host
- Name Server (NS) - Name of a name server for a domain
- Host Information (HINFO) - Operating system, hardware etc.
- Canonical Name (CNAME) - Alias for a host
- Well-Known Services (WKS) - List of services supported (Telnet etc.)
- Mail Exchanger (MX) - Mail manager for the domain
- Gateway Pointer (PTR) - Gateway address

Resolver
- Runs in each client
- translates names to addresses by talking to DNS severs

Root Servers
- know about the top level of the domain name space

Authoritive servers
- know about a particular subset of the name space

Backup servers
- backup an authoritive server in case it is down

A query on a name server will result in either an appropriate answer or the name and address of another server that has the information requested.

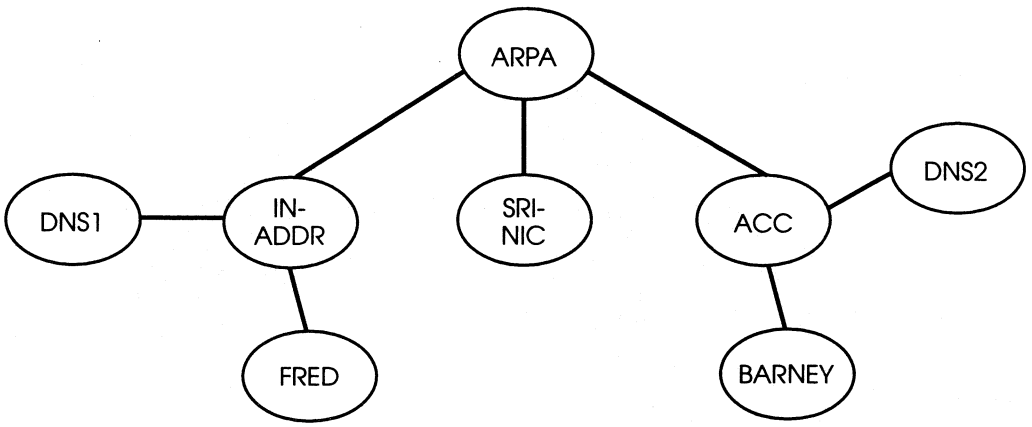


Figure 6.10

Each host has a *resolver* that handles name server queries on behalf of an application program. The resolver is required to store the information it obtains locally, in a cache, so that the information can be reused in similar queries and hence reduce load on the name servers.

Notes:

Types of translation
forward - Microsoft.com to an IP address
Reverse - 131.107.197.1 to a domain name
You may wish to stop DNS accessing root servers
- if you are not connected to the internet

Electronic Mail Transfer

The transfer of email between attached hosts is becoming an increasingly important role for many networks, as is the transfer of mail between networks.

TCP/IP provides a number of standards which define email related services. The prime email application, which is widely implemented with TCP/IP software, is SMTP - the Simple Mail Transfer Program, defined in RFC821. The companion standard, RFC822 defines two other important components of any email environment - the message format and the addressing method.

RFC822 Message Format

The RFC822 format uses only readable text - this has the advantage that everything is human-readable, and therefore easier to debug. The message header format consists of a series of keywords, followed by a colon, followed by the value.

RFC822 Addressing

The address format used is a two-part name:

local-part@domain-name

Domain name is the Internet domain name of the host on which the recipient is resident (e.g. iit.oz.au) with the local part being the recipient's login name.

The Internet provides an end-to-end service for mail transfer which establishes a connection with the remote system, transfers the message, the remote system acknowledges receipt, and the local copy of the mail is removed. This provides guaranteed delivery. Many systems are connected via *mail gateways* which provide a store and forward facility (and often are used to connect dissimilar mail systems) and often do not guarantee delivery.

Simple Mail Transfer Protocol - SMTP

SMTP is solely responsible for transferring mail between hosts, it does not specify how the mail is created, or presented to the user - this is implemented by the operating system.

Mail transfer is based on TCP. The client contacts the remote system (TCP port 25) and waits for the server to send a 220 READY FOR MAIL message. When this is received, the client and server exchange domain names and transfer then commences. The sender name is passed, followed by a list of recipients and the message is transferred. On completion of all transfers, the client may reverse the direction of transfer (using the TURN command) and the client and server swap roles and transfer mail in the opposite direction.

Error handling within the protocol is good - incorrectly addressed messages are trapped and returned to recipients for example. It is also possible that mail can be forwarded to new addresses automatically, with details of the new address being stored by the client for future use.

Each station needs:
- IP address
- Netmask
- Broadcast address
- Default router
- DNS server
- WINS server
- Boot file

page 167

Notes:

Network File System

The Network File System, developed by SUN Microsystems, is a means of sharing access to remote files. The basis of NFS is the Remote Procedure Call (RPC).

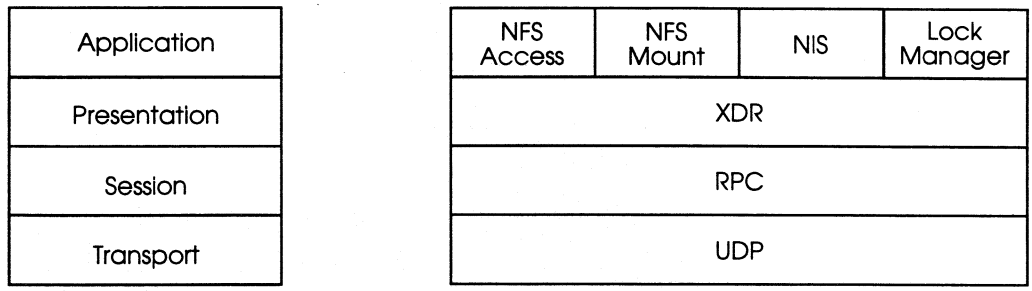


Figure 6.11

RPC provides remote communication from an application program, using the semantics of a function call. The arguments and return value of the *function call* are carried across the network in an architecture-independent format called XDR (eXternal Data Representation).

The services provided by RPC are low-level blocks designed for building into client/server architectures. RPC is available both to in-kernel routines (which is how NFS is usually implemented) and to application programs (which is how the Lock Manager is implemented).

Notes:

RPC is usually provided over UDP, though ISO versions are appearing.

NFS Mount and use remote file systems as if they were local (user-transparent). NFS is stateless, so recovery from server, client or network crashes is straightforward. NFS is widely used and available for many operating systems.

Lock Mgr File locking compatible with UNIX System V. This service is stateful.

NIS Network Information Service (formerly Yellow Pages, which is a registered trademark of British Telecom in Great Britain). A network administration service that replicates data across the network.

NFS	RFC-1094
RPC	RFC-1057
XDR	TRFC-1014

Notes:

Network Information Service

NIS is a directory service that was developed by Sun Microsystems as a part of the Network File System (NFS), and it is normally distributed with it. NIS used to be called *Yellow Pages (YP)*, but the name was a registered trade mark and had to be changed.

The primary purpose of NIS is to distribute a centrally managed copy of the file */etc/passwd* to each host. This ensures that each host uses the same user name, UID and GID when creating and accessing files (obviously a requirement if files are centrally located on a NFS file server). The hosts are grouped in *domains* - these are not the same (necessarily) as DNS domains. Each domain is managed by a server that passes copies of files (referred to as *maps*) to each host as it starts up, or broadcast them if they are modified.

Notes:

FTP 44.0.0.99
enter userid
enter password
CD PEACS
PWD (display current directory)
GET RFC854.TXT
QUIT

TFTP GET local-file 44.0.0.99 remote-file-full-path
Trivial file transfer protocol

NIS also handles a number of other */etc* files:

/etc/passwd
/etc/group
/etc/hosts
/etc/ethers
/etc/networks
/etc/rpc
/etc/services
/etc/protocols
/etc/netgroup
/usr/lib/aliases

NIS is very widely used for managing NFS-based networks but its use is relatively controversial and it does create security problems. There may be conflicts with DNS when both systems are used on the same host.

Notes:

Bootstrap Protocol

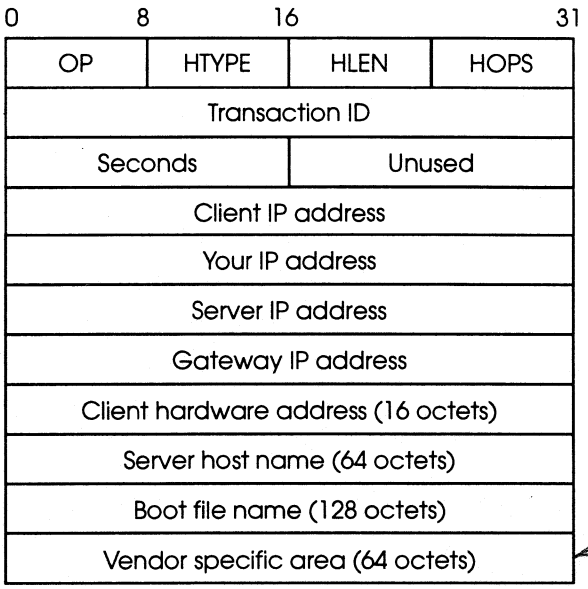


Figure 6.12

RARP has a number of shortcomings - it cannot be used on networks that dynamically assign addresses; it requires access to hardware at a low level that may not be available to the programmer on some machines; it is very wasteful as it only returns the 4 byte IP address when much more data could be held in the packet.

BOOTP overcomes many of these problems. It is based on UDP/IP and hence does not require access to low level hardware drivers. A single BOOTP message contains a great deal of information, including the client and server IP addresses, the address of any gateway between

Notes:

client and server and details of the bootable image file. There is also a vendor specific area that allows vendors to send information specific to their machines. Use of these fields in a standard way has been described in an RFC and this allows much other information of use to a diskless workstation to be disseminated. (A list of Gateways, the Domain Name Service, Time Service etc.).

BOOTP relies on the use of the IP limited broadcast facility (setting address = 255.255.255.255) that allows IP to broadcast without knowing its own IP address, or the address of the local network. Note that the reply from the server also has to be sent using this mechanism (although the server knows the client's IP address) as the client IP software can only respond to a broadcast message without a knowledge of its own address.

BOOTP uses ARP and it therefore has to provide an error recovery capability. This is supported using a timeout/retransmission mechanism. To avoid the possibility of overloading a server following a power failure (or similar event), the timeout interval is a random interval and should increase with successive retries.

Booting using BOOTP is a two-step procedure. BOOTP does not provide a memory image, it only provides the information needed to find one - the image is downloaded using a second protocol (usually TFTP).

Notes:

7

Interrelationship of TCP/IP with Unix

Interrelationship of TCP/IP with Unix

Unix International

An independent, international, non-profit industry association funded by its members to promote and provide future direction of the UNIX system V operating system. It has over 130 members and works closely with X/Open.

For more information, contact:
UNIX International
20 Waterview Blvd
PARSIPPANY
New Jersey 07054
(201) 263-8400

Multics Project — A joint venture between General Electric, AT&T Bell Laboratories, and MIT to create an operating system on a GE 645 computer.

Multics — **Mult**iplexed **I**nformation and **C**omputing **S**ystem

Unics — **Uni**plexed **I**nformation and **C**omputing **S**ystem

The name was later changed to UNIX. The PDP-7 version of UNIX was later enhanced to allow two users to work at the same time.

UNIX — A multiuser, multitasking operating system from AT&T that runs on a wide variety of computer systems from micro to mainframe. UNIX is written in C (also developed at AT&T), which is a language designed for system-level programming. It is C's inherent portability that allows UNIX to run on so many different computers.

UNIX is made up of the *kernel*, the heart of the operating system, the file system, a hierarchical directory method for organising files on the disk and on the shell, the user interface which provides the way the user commands the system.

Normal UNIX commands are very cryptic but they can be replaced with shells that are easier to use, including Graphical User Interfaces (GUIs), such as X-Windows, Open Look and OSF/Motif.

The following list shows typical UNIX commands with their DOS counterparts:

Command	UNIX	DOS
List Directory	ls	dir
Copy a file	cp	copy
Delete a file	rm	del
Rename a file	mv	rename
Display contents	cat	type
Print a file	lpr	print
Check disk space	df	chkdsk

Notes:

The History of UNIX

UNIX was developed in 1969 by Ken Thompson for the PDP-7. Additional work was done by Dennis Ritchie, and by 1974, UNIX had matured into a state-of-the-art operating system primarily running on PDP computers. UNIX became very popular in scientific and academic environments.

Considerable enhancements were made to UNIX at the University of California at Berkeley, and versions of UNIX include the Berkeley extentions, which became widely used on Digital's VAX systems. By the late 1970s, commercial versions of UNIX, such as IS/1 and XENIX, became available.

In the early 1980s, AT&T began to consolidate the many versions of UNIX into standards which evolved into System III and eventually System V. Before divestiture (1984), AT&T licenced UNIX to universities and other organisations, but was prohibited from outright marketing of the product. After divestiture, it began to market UNIX aggressively.

In January 1989, the UNIX Software Operation was formed as a separate division devoted exclusively to the product. In November 1989, it introduced the most significant release of UNIX: System V Release 4.0, which incorporates XENIX, Sun OS, Berkeley 4.3BSD and System V into one standard. AT&T's SVID (System V Interface Definition) specifies the requirements for UNIX compatibility.

In June 1990, UNIX Software Operation was turned into UNIX System Laboratories, Inc, a subsidiary of AT&T.

Notes:

The name UNIX was coined for a single-user (un) version of MULTICS, as it was intended to be a scaled-down version of that very elaborate operating system.

Ironically, today UNIX's multiuser capabilities are one of its most important features.

To encourage university researchers to adopt and use the new TCP/IP protocols, DARPA made an implementation available at low cost. At that time most university computer science departments were running a version of the UNIX operating system, available in the University of California's *Berkeley Software Distribution*, commonly called Berkeley UNIX or BSD UNIX.

By funding Bolt Beranek and Newman, Inc (BBN) to implement its TCP/IP protocols for use with UNIX, and funding Berkeley to integrate the protocols with its software distribution, DARPA was able to reach over 90% of the university computer science departments. The new protocol software came at a particularly significant time because many departments were just acquiring second or third computers and connecting them together with local area networks. The departments needed communication protocols and no others were generally available.

The Berkeley Software Distribution became popular because it offered more than basic TCP/IP protocols. In addition to standard TCP/IP application programs, Berkeley offered a set of utilities for network services that resembled the UNIX services used on a single machine. The chief advantage of the Berkeley utilities lay in their similarity to standard UNIX.

Notes:

Beside a set of utility programs, Berkeley UNIX provides a new operating system abstraction known as a *socket* that allows application programs to access communication protocols. A generalisation of the UNIX mechanism for I/O, the socket has options for several types of network protocols in addition to TCP/IP. Independent of its overall merits, however, the introduction of the socket abstraction was important because it allowed programmers to use TCP/IP protocols with little effort. Thus, it encouraged programmers to experiment with TCP/IP.

3 BSD	(Berkeley Standard Distribution)
BSD 4.1	(1981)
BSD 4.2	(1983)
BSD 4.3	(1986)

We think of a socket as a generalisation of the UNIX file access mechanism that provides an end point for communication. As with file access, application programs request the operating system to create a socket when one is needed. The system returns an integer that the application program uses to reference the newly created socket.

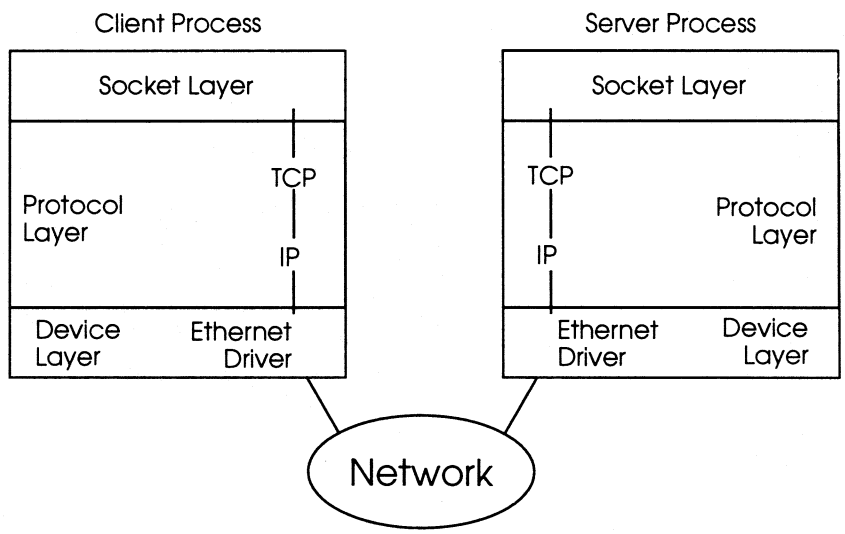


Figure 7.1

The chief difference between file descriptors and socket descriptors is that the operating system binds a file descriptor to a specific file or device when the application calls *open*, but it can create sockets without binding them to specific destination addresses. The application can choose to supply a destination address each time it

uses a socket (e.g. when sending datagrams), or it can choose to bind the destination address to the socket and avoid specifying the destination repeatedly (e.g. when making a TCP connection).

Whenever it makes sense, sockets perform exactly like UNIX files or devices, so they can be used with traditional operations like *read* and *write*. For example, once an application program creates a socket and creates a TCP connection from the socket to a foreign destination, the program can use *write* to send a stream of data across the connection (the application program at the other end can use *read* and receive it). To make it possible to use primitives like *read* and *write* with both files and sockets, the operating system allocates socket descriptors and file descriptors from the same set of integers and makes sure that if a given integer has been allocated as a file descriptor, it will not also be allocated as a socket descriptor.

The *socket* system call creates sockets on demand. It takes three integer arguments and returns an integer result.

af specifies the protocol family to be used with the socket. It specifies how to interpret addresses when they are supplied. Current families include the following:

(AF_INET)	TCP/IP internet
(AF_PUP)	Xerox Corporation PUP internet
(AF_APPLETALK)	Apple Computer Incorporated Appletalk network
(AF_UNIX)	UNIX file system

as well as many others.

Notes:

type specifies the type of communication desired. Possible types include:

(SOCK_STREAM) Reliable stream delivery service

(SOCK_DGRAM) Connectionless datagram delivery service

(SOCK_RAW) A raw type that allows privileged programs to access low-level protocols or network interfaces

Two additional types have been planned but are not implemented.

To accommodate multiple protocols within a family, the socket call has a third argument that can be used to select a specific protocol. To use the third argument, the programmer must understand the protocol family well enough to know the type of service each protocol supplies.

Once a socket has been created, a server uses the *bind* system call to establish a local address for it. Argument socket is the integer descriptor of the socket to be bound. Argument *localaddr* is a structure that specifies the local address to which the socket should be bound, and *addrlen* is an integer that specifies the length of the address measured in bytes.

Instead of giving the address merely as a sequence of bytes, the designers chose to use a structure for addresses.

Initially, a socket is created in the *unconnected state*, which means that the socket is not associated with any foreign destination. The system call *connect* binds a permanent destination to a socket, placing it in the *connected state*. An application program must call *connect* to establish a connection before it can transfer data through a reliable stream socket.

Notes: WINSOCK.DLL

Sockets used with connectionless datagram services need not be connected before they are used, but doing so makes it possible to transfer data without specifying the destination each time.

Argument *socket* is the integer descriptor of the socket to connect. Argument *destaddr* is a socket address structure that specifies the destination address to which the socket should be bound. Argument *addrlen* specifies the length of the destination address measured in bytes.

The semantics of *connect* depend on the underlying protocols. Selecting the reliable stream delivery service in the AF_INET family means choosing TCP. In such cases, *connect* builds a TCP connection with the destination and returns an error if it cannot. In the case of connectionless service, *connect* does nothing more than store the address locally.

When an application program has established a socket, it can use the socket to transmit data. There are five possible operating system calls from which to choose: *send, sento, sendmsg, write,* and *writev. Send, write* and *writev* only work with connected sockets because they do not allow the caller to specify a destination address.

Argument *res* contains an integer socket descriptor (*write* can also be used with other types of descriptors). Argument *buffer* contains the address of the data to be sent, and argument *length* specifies the number of bytes to send.

Analogous to the five different output operations, 4BSD UNIX offers five system calls that a process can use to receive data through a socket: *read, readv, recv, recvfrom,* and *recvmsg*. The conventional UNIX

Notes:

input operation, *read*, can only be used when the socket is connected. Where *descriptor* gives the integer descriptor of the socket or file descriptor from which to read data, *buffer* specifies the address in memory at which to store the data, and *length* specifies the maximum number of bytes to read.

A summary of the main system calls for both clients and servers:

Phase	Active		**Passive**
Open end point		socket	
Designate end point		*bind*	
Connection set-up (TCP)	connect		*listen*
			accept
Send data		*write, send*	
		sendto, sendmsg	
Receive data		*read, recv*	
		recvfrom, recvmsg	
Close connection (TCP)		*shutdown*	
Shutdown end point		*close*	
Miscellaneous		*getpeername, getsockname*	
		getsockopt, setsockopt	
Accept input from multiple sources		*select*	

The use of sockets considerably simplifies the programming of client-server relationships - it forms the basis of NFS, X-Windows and DCE.

Notes:

Configuration Files

/etc/hosts

```
# Local network host addresses
127.1           local localhost
192.9.150.202   oxford
192.9.150.201   cambridge
192.9.150.203   dundee
```

/etc/networks

```
# Network name database
loopback-net  127      s/w-loop
inset         192.9.150
fnash         192.9.121
```

/etc/protocols

```
# Internet (IP) protocols
ip  0  IP       #internet protocol
icmp 1 ICMP     #internet control mes
ggp 3 GGP       #gateway-gateway pr
tcp 6 TCP       #transport control pr
udp 17 UDP      #user datagram prot
```

/etc/services

```
# Network services, internet style
ftp             21/tcp
telnet          23/tcp
smtp            25/tcp    mail
tftp            69/udp
login           513/tcp
```

Unix configuration files are normally stored in the directory */etc*, along with the other system administration files and tools. Configuration is contained in four files:

/etc/hosts

The purpose of this file is to provide a symbolic name for an Internet address, as users are unlikely to remember a cryptic addres such as 192.9.150.202

Entries in */etc/hosts* contain an IP address and an official name. Optionally, several aliases may follow the official name.

Notes: HOSTS
NETWORK
PROTOCOLS
SERVICES
RESOLVE

/etc/networks

This file contains network names and their numbers and aliases. An Internet connected host would have a large table, most sites have at least their own network name to improve the readability of reports generated by the diagnostic software.

/etc/protocols

This file lists the protocol numbers for the various transport protocols. This is the number found in the *proto* field of the IP datagram header. This file should not normally be modified.

/etc/services

This file contains the port number and designations of the network services provided for TCP and UDP. The provision of extra services locally would require the modification of this file, which is normally read at start-up by application programs.

You should note that if your network is using *Yellow Pages* or *Domain Name Service* the data held in these files is replaced or supplemented by information in the directory services. The interrelation of these sources is a possible problem area associated with incorrect addressing of hosts on the network.

Notes: Port 0-1023 reserved (privileged)

The Internet Daemon

The Internet Daemon, *inetd*, is used as a *super server* to intercept service requests.

On receipt of a request an appropriate server is started to deal with it.

Configuration is controlled via */etc/inetd.cf*.

Early Unix implementations of TCP/IP launched a server process for each service at system startup - this could involve up to 15 processes, many of which would never be used. 4.3BSD introduced the concept of a *super server*, called *inetd*, which negotiates with hosts requesting services via the network. Once a connection has been established *inetd* starts a client-specific *connection* and the appropriate server process is started (details of the connections are delivered in file descriptor 0).

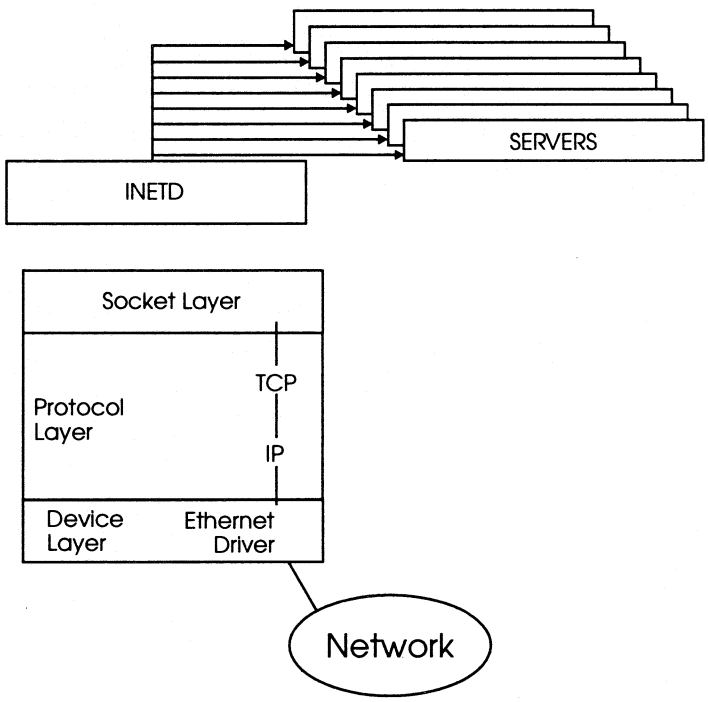


Figure 7.2

Inetd is configured via the file */etc/inketd.cf.* An example shows the format of the file:

```
#
# Internet server configuration database
#
```

ftp	stream	tcp	nowait	root	/etc/ftpd	ftpd
telnet	stream	tcp	nowait	root	/etc/telnetd	telnetd
shell	stream	tcp	nowait	root	/etc/rshd	rshd
login	stream	tcp	nowait	root	/etc/rlogind	rlogind
exec	stream	tcp	nowait	root	/etc/rexecd	rexecd
tftp	dgram	udp	wait	bin	/etc/tftpd	tftpd
ntalk	dgram	udp	wait	root	/etc/ntalkd	ntalkd
echo	stream	tcp	nowait	root	internal	
ime	stream	tcp	nowait	root	internal	

Service name - as defined in */etc/services*.

Socket type - type of communication mechanism (*stream, dgram* or *raw*).

Protocol - TCP or UDP as given in */etc/protocols*.

Wait/nowait - for datagram servers only, indicates whether *inetd* can accept further connections for the same port, or whether it should wait for the server to finish.

User name - the login ID under which the server should be started.

Server program name - path name of the server program.

Parameters - to be passed to server at startup (including program name - parameter 0).

Some test and information services such as *echo, discard, chargen, daytime* and *time* are internally processed by *inetd*. Note the above describes the 4.3BSD implementation. System V does things in broadly the same way.

Network Applications

The BSD implementation of TCP/IP also includes a number of operating system extensions which integrate network facilities into the operating environment. Along with sockets, these are widely implemented in all UNIXs.

- rlogin - remote login facility
- remsh - remote execution of a command
- rcp - file copy
- rexec - remote execution of a command

In addition to the standard TCP/IP applications - FTP, TFTP and Telnet - that are found in virtually all implementations, UNIX has a number of specific network applications first introduced in 4.3BSD, but since widely adopted by all versions. These applications are collectively referred to as the *r-utilities*, because they all begin with the letter *r*, as in *remote*.

There are a number of these applications, the main ones are described below. In addition most systems implement *rwho* and *ruptime*, which provide information about other network users and hosts.

rlogin

rlogin (or *remote login*) provides a virtual terminal service in a similar manner to TELNET. Connection is made to TCP port 513, passing a login string containing client and server login IDs and the terminal type. All characters entered at the keyboard are transparently sent to the server, where they are passed to a pseudo terminal device. All returned characters are displayed on the user's local terminal.

Notes:

remsh

remsh (or *remote shell*) allows users to execute commands on a different computer; connection is made via TCP port 514 passing the command line to the remote server. The server connects the *stdin, stdout* and *stderr* channels to the local process via a pair of TCP connections. Quoted redirection commands are interpreted remotely. It is possible to build some very complex and powerful commands using *remsh.*

rcp

rcp (or *remote copy*) extends the UNIX *cp* command into the network domain. It offers a number of enhancements to FTP, including the recursive copying of directories and third party transfers.

rexec

rexec (or *remote execution*) provides an alternative to *rsh* that incorporates the transmission of an encrypted password with the initial request. The service is often only available to programmers, via the *rexec()* system call, although some systems do provide an *rexec* program. The extra work involved in passwork authentication makes *rexec* less popular than *rsh.*

Notes:

Transport Layer Interface

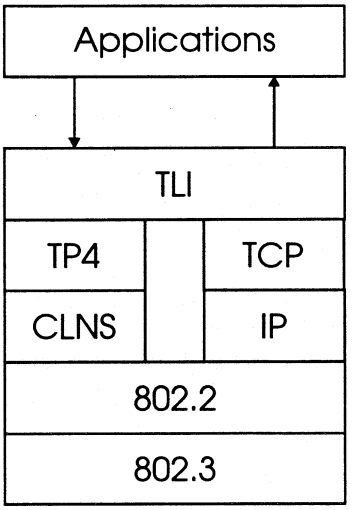


Figure 7.3

System V.3 introduced *Transport Layer Interface (TLI)* to provide a standard (based on ISO-OSI standards) mechanism for accessing any transport protocol.

Implemented for TCP/IP and ISO TP4 - programmers can produce network independent applications.

X/Open introduced improved version - *X/Open Transport Interface (XTI).*

AT&T introduced the concept of a *Transport Layer Interface (TLI)* System V release 3.0, along with *streams*. The objective was to provide a transport-independent API to facilitate eventual migration to OSI

Notes:

protocols. The idea is that programmers should programme to the TLI and therefore the application will work over any supported transport system. TLI is based on existing standards, specifically IS 8072-1986, which defines the ISO *transport service definition* - the interface to the transport layer - and it implements the OSI transport service definitions.

X/Open produced a refined version of TLI (called the *X/Open Transport Interface (XTI)* that makes small adjustments to the TLI interface. XTI was introduced along with XPG3 and it is likely that it will replace the socket interface as the industry standard method of connecting to a transport service.

TLI/XTI utilise a *lowest common denominator* approach in that some TCP and TP4 functions are not adopted, only those that are common to both protocols. TLI/XTI is more complicated to use for the application programmer than sockets.

Notes:

Future Developments

Widespread use of networked workstations has meant that UNIX acquires more and better network support. The next major development is *Distributed Computing Environment* (DCE) from the Open Software Foundation.

- Fully distributed environment.
- Protocol independent but will rely heavily on TCP/IP.
- Provides a range of services to enable major computer installations to be built across a network.

The socket interface is being replaced by a new API called *Transport Layer Interface* (TLI) which provides mapping into TCP and/or ISO OSI Transport Protocol 4 (TP4). New applications developed using TLI will be able to switch to an OSI stack when this becomes available.

- New applications should use TLI instead of sockets.
- Sockets will continue to be supported for some years.

Notes:

8

Network Management and SNMP

iit TRAINING

Network Management and SNMP

As well as protocols that provide network level services and application programs that use those services, an internet requires software that allows managers to debug problems, control routing and locate computers that violate protocol standards.

In a TCP/IP internet, IP gateways form the active switches that managers need to examine and control. Because these gateways connect heterogeneous networks, protocols for internet management operate at the application level and use TCP/IP for message transport.

Notes:

Network Management Model

Network management efforts identify five elements - security, fault analysis, accounting, performance evaluation and configuration.

This model is now widely accepted by many vendors - main players are DEC, HP, AT&T and IBM.

TCP/IP vendors recognise the model also.

The five broad areas of concern are explained below.

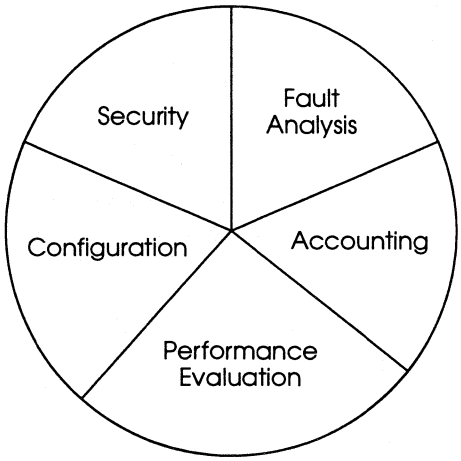


Figure 8.1

Security

Security of distributed computer networks is a topic that is becoming of greater importance - the problems that can arise from the free exchange of data between systems is clearly illustrated by the Internet worm.

Fault Analysis and Resolution

Many people see this as the most important part of any network management environment - this may not be the case but it is certainly the most visible.

Notes:

Performance Evaluation

Performance levels are results of the planning and investment that went into building the network.

Performance management involves making the best of available resources.

Configuration

Configuration is setting, collecting and storing selected information about everything in the network. It is a basic aspect of network management because all other functions such as performance monitoring, fault analysis, security and accounting rely on configuration information for each managed object.

Accounting

Accounting is a critical area for large organisations with extensive network infrastructure - somebody has to pay for it and therefore there has to be a mechanism for measuring usage and apportioning cost.

This structure has been largely embraced by DEC (Enterprise Management Architecture) and HP (OpenView); IBM's NetView and AT&T's Accumaster are both to support OSI structures in the future.

Notes:

The Internet Approach

The Internet has grown at an astonishing rate during the past decade. Apart from the more obvious problems that such growth brings (pressure on capacity, proliferation of networks and gateways), network reliability has become more of an issue. The core of the Internet is fairly well managed, being under the direct control of the INOC, but there is little management of autonomous systems - this inevitably leads to poor reliability.

The solution proposed by the IAB was to develop a full set of network management standards that could be used by vendors to produce manageable products rapidly. An R&D framework for this was established in 1988 - progress in this area has been very rapid.

When the IAB recommended that a set of Internet Network Management Standards be developed, a two-prong strategy for network management of TCP/IP-based internets was undertaken. In the short-term, the Simple Network Management Protocol (SNMP) was to be used to manage nodes in the Internet community. In the long-term, the use of the OSI network management framework was to be examined. Two documents were produced to define the management information: RFC 1065, which defined the Structure of Management Information (SMI), and RFC 1066, which defined the Management Information Base (MIB). Both of these documents were designed to be compatible with both the SNMP and the OSI network management framework.

Notes:

This strategy was quite successful in the short-term: Internet-based network management technology was fielded by both the research and commercial communities within a few months. As a result of this, portions of the Internet community became network manageable in a timely fashion.

The requirements of the SNMP and the OSI network management frameworks were more different than anticipated. As such, the requirement for compatibility between the SMI/MIB and both frameworks was suspended. This action permitted the operational network management framework, the SNMP, to respond to new operational needs in the Internet community by producing documents defining new MIB items.

Notes:

SNMP Architecture

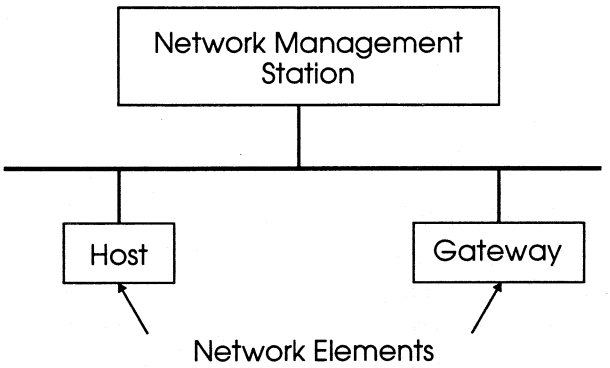


Figure 8.2

Implicit in the SNMP architectural model is a collection of Network Management Stations (NMS) and network agents. Network management stations execute management applications which monitor and control network elements.

Network elements are devices such as hosts, gateways, terminal servers and the like, which have management agents responsible for performing the network management functions requested by the network management stations. The Simple Network Management Protocol (SNMP) is used to communicate management information between the network management stations and the agents in the network elements. It allows the NMS to poll the agents at regular intervals for management data; receive alerts from agents following some form of network event; and the reconfiguration of network elements.

Notes:

The most obvious problem with SNMP (apart from the fact that it is very strongly tied to TCP/IP) is that there is no authentication mechanism. Potentially, anybody on the network may issue commands to start up or shut down devices. This situation is being urgently addressed, but in the meantime some vendors have decided not to implement the *SET* command, that enables remote configuration. This inevitably detracts from the usefulness of SNMP as a network management environment.

It is a reasonably straightforward task for a vendor to produce an SNMP agent and incorporate it into an existing product, replacing any proprietary management system. There are a few vendors who have yet to tackle the more complex requirements of the NMS. Many of the vendors are offering proprietary extensions to the MIB - most of these have been published in order to attract third party support. It is important to ensure that an NMS is able to support these extensions - otherwise it is of little use.

Notes:

Current Status

The current state of network management is defined by three RFCs:

- *Structure of Management Information (SMI)* RFC 1155 describes how managed objects contained in the MIB are defined.
- *Management Information Base (MIB)* RFC 1156 describes the managed objects contained in the MIB.
- *Simple Network Management Protocol (SNMP)* RFC 1157 defines the protocol used to manage these objects.

The *Common Management Information Services and Protocol over TCP/IP (CMOT)* may be used in addition to SNMP.

Current Status

The current network management framework for TCP/IP-based internets consists of: Structure and Identification of Management Information for TCP/IP-based Internets, which describes how managed objects contained in the MIB are defined as set forth in RFC 1155; Management Information Base for Network Management of TCP/IP-based Internets, which describes the managed objects contained in the MIB as set forth in RFC 1156; and, the Simple Network Management Protocol, which defines the protocol used to manage these objects.

Management Information Base

The MIB describes the attributes of each network device, what it does, how it does it, what instructions it can receive and what information it can provide. The MIB is extensible by vendors - i.e. a vendor can define a new entry that describes their particular piece of equipment.

Structure of Management Information

The SMI describes a set of rules that are used for interrogating the MIB - they define how the data is laid out. SMI is based on OSI Abstract Syntax Notation (ASN.1) syntax rules - hence it could be extended into the OSI - CMIP framework. SMI divides MIB objects into two categories - public and private. The public entries are effectively proprietary extensions made by individual vendors.

SNMP has now grown beyond its original objectives of a *stop-gap* protocol for use pending the arrival of OSI-based network management systems. It has been widely adopted by in excess of 50 vendors, including HP and IBM, as well as many important

networking vendors such as Retix and Cisco. This inevitably means that it will become entrenched, and hence will delay the introduction of universal OSI-based network management solutions further.

Below are examples of a few MIB variables along with their categories.

MIB Variable	Category	Meaning
sysUp Time	system	Time since last reboot
ifNumber	interfaces	Number of network interfaces
ifMtu	interfaces	MTU for a particular interface
ipDefaultTTL	ip	Value IP uses in time-to-live field
ipInReceives	ip	Number of datagrams received
ipForwDatagrams	ip	Number of datagrams forwarded
ipOutNoRoutes	ip	Number of routing failures
ipReasmOKs	ip	Number of datagrams reassembled
ipFragOKs	ip	Number of datagrams fragmented
ipRoutingTable	ip	IP Routing table
icmpInEchos	icmp	Number of ICMP Echo Requests received
tcpRtoMin	tcp	Minimum retransmission time TCP allows
tcpMaxConn	tcp	Maximum TCP connections allowed
tcpInSegs	tcp	Number of segments TCP has received
udpInDatagrams	udp	Number of UDP datagrams received
egpInMsgs	egp	Number of EGP messages received

Values for most of the items listed above can be stored in a single integer. However, the MIB also defines more complex structures. For example, the MIB variable *ipRouting Table* refers to a gateway's routing table. Additional MIB variables define the contents of a routing table entry and allow the network management protocols to reference the data for individual entries.

PDU

SNMP applies a simple approach to network management by using *fetch-store* operations rather than maintaining a large set of commands.

Command	Meaning
get-request	Fetch a value from a specific variable
get-next-request	Fetch a value without knowing its exact name
get-response	Reply to a fetch operation
set-request	Store a value in a specific variable
trap	Reply triggered by an event

An SNMP message consists of 3 main parts:

a. Protocol version
b. SNMP community identifier
c. Data area

The data area is divided into protocol data units (PDUs) - the commands listed above are examples of PDUs.

Notes:

Network Management Tools

RFC 1147 contains a comprehensive list of tools and many other products that may be used with TCP/IP-based networks. Many of these tools are free (just contact the appropriate university or research organisation), although the RFC also lists all known commercial products, ranging from NMS software to system configuration tools.

Notes:

The tools are broken into a number of categories:

Alarm A reporting/logging tool that can trigger on specific events within a network.

Analyser A traffic monitor that reconstructs and interprets protocol messages that span several packets.

Benchmark A tool used to evaluate the performance of network components.

Control A tool that can change the state or status of a remote network resource.

Debugger A tool that by generating arbitrary packets and monitoring traffic, can drive a remote network component to various states and record its responses.

Generator A traffic generation tool.

Manager A distributed network management system or system component.

Map A tool that can discover and report a system's topology or configuration.

Reference A tool for documenting MIB structure or system configuration.

Routing A packet route discovery tool.

Security A tool for analysing or reducing threats to security.

Status A tool that remotely tracks the status of network components.

Traffic A tool that monitors packet flow.

9

TCP/IP and the Future

TCP/IP and the Future

OSI and TCP/IP have same basic objective

- Transparent multivendor internetworking

TCP/IP is mature, stable and widely implemented. OSI is immature, expanding and not widely or fully implemented.

Both protocols have similarities of approach

- Layered, peer-to-peer architecture
- Global perspective for naming and addressing

There are also some differences of approach

- Service types
- Reliability

OSI and TCP/IP are both intended to achieve the same objective - provision of a uniform network service independent of vendor. Inevitably this means that there are many similarities between the two systems. There are also some marked differences of emphasis.

What is most important today is that TCP/IP represents a mature, functioning and widely implemented Internet standard. OSI represents a technically rich but incomplete Internet standard that has few implementations and is far from mature.

A major issue with OSI has been the differences between the US approach to networks (Connectionless) and the European approach, influenced by the services provided by the government controlled PTTs through the ITU-T (Connection Orientated X.25). With the introduction of Connectionless OSI which is remarkably similar to IP, it is likely that the US view will come to dominate the OSI world.

There are other differences of philosophy between OSI and TCP/IP, mostly related to the provision of services. OSI emphasises a strictly layered approach to the provision of services, reliability of transfer is provided at multiple levels. TCP/IP views reliability as an end-to-end problem and focuses all the error control into the transport layer.

OSI Competitors

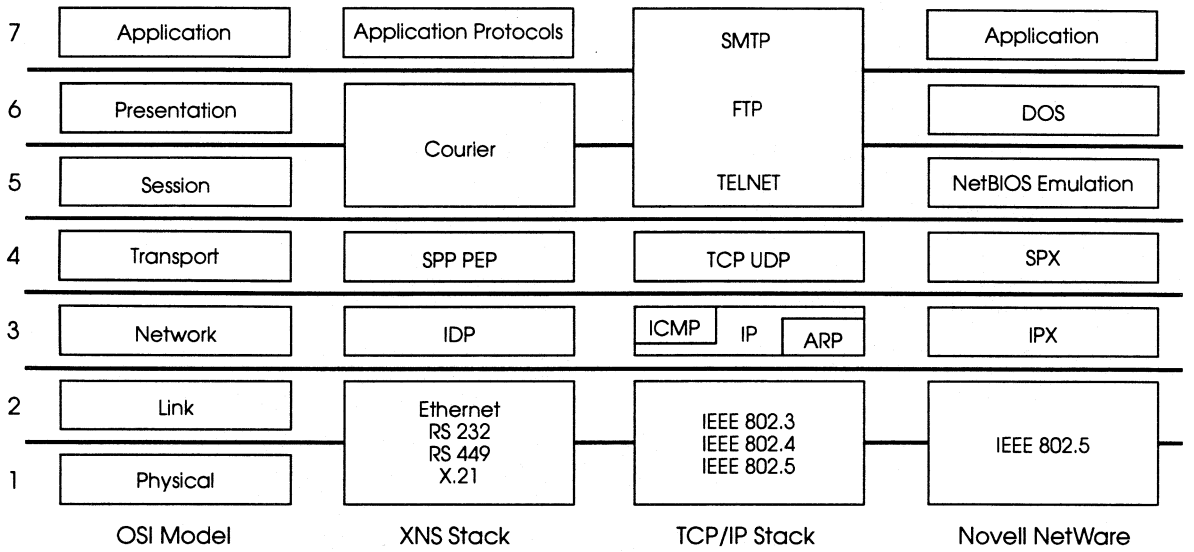


Figure 9.1

Notes:

GOSIP

<table>
<tr><td colspan="2">FTAM
ACSE</td><td colspan="2">X.400 (1988)
RTS</td></tr>
<tr><td colspan="2">ISO Presentation
8327</td><td colspan="2"></td></tr>
<tr><td colspan="2">ISO Session
8327</td><td colspan="2">ISO Session
8327</td></tr>
<tr><td colspan="4">ISO Transport Protocol 0, 2 and 4
8073</td></tr>
<tr><td>CLNS
8473</td><td colspan="3">CONS
8878/8881/8208 X.25 (1984)</td></tr>
<tr><td>LLC-1
8802-2</td><td colspan="2">LLC-2
8802-2</td><td>X.25 (1984)
7776</td></tr>
<tr><td>8802-3</td><td>8802-5</td><td>9314</td><td>X.21</td></tr>
</table>

Figure 9.2

The US government, via the National Institute for Standards and Technology (NIST), has adopted a profile of OSI called GOSIP (Government OSI Protocols). This is based on the MAP/TOP profiles of OSI which have been adopted on a limited scale by industry.

As a part of a wider move towards using Open Systems wherever possible in public procurement of computing services and products, the US government has decided that OSI will replace TCP/IP as the government communications standard. This decision took effect from 1st August 1990 - in theory future procurements must mandate OSI. The difficulty of migrating from TCP/IP to OSI means that it will take many years to implement this decision fully.

Figure 9.2 shows elements of GOSIP 3, which is the current profile.

Virtually every other Western government, and most of the Open Systems bodies such as X/Open have committed to moving from TCP/IP to OSI. GOSIP 3 has become the *de facto* profile for this - because of the size of the US government computing budget and because of the development of migration tools based on GOSIP.

Migration to OSI

In the late 1980s the Internet was expected to migrate from TCP/IP to OSI during the early 1990s.

Number of R&D programmes underway to develop migration tools.

Number of experimental OSI services already available within Internet community.

Migration is not going to be easy! The two systems will have to co-exist for some years.

One key limitation to TCP/IP is the addressing space.

Following on from the US government decision to adopt OSI as the basis of its communications policy, based on GOSIP, the IAB has decided that the Internet should migrate to OSI. This is largely at the behest of the US DoD who have decided to adopt OSI at the earliest opportunity. Their views are set out in RFC 1039, which states that the DoD will adopt GOSIP as a *...co-standard with the DoD protocols...* (TCP/IP) and goes on to outline the role that the DoD intends to play by setting up and joining various working parties. This work has been going on for some time now, and a number of papers have been published (and R&D grants awarded) that identify a number of migration strategies. Before examining these in detail, some terms are defined below:

Coexistence The sharing of network resources without the capacity to interoperate.

Migration Gradual transition from TCP - OSI, implying some period of interoperability.

Notes:

Migration Techniques

Number of alternative approaches to facilitate co-existence and migration.

- Shared networks
- Encapsulation *(Tunnel)*
- Dual stacks
- Mixed or similar stacks
- Gateways

A number of coexistance and migration techniques have been identified:

Shared Networks

Different protocol stacks can often share the same network resources without interfering with one another - very common on Ethernet. This does not provide any interoperation.

Encapsulation

Sometimes called *tunnelling* - the host network carries traffic for the other protocol transparently. A good example of this used all the time is TCP/IP over X.25. This offers no interoperability.

Dual Stacks

A dual stacked host has both protocol stacks implemented in software and is able to talk to either network simultaneously. The user can choose the most appropriate protocol. This does not necessarily provide interoperability.

Notes:

Mixed Stacks

This allows a combination of both protocol stacks to be used together. For example TCP can provide a pseudo-network service to carry TPO, and hence any OSI application such as X.400 or FTAM can run over TCP/IP. There are a number of approaches to this currently under experimentation.

Gateways

Gateways provide a conversion between two protocols. Transport gateways allow either TCP or OSI applications to run over either network. Application level gateways swap between applications e.g. X.400/SMTP.

Notes:

Sharing Physical Networks

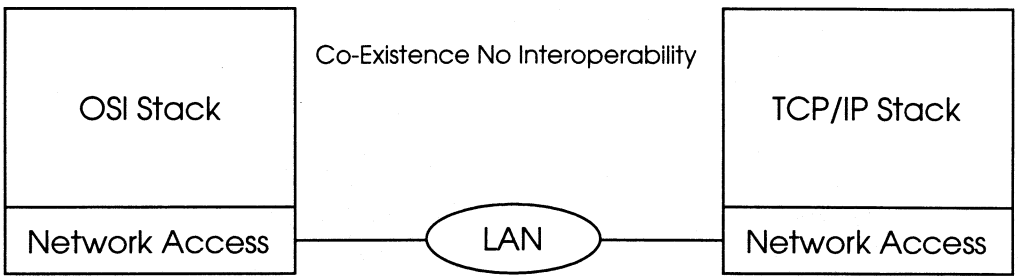


Figure 9.3

Shared Networks

This approach allows sharing of the physical network, such as a LAN. It does not provide any interoperability between the two stacks (which are incompatible), nor does it allow routing of packets between the two stacks. In a LAN environment this does not matter too much as the majority of communications are point-to-point, and so the two systems can ignore each other.

Notes:

Tunnelling

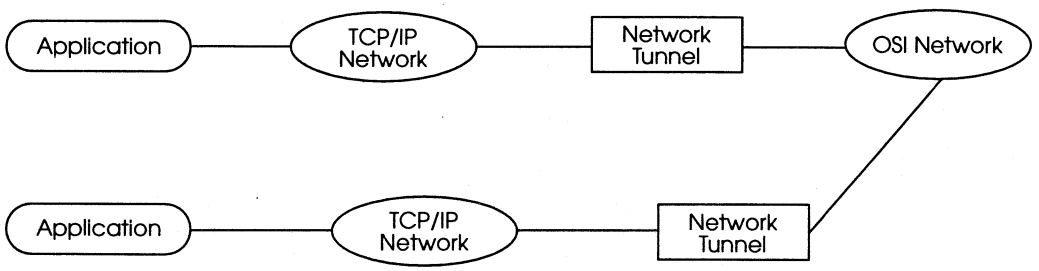


Figure 9.4

Tunnelling is an extension of the idea behind shared networks - we use one network to carry the traffic of the other protocol stack. In the example shown the OSI network acts as a carrier for the TCP/IP traffic. An excellent example of this happening today is the X.25 networks used by TCP/IP (very few X.25 networks actually carry OSI traffic, but so what!). The IP over X.25 protocol means that this usage is completely transparent to other traffic.

An alternative example, also based on X.25, would see the tunnel being established by a pair of half-bridges, connecting two LANs together via an X.25 virtual circuit.

Notes:

Dual Stacks

This is a very simple approach to migration at a user level. It offers no interoperability from a network viewpoint, and yet the user is offered a choice of protocol stacks. In practice this means that he has to use TELNET to access the UNIX host, and VT to access the VAX host (running DECnet/OSI).

This is certainly a valid solution but in many cases it will be necessary to hide away the implementation details from the user behind a suitable front-end - with the inevitable loss of functionality.

Notes:

Mixed Stacks

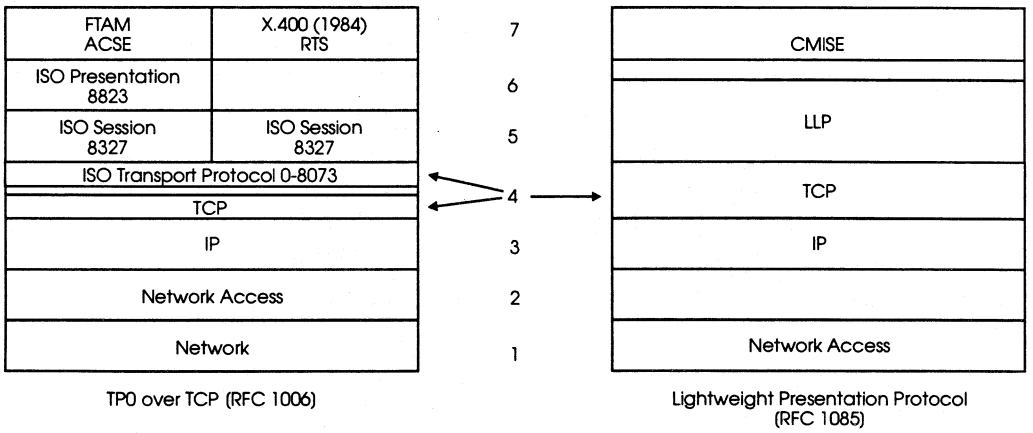


Figure 9.5

This approach is proving quite popular - the provision of OSI applications (much richer than the TCP/IP equivalents) over a TCP/IP transport. The two examples above show:

TPO over TCP

TPO is a very simple OSI transport protocol that assumes that the underlying network is completely reliable (Class A) - this is a good assumption for TCP. The provision of a TPO service means that the upper layers of GOSIP can be supported (as if the underlying TCP/IP network were an X.25 network). This approach is described in RFC 1006 and has been implemented by a number of Internet sites.

Notes:

Lightweight Presentation Protocol

LPP - the *Lightweight Presentation Protocol* - is described in RFC1085. It is a *fudge* piece of software that implements the minimum requirements of the ISO Presentation layer - sufficient to support CMIS, and hence provide support for the CMOT management protocols over TCP/IP. This is a lot *dirtier* than providing TPO over TCP, the only genuine OSI bit in here is CMIS (more specifically ACSE, ROSE and CMISE).

The original objective of this approach was to encourage users to adopt OSI-based applications and then chop the transport from TCP to OSI-TP4 one night when they are not looking! This may well happen, but in the meantime the analogue of this is happening as a result of the use of TLI/XTI - TCP/IP applications are being supported over OSI networks, which will probably lengthen their life considerably!

Notes:

Transport Relay

Transport Gateway

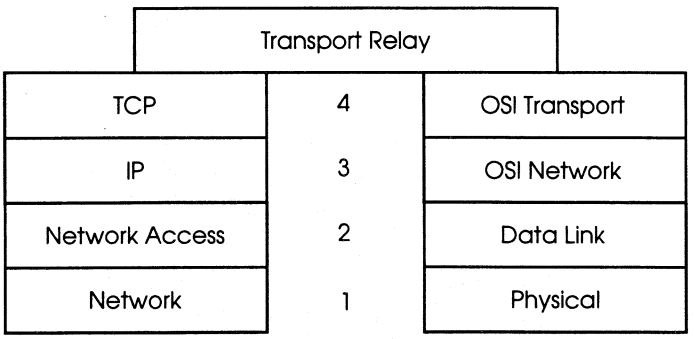


Figure 9.6

Conceptually, the transport gateway is our first example of true interoperation between TCP and OSI. It relies on the close similarity between TCP and TP4 and offers a gateway (or protocol conversion) between the two systems.

What this means in real terms is that TCP/IP and OSI-based systems can interconnect, providing they use the same application protocols they will be able to interoperate. In other words we are half way towards internetworking in a mixed TCP/OSI environment.

The ability of applications to be protocol independent (via TLI/XTI) means that it will be possible to have either a TCP or OSI application on a host talking to its peer on another host. The actual communication path may have been established and managed via either TCP or OSI - this is transparent to the application.

Notes:

X.400/RFC 822 Gateway

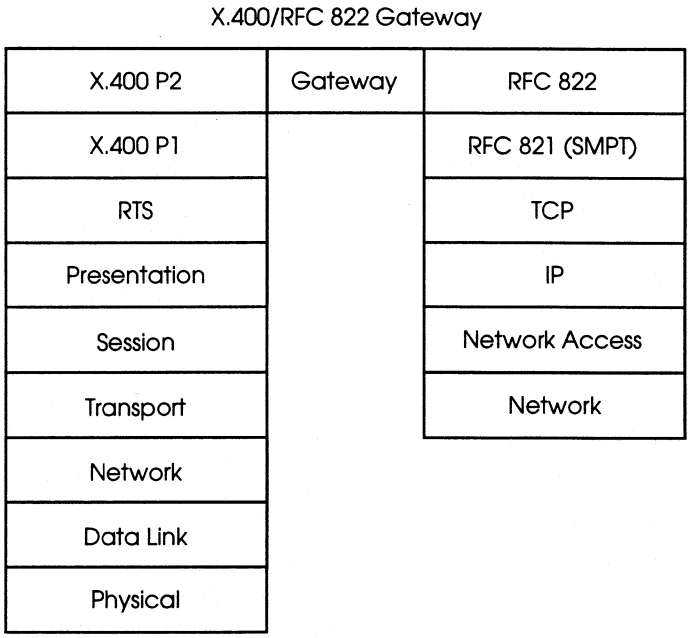


Figure 9.7

Here we illustrate an application gateway - a protocol convertor that links together two different applications and allows them to interoperate. If we combine transport and application gateways together we have full interoperation between TCP/IP and OSI.

The transport gateway exists, developed by Wollongong Group; this mail application gateway exists, developed by University College London (UCL). We therefore can offer a degree of interoperation between the two items - but we also need application gateways to support other core applications. This may not be as easy as mail, because mail represents a very simple store and forward application. Consider the problems of other gateways:

VT/Telnet

A gateway between Telnet and the OSI equivalent (Virtual Terminal Protocol) would be very difficult to provide, although technically feasible to design. Imagine the amount of CPU power needed to handle an *on-the-fly* conversion between the two protocols - it would be quite significant. It is more likely that gateway computers will be provided to enable users to login using, say Telnet, and then establish a VT session with an OSI host. Even simpler is the mixed stack or dual stack approach.

FTAM/FTP

Conceptually, file transfer is merely a bigger version of mail transfer (actually mail transfer developed as a subset of file transfer). In reality the uncertainty over the size and shape of a file transfer means that immense resource would have to be provided to support FTAM/FTP interworking. A solution under discussion is to provide a *File Transfer Gateway* host; files would be transferred to the host using say, FTP. They would then be transferred onward using FTAM either automatically by the gateway or by the user logging into the gateway and completing the transfer manually.

Within the Internet it may be possible to provide the resources necessary to provide dedicated gateways of the type illustrated above, but most commercial organisations would have difficulty justifying the expenditure.

Notes:

Designing for the Future

TCP/IP will remain an important protocol for many years to come. As each year goes by when OSI is not widely implemented, the installed base of TCP/IP systems grows and hence ensures that the life of the protocols is extended.

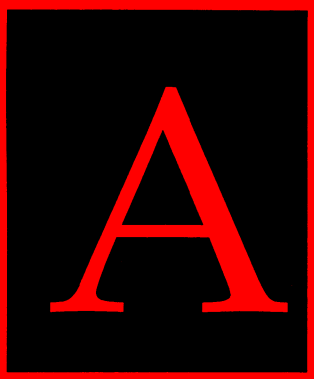
To guard your investment in applications and networks into the future you are recommended to pursue a *defensive* design policy:

1. Stick to standardised network components. For example, if installing a new LAN make sure that not only the components are IEEE 802.n-based, but also that TCP/IP is running via LLC/SNAP as this will provide transparent coexistence with OSI and ease interworking options. You may have to force your supplier to provide a LLC/SNAP implementation today!

2. Develop network applications using a standards profile such as XPG from X/Open. This will ensure that your applications can be migrated to OSI in the future, if required (as well as offering portability between different vendors hardware).

3. Consider the possibility of migrating to OSI in the mid-term. When the OSI bandwagon begins to roll it will attract huge quantities of investment in terms of R&D cash and talent - you may well want to take advantage of that investment. One thing is for sure - the cash mountain provided by the US government over the years for the development of TCP/IP is reducing; industry will only invest in TCP/IP for as long as OSI investment is unattractive.

There is a good future for TCP/IP for many years yet, but it pays to be prudent and ensure that investment is maximised.



Appendix A
Internet Activities Board

Network Working Group
Request for Comments: 1160
Obsoletes: RFC 1120

V. Cerf
NRI
May 1990

The Internet Activities Board

Status of this Memo

This RFC provides a history and description of the Internet Activities Board (IAB) and its subsidiary organizations. This memo is for informational use and does not constitute a standard. This is a revision of RFC 1120. Distribution of this memo is unlimited.

1. Introduction

In 1968, the U.S. Defense Advanced Research Projects Agency (DARPA) initiated an effort to develop a technology which is now known as packet switching. This technology had its roots in message switching methods, but was strongly influenced by the development of low-cost minicomputers and digital telecommunications techniques during the mid-1960's [BARAN 64, ROBERTS 70, HEART 70, ROBERTS 78]. A very useful survey of this technology can be found in [IEEE 78].

During the early 1970's, DARPA initiated a number of programs to explore the use of packet switching methods in alternative media including mobile radio, satellite and cable [IEEE 78]. Concurrently, Xerox Palo Alto Research Centre (PARC) began an exploration of packet switching on coaxial cable which ultimately led to the development of Ethernet local area networks [METCALFE 76].

The successful implementation of packet radio and packet satellite technology raised the question of interconnecting ARPANET with other types of packet nets. A possible solution to this problem was proposed by Cerf and Kahn [CERF 74] in the form of an internetwork protocol and a set of gateways to connect the different networks. This solution was further developed as part of a research program in internetting sponsored by DARPA and resulted in a collection of computer communications protocols based on the original Transmission Control Protocol (TCP) and its lower level counterpart, Internet Protocol (IP). Together, these protocols, along with many others developed during the course of the research, are referred to as the TCP/IP Protocol Suite [RFC 1140, LEINER 85, POSTEL 85, CERF 82, CLARK 86].

In the early stages of the Internet research program, only a few researchers worked to develop and test versions of the internet protocols. Over time, the size of this activity increased until, in 1979, it was necessary to form an informal committee to guide the technical evolution of the protocol suite. This group was called the Internet Configuration Control Board (ICCB) and was established by Dr. Vinton Cerf who was then the DARPA program manager for the effort. Dr. David C. Clark of the Laboratory for Computer Science at Massachusetts Institute of Technology was named the chairman of this committee.

In January, 1983, the Defense Communications Agency, then responsible for the operation of the ARPANET, declared the TCP/IP protocol suite to be standard for the ARPANET and all systems on the network converted from the earlier Network Control Program (NCP) to TCP/IP. Late that year, the ICCB was reorganized by Dr. Barry Leiner, Cerf's successor at DARPA, around a series of task forces considering different technical aspects of internetting. The reorganized group was named the Internet Activities Board.

As the Internet expanded, it drew support from U.S. Government organizations including DARPA, the National Science Foundation (NSF), the Department of Energy (DOE) and the National Aeronautics and Space Administration (NASA). Key managers in these organizations, responsible for computer networking research and development, formed an informal Federal Research Internet Coordinating Committee (FRICC) to coordinate U.S. Government support for and development and use of the Internet system. The FRICC sponsored most of the U.S. research on internetting, including support for the Internet Activities Board and its subsidiary organizations.

In 1990, the FRICC was reorganized as part of a larger initiative sponsored by the networking subcommittee of the Federal Coordinating Committee on Science, Engineering and Technology (FCCSET). The reorganization created the Federal Networking Council (FNC) and its Working Groups. The membership of the FNC included all the former FRICC members and many other U.S. Government representatives. The first chairman of the FNC is Dr. Charles Brownstein of the National Science Foundation. The FNC is the Federal Government's body for coordinating the agencies that support the Internet. It provides liaison to the Office of Science and Technology Policy (headed by the President's Science Advisor) which is responsible for setting science and technology policy affecting the Internet. It endorses and employs the existing planning and operational activities of the community-based bodies that have grown up to manage the Internet in the United States. The FNC plans to involve user and supplier communities through creation of an external advisory board and will coordinate Internet activities with other Federal initiatives ranging from the Human Genome and Global Change programs to educational applications. The FNC has also participated in planning for the creation of a National Research and Education Network in the United States.

At the international level, a Coordinating Committee for Intercontinental Research Networks (CCIRN) has been formed which includes the U.S. FNC and its counterparts in North America and Europe. Co-chaired by the executive directors of the FNC and the European Association of Research Networks (RARE), the CCIRN provides a forum for cooperative planning among the principal North American and European research networking bodies.

2. Internet Activities Board

The Internet Activities Board (IAB) is the coordinating committee for Internet design, engineering and management. The Internet is a collection of over two thousand of packet switched networks located principally in the U.S., but also in many other parts of the world, all interlinked and operating using the protocols of the TCP/IP protocol suite. The IAB is an independent committee of researchers and professionals with a technical interest in the health and evolution of the Internet system. Membership changes with time to adjust to the current realities of the research interests of the participants, the needs of the Internet system and the concerns of constituent members of the Internet.

IAB members are deeply committed to making the Internet function effectively and evolve to meet a large scale, high speed future. New members are appointed by the chairman of the IAB, with the advice and consent of the remaining members. The chairman serves a team of two years and is elected by the members of the IAB. The IAB focuses on the TCP/IP protocol suite, and extensions to the Internet system to support multiple protocol suites.

The IAB has two principal subsidiary task forces:

1) Internet Engineering Task Force (IETF)

2) Internet Research Task Force (IRTF)

Each of these Task Forces is led by a chairman and guided by a Steering Group which reports to the IAB through its chairman. Each task force is organized, by the chairman, as required, to carry out its charter. For the most part, a collection of Working Groups carries out the work program of each Task Force.

All decisions of the IAB are made public. The principal vehicle by which IAB decisions are propagated to the parties interested in the Internet and its TCP/IP protocol suite is the Request for Comment (RFC) note series. The archival RFC series was initiated in 1969 by Dr. Stephen D. Crocker as a means of documenting the development of the original ARPANET protocol suite [RFC 1000]. The editor-in-chief of this series, Dr. Jonathan B. Postel, has maintained the quality of and managed the archiving of this series since its inception. A small proportion of the RFCs document Internet standards. Most of them are intended to stimulate comment and discussion. The small number which document standards are especially marked in a "status" section to indicate the special status of the document. An RFC summarizing the status of all standard RFCs is published regularly [RFC 1140].

RFCs describing experimental protocols, along with other submissions whose intent is merely to inform, are typically submitted directly to the RFC editor. A Standard Protocol starts out as a Proposed Standard and may be promoted to Draft Standard and finally Standard after suitable review, comment, implementation and testing.

Prior to publication of a Proposed Standard RFC, it is made available for comment through an on-line Internet-Draft directory. Typically, these Internet-Drafts are working documents of the IAB of the working groups of the Internet Engineering and Research Task Forces. Internet-Drafts are either submitted to the RFC editor for publication or discarded within 3-6 months. Prior to promotion to Draft Standard or Standard, an Internet-Draft publication and review cycle may be initiated if significant changes to the RFC are contemplated.

The IAB performs the following functions:

1) Sets Internet Standards,
2) Manages the RFC publications process,
3) Reviews the operation of the IETF and IRTF,
4) Performs strategic planning for the Internet, identifying long-range problems and opportunities,
5) Acts as an international technical policy liaison and representative for the Internet community, and
6) Resolves technical issues which cannot be treated within the IETF or IRTF frameworks.

To supplement its work via electronic mail, the IAB meets quarterly to review the condition of the Internet, to review and approve proposed changes or additions to the TCP/IP suite of protocols, to set technical development priorities, to discuss policy matters which may need the attention of the Internet sponsors, and to agree on the addition or retirement of IAB members and on the addition or retirement of task forces reporting to the IAB. Typically, two of the quarterly meetings are by means of video teleconferencing (provided, when possible, through the experimental Internet packet video-conferencing system). The minutes of the IAB meeting are published in the Internet Monthly on-line report.

The IAB membership is currently as follows:

Vinton Cerf/CNRI	Chairman
Robert Braden/USC-ISI	Executive Director
David Clark/MIT-LCSIRTF	Chairman
Phillip Gross/CNRIIETF	Chairman
Jonathan Postel/USC-ISIRFC	Editor
Hans-Werner Braun/Merit	Member
Lyman Chapin/DG	Member
Stephen Kent/BBN	Member
Anthony Lauck/Digital	Member
Barry Leiner/RIACS	Member
Daniel Lynch/Interop, Inc.	Member

3. The Internet Engineering Task Force

The Internet has grown to encompass a large number of widely geographically dispersed networks in academic and research communities. It now provides an infrastructure for a broad community with various interests. Moreover, the family of Internet protocols and system components has moved from experimental to commercial development. To help coordinate the operation, management and evolution of the Internet, the IAB established the Internet Engineering Task Force (IETF). The IETF is chaired by Mr. Phillip Gross and managed by its Internet Engineering Steering Group (IESG). The IAB has delegated to the IESG the general responsibility for making the Internet work and for the resolution of all short- and mid-range protocol and architectural issues required to make the Internet function effectively.

The charter of the IETF includes:

1) Responsibility for specifying the short and mid-term Internet protocols and architecture and recommending standards for IAB approval.

2) Provision of a forum for the exchange of information within the Internet community.

3) Identification of pressing and relevant short- to mid-range operational and technical problem areas and convening of Working Groups to explore solutions.

The Internet Engineering Task Force is a large open community of network designers, operators, vendors, and researchers concerned with the Internet and the Internet protocol suite. It is organized around a set of eight technical areas, each managed by a technical area director. In addition to the IETF Chairman, the area directors make up the IESG membership. Each area director has primary responsibility for one area of Internet engineering activity, and hence for a subset of the IETF Working Groups. The area directors have jobs of critical importance and difficulty and are selected not only for their technical expertise but also for their managerial skills and judgement. At present, the eight technical areas and chairs are:

1) Applications - Russ Hobby/UC-Davis

2) Host and User Services - Craig Partridge/BBN

3) Internet Services - Noel Chiappa/Consultant

4) Routing - Robert Hinden/BBN

5) Network Management - David Crocker/DEC

6) OSI Integration - Ross Callon/DEC and Robert Hagens/UWisc.

7) Operations - Phill Gross/CNRI (Acting)

8) Security - Steve Crocker/TIS

The work of the IETF is performed by subcommittees known as Working Groups. There are currently more than 40 of these. Working Groups tend to have a narrow focus and a lifetime bounded by completion of a specific task, although there are exceptions. The IETF is a major source of proposed protocol standards, for final approval by the IAB. The IETF meets quarterly and extensive minutes of the plenary proceedings as well as reports from each of the working groups are issued by the IAB Secretariat at the Corporation for National Research Initiatives.

4. The Internet Research Task Force

To promote research in networking and the development of new technology, the IAB established the Internet Research Task Force (IRTF).

In the area of network protocols, the distinction between research and engineering is not always clear, so there will sometimes be overlap between activities of the IETF and the IRTF. There is, in fact, considerable overlap in membership between the two groups. This overlap is regarded as vital for cross-fertilization and technology transfer. In general, the distinction between research and engineering is one of viewpoint and sometimes (but not always) time-frame. The IRTF is generally more concerned with understanding than with products or standard protocols, although specific experimental protocols may have to be developed, implemented and tested in order to gain understanding.

The IRTF is a community of network researchers, generally with an Internet focus. The work of the IRTF is governed by its Internet Research Steering Group (IRSG). The chairman of the IRTF and IRSG is David Clark. The IRTF is organized into a number of Research Groups (RGs) whose chairs of these are appointed by the chairman of the IRSG. The RG chairs and others selected by the IRSG chairman serve on the IRSG. These groups typically have 10 to 20 members, and each covers a broad area of research, pursuing specific topics, determined at least in part by the interests of the members and by recommendations of the IAB.

The current members of the IRSG are as follows:

David Clark/MIT LCS	Chairman
Robert Braden/USC-ISI	End-to-End Services
Douglas Comer/PURDUE	Member-at-Large
Deborah Estrin/USC	Autonomous Networks
Stephen Kent/BBN	Privacy and Security
Keith Lantz/Consultant	Collaboration Technology
David Mills/UDEL	Member-at-Large

5. The Near-term Agenda of the IAB

There are seven principal foci of IAB attention for the period 1989 - 1990:

1) Operational Stability

2) User Services

3) OSI Coexistence

4) Testbed Facilities

5) Security

6) Getting Big

7) Getting Fast

Operational stability of the Internet is a critical concern for all of its users. Better tools are needed for gathering operational data, to assist in fault isolation at all levels and to analyze the performance of the system. Opportunities abound for increased cooperation among the operators of the various Internet components [RFC 1109]. Specific, known problems should be dealt with, such as implementation deficiencies in some versions of the BIND domain name service resolver software. To the extent that the existing Exterior Gateway Protocol (EGP) is only able to support limited topologies, constraints on topological linkages and allowed transit paths should be enforced until a more general Inter-Autonomous System routing protocol can be specified. Flexibility for Internet implementation would be enhanced by the adoption of a common internal gateway routing protocol by all vendors of Internet routers. A major effort is recommended to achieve conformance to the Host Requirements RFCs which were published in the fourth quarter of calendar 1989.

Among the most needed user services, the White Pages (electronic mailbox directory service) seems the most pressing. Efforts should be focused on widespread deployment of these capabilities in the Internet by mid-1990. The IAB recommends that existing white pages facilities and newer ones, such as X.500, be populated with up-to-date user information and made accessible to Internet users and users of other systems (e.g., commercial email carriers) inked to the Internet. Connectivity with commercial electronic mail carriers should be vigorously pursued, as well as links to other network research communities in Europe and the rest of the world.

Development and deployment of privacy-enhanced electronic mail software should be accelerated in 1990 after release of public domain software implementing the private electronic mail standards [RFC 1113, RFC 1114 and RFC 1115]. Finally, support for new or enhanced applications such as computer-based conferencing, multi-media messaging and collaboration support systems should be developed.

The National Network Testbed (NNT) resources planned by the FRICC should be applied to support conferencing and collaboration protocol development and application experiments and to support multi-vendor router interoperability testing (e.g., interior and exterior routing, network management, multi-protocol routing and forwarding).

With respect to growth in the Internet, architectural attention should be focused on scaling the system to hundreds of millions of users and hundreds of thousands of networks. The naming, addressing, routing and navigation problems occasioned by such growth should be analyzed. Similarly, research should be carried out on analyzing the limits to the existing Internet architecture, including the ability of the present protocol suite to cope with speeds in the gigabit range and latencies varying from microseconds to seconds in duration.

The Internet should be positioned to support the use of OSI protocols by the end of 1990 or sooner, if possible. Provision for multi-protocol routing and forwarding among diverse vendor routes is one important goal. Introduction of X.400 electronic mail services and interoperation with RFC 822/SMTP [RFC 822, RFC 821, RFC 987, RFC 1026, and RFC 1148] should be targeted for 1990 as well. These efforts will need to work in conjunction with the White Pages services mentioned previously.

The IETF, in particular, should establish liaison with various OSI working groups (e.g., at NIST, RARE, Network Management Forum) to coordinate planning for OSI introduction into the Internet and to facilitate registration of information pertinent to the Internet with the various authorities responsible for OSI standards in the United States.

Finally, with respect to security, a concerted effort should be made to develop guidance and documentation for Internet host managers concerning configuration management, known security problems (and their solutions) and software and technologies available to provide enhanced security and privacy to the users of the Internet.

REFERENCES

[BARAN 64] Baran, P., et al, "On Distributed Communications", Volumes I-XI, RAND Corporation Research Documents, August 1964.

[CERF 74] Cerf V., and R. Kahn, "A Protocol for Packet Network Interconnection", IEEE Trans. on Communications, Vol. COM-22, No. 5, pp. 637-648, May 1974.

[CERF 82] Cerf V., and E. Cain, "The DoD Internet Protocol Architecture", Proceedings of the SHAPE Technology Centre Symposium on Interoperability of Automated Data Systems, November 1982. Also in Computer Networks and ISDN, Vol. 17, No. 5, October 1983.

[CLARKE 86] Clark, D., "The Design Philosophy of the DARPA Internet Protocols", Proceedings of the SIGCOMM '88 Symposium, Computer Communications Review, Vol. 18, No. 4, pp. 106-114, August 1988.

[HEART 70] Heart, F., Kahn, R., Ornstein, S., Crowther, W., and D. Walden, "The Interface Message Processor for the ARPA Computer Network", AFIPS Conf. Proc. 36, pp. 551-567, June 1970.

[IEEE 78] Kahn, R. (Guest Editor), Uncapher, E. and H. Van Trees, (Associate Guest Editors), Proceedings of the IEEE, Special Issue on Packet Communication Networks, Vol. 66, No. 11, pp. 1303-1576, November 1978.

[IEEE 87] Leiner, B. (Guest Editor), Nielson, D., and F. Tobagi (Associate Guest Editors), Proceedings of the IEEE, Special Issue on Packet Radio Networks, Vol. 75, No. 1, pp. 1-272, January 1987.

[LEINER 85] Leiner, B., Cole, R., Postel, J., and D. Mills, "The DARPA Protocol Suite", IEEE INFOCOM 85, Washington, D.C., March 1985. Also in IEEE Communications Magazine, March 1985.

[METCALFE 76] Metcalfe, R., and D. Boggs, "Ethernet: Distributed Packet for Local Computer Networks", Communications of the ACM, Vol. 19, No. 7, pp. 395-404, July 1976.

[POSTEL 85] Postel, J., "Internetwork Applications Using the DARPA Protocol Suite", IEEE INFOCOM 85, Washington, D.C., March 1985.

[RFC 821] Postel, J., "Simple Mail Transfer Protocol", RFC 821, USC/Information Sciences Institute, August 1982.

[RFC 822] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", RFC 822, University of Delaware, August 1982.

[RFC 987] Kille, S., "Mapping between X.400 and RFC 822", University College London, June 1986.

[RFC 1000] Reynolds, J., and J. Postel, "The Request for Comments Reference Guide", RFC 1000, USC/Information Sciences Institute, August 1987.

[RFC 1026] Kille, S., "Addendum to RFC 987: (Mapping between X.400 and RFC 822)", RFC 1026, University College London, September 1987.

[RFC 1109] Cerf, V., "Report of the Second Ad Hoc Network Management Review Group", RFC 1109, NRI, August 1989.

[RFC 1113] Linn, J., "Privacy Enhancement for Internet Electronic Mail: Part I -- Message Encipherment and Authentication Procedures", RFC 1113, IAB Privacy Task Force, August 1989.

[RFC 1114] Kent, S., and J. Linn, "Privacy Enhancement for Internet Electronic Mail: Part II -- Certificate-based Key Management", RFC 1114, IAB Privacy Task Force, August 19889.

[RFC 1115] Linn, J., "Privacy Enhancement for Internet Electronic Mail: Part III -- Algorithms, Modes and Identifiers", RFC 1115, IAB Privacy Task Force, August 1989.

[RFC 1140] Postel, J., Editor, "IAB Official Protocol Standards", RFC 1140, Internet Activities Board, May 1990.

[RFC 1148] Kille, S., "Mapping between X.400 (1988)/ISO 10021 and RFC 822", RFC 1048, UCL, March 1990.

[ROBERTS 70] Roberts, L., and B. Wessler, "Computer Network Development to Achieve Resource Sharing", pp. 543-549, Proc. SJCC 1970.

[ROBERTS 78] Roberts, L., "Evolution of Packet Switching", Proc. IEEE, Vol. 66, No. 11, pp. 1307-1313, November 1978.

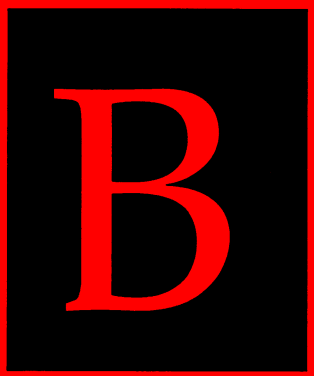
Note: RFCs are available from the Network Information Centre at SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, (1-800-235-3155), or on-line via anonymous file transfer from NIC.DDN.MIL.

Author's Address

Vinton G. Cerf
Corporation for National Research Initiatives
1895 Preston White Drive, Suite 100
Reston, VA 22091

Phone: (703) 620-8990

EMail: VCERF@NRI.RESTON.VA.US



Appendix B
Important RFCs

Important RFCs

This section is organised by topic, followed by a list of RFC numbers and associated titles. Some RFCs may appear in several sections - this is quite deliberate. This is not, by any means, an exhaustive list. See a current index of RFCs for that - the index indicates whether an RFC is current, updated or obsoleted.

Administration

1166 Internet numbers

1160 Internet Activities Board

1140 IAB official protocol standards

1118 Hitchhikers Guide to the Internet

1060 Assigned numbers

1039 DoD statement on OSI protocols

Network Management

1173 Responsibilities of Host and Network managers

1161 SNMP over OSI

1162 Connectionless Network Protocol & End System to Intermediate System MIB

1158 MIB for TCP/IP-based internets: MIB II

1157 Simple Network Management Protocol (SNMP)

1156 MIB for TCP/IP-based internets

1155 SMI for TCP/IP-based internets

1147 Tools for monitoring and debugging TCP/IP internets

1095 Common Management Information Services over TCP/IP (CMOT)

1089 SNMP over Ethernet

1052 IAB recommendations for development of Internet management standards

Miscellaneous Servers

1165 Network Time Protocol over OSI ROS

1129 Internet time synchronisation: The Network Time Protocol

1119 Network Time Protocol (version 2)

1094 NFS: Network File System specification

1084 BOOTP vendor information extensions

1057 Remote Procedure Call specification (version 2)

951 Bootstrap Protocol: BOOTP

931	Authentication server
867	Daytime Protocol
866	Active Users
865	Quote of the Day Protocol
864	Character Generator Protocol
863	Discard Protocol
862	Echo Protocol

Internet Layer

1134	Administrative domains & routing domains: A model for routing in the Internet
1160	Internet Activities Board
1103	Proposed standards for transmission of IP datagrams over FDDI networks
1088	Transmission of IP datagrams over NETBIOS Networks
1063	IP MTU discovery options
1058	Routing Information Protocol (RIP)
1054	Transmission of IP datagrams over serial lines: SLIP
1051	Transmission of IP datagrams over ARCNET networks
1044	IP on HYPERchannel
1042	Transmission of IP datagrams over IEEE 802 networks
903	Reverse Address Resolution Protocol: RARP
894	Transmission of IP datagrams over Ethernet networks
893	Trailer encapsulations
877	Transmission of IP datagrams over public networks: IP-X25
826	Ethernet Address Resolution Protocol: ARP
792	Internet Control Message Protocol: ICMP
791	Internet Protocol

TCP-OSI Migration

1165	Network Time Protocol over OSI ROS
1162	Connectionless Network Protocol & End System to Intermediate System MIB
1148	Mapping between X.400 (1988) and RFC822
1139	Echo function for ISO 8473
1138	Mapping between X.400 (1988) and RFC822
1086	ISO-TPO bridge between TCP and X.25

1085	ISO presentation services on top of TCP/IP-based internets
1070	Use of the Internet as a subnetwork for experimenting with the OSI network layer
1069	Guidelines for the use of IP addresses in ISO Connectionless Network Protocol
1039	DoD statement on OSI protocols
1026	Addendum to RFC987: Mapping between X.400 and RFC822
1008	Implementation guide for the ISO transport protocol
1006	ISO transport services on top of TCP
987	Mapping between X.400 (1984) and RFC822
822	Format for Internet text messages
821	Simple Mail Transfer Protocol: SMTP

Transport Layer

793	Transmission Control Protocol: TCP
768	User Datagram Protocol: UDP

Application Layer

1080	Telnet terminal speed option
1079	Telnet terminal speed option
1073	Telnet window size option
1053	Telnet X.3 PAD option
1041	Telnet 3270 regime option
959	File Transfer Protocol
913	Simple File Transfer Protocol
906	Bootstrap loading using TFTP
783	Trivial File Transfer Protocol: TFTP

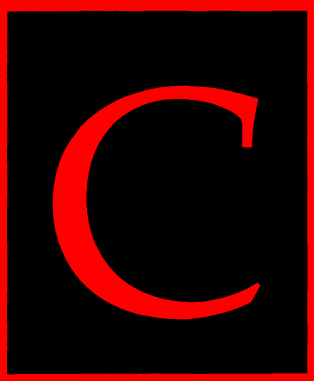
Mail

1148	Mapping between X.400 (1988) and RFC822
1138	Mapping between X.400 (1988) and RFC822
1026	Addendum to RFC987: Mapping between X.400 and RFC822
987	Mapping between X.400 (1984) and RFC822
976	UUCP mail interchange format standard
974	Mail routing and the domain system

Copies of RFCs

Many RFCs are available online; if not, this is indicated by (Not online). Paper copies of all RFCs are available from the NIC, either individually or on a subscription basis (for more information contact NIC@NIC.DDN.MIL). Online copies are available via FTP or Kermit from NIN.DDN.MIL as RFC:RFC####.TXT or RFC:RFC####.PS (#### is the RFC number without leading zeroes).

Additionally, RFCs may be requested through electronic mail from the automated NIC mail server by sending a message to SERVICE@NIC.DDN.MIL with a subject line of "RFC####" for text versions or a subject line of "RFC####.PS" for postscript versions. To obtain the RFC index, the subject line of your message should read "RFC index".



Appendix C

IPv6 Address Types

Address Type Representation

The specific type of an IPv6 address is indicated by the leading bits in the address. The variable-length field comprising these leading bits is called the Format Prefix (FP). The initial allocation of these prefixes is as follows:

Allocation	Prefix	Fraction of (binary) Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP allocation	0000 001	1/128
Reserved for IPX allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Unassigned	001	1/8
Provider-based Unicast address	010	1/8
Unassigned	011	1/8
Reserved for Neutral-interconnect-based Unicast addresses	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link local use addresses	1111 1110 10	1/1024
Site local use addresses	1111 1110 11	1/1024
Multicast addresses	1111 1111	1/256

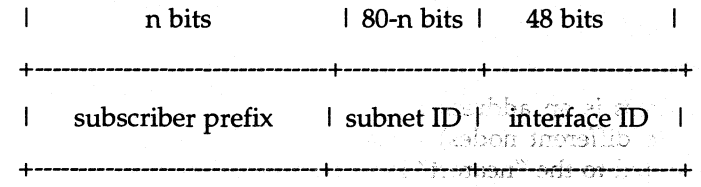
This allocation supports the direct allocation of provider addresses, local use addresses, and multicast addresses. Space is reserved for NSAP addresses, IPX addresses, and neutral-interconnect addresses. The remainder of the address space is unassigned for future use. This can be used for expansion of existing use (e.g. additional provider addresses) or new uses (e.g. separate locators and identifiers). Fifteen percent of the address space is initially allocated. The remaining 85% is reserved for future use.

Unicast addresses are distinguished from multicast addresses by the value of the high-order octet of the addresses: a value of FF (11111111) identifies an address as a multicast address; any other value identifies an address as a unicast address. Anycast addresses are taken from the unicast address space, and are not syntactically distinguishable from unicast addresses.

Unicast Addresses

The IPv6 unicast address is contiguous bit-wise maskable, similar to IPv4 addresses under Class-less Interdomain Routing [CIDR]. There are several forms of unicast address assignment in IPv6, including the global provider based unicast address, the neutral-interconnect unicast address, the NSAP address, the IPX hierarchical address, the site-local-use address, the link-local-use address, and the Ipv4-capable host address.

IPv6 nodes may have considerable or little knowledge of the internal structure of the IPv6 address, depending on the role the node plays (for instance, host versus router). At a minimum, a node may consider that unicast addresses (including its own) have no internal structure. An example of a Unicast address format which will likely be common on LANs and other environments where IEEE 802 MAC addresses are available is:



Where the 48-bit Interface ID is an IEEE-802 MAC address. The use of IEEE 802 MAC addresses as an interface ID is expected to be very common in environments where nodes have an IEEE 802 MAC address. In other environments, where IEEE 802 MAC addresses are not available, other types of link layer addresses can be used, such as E.164 addresses, for the interface ID. The inclusion of a unique global interface identifier, such as an IEEE MAC address, makes possible a very simple form of auto-configuration of addresses.

Unspecified Address

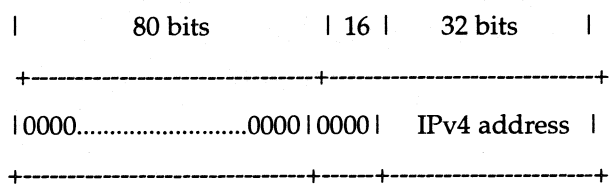
The address 0:0:0:0:0:0:0:0 is called the unspecified address. It must never be assigned to any node. It indicates the absence of an address. One example of its use is in the source address field of any Ipv6 datagrams sent by an initialising host before it has learned its own address. The unspecified address must not be used as the destination address of IPv6 datagrams or in IPv6 Routing Headers.

The Loopback Address

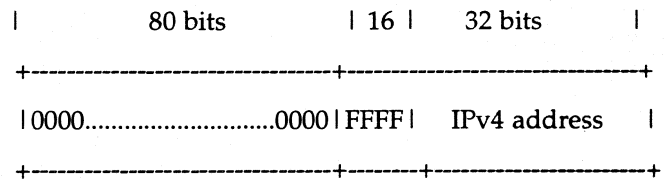
The unicast address 0:0:0:0:0:0:0:1 is called the loopback address. It may be used by a node to send an IPv6 datagram to itself. It may never be assigned to any interface.

IPv6 Addresses with Embedded IPv4 Addresses

The IPv6 transition mechanisms include a technique for hosts and routers to dynamically tunnel IPv6 packets over IPv4 routing infrastructure. IPv6 nodes that utilise this technique are assigned special IPv6 unicast addresses that carry an IPv4 address in the low-order 32-bits. This type of address is termed an "IPv4-compatible IPv6 address" and has the format:



A second type of IPv6 address which holds an embedded IPv4 address is also defined. This address is used to represent the addresses of IPv4-only nodes (those that *do not* support IPv6) as IPv6 addresses. This type of address is termed an "IPv4-mapped IPv6 address" and has the format:



Anycast Addresses

An IPv6 anycast address is an address that is assigned to more than one interface (typically belonging to different nodes), with the property that a packet sent to an anycast address is routed to the "nearest" interface having that address, according to the routing protocols' measure of distance.

Anycast addresses are allocated from the unicast address space, using any of the defined unicast address formats. Thus, anycast addresses are syntactically indistinguishable from unicast addresses. When a unicast address is assigned to more than one interface, thus turning it into an anycast address, the nodes to which the address is assigned must be explicitly configured to know that it is an anycast address.

For any assigned anycast address, there is a longest address prefix P that identifies the topological region in which all interfaces belonging to that anycast address reside. Within the region identified by P, each member of the anycast set must be advertised as a separate entry in the routing system (commonly referred to as a "host route"); outside the region identified by P, the anycast address may be aggregated into the routing advertisement for prefix P.

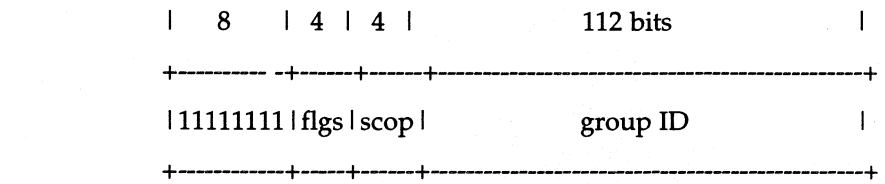
One expected use of anycast addresses is to identify the set of routers belonging to an internet service provider. Such addresses could be used as intermediate addresses in an IPv6 Routing header, to cause a packet to be delivered via a particular provider or

sequence of providers. The following implementation restrictions are imposed on IPv6 anycast addresses:

- An anycast address MUST NOT be used as the source address of an IPv6 packet.
- An anycast address MUST NOT be assigned to an IPv6 host, that is, it may be assigned to an IPv6 router only.

Multicast Addresses

An IPv6 multicast address is an identifier for a group of nodes. A node may belong to any number of multicast groups. Multicast addresses have the following format:



The bit setting 11111111 at the start of the address identifies the address as being a multicast address.

Pre-Defined Multicast Addresses

The reserved multicast addresses that are pre-defined are: FF00::0 through FF0F:00. Other multicasts of importance are:

- All v6 Nodes - FF01:0:0:0:0:0:0:1
- All routers - FF01:0:0:0:0:0:0:2
- All Hosts Addresses - FF01:0:0:0:0:0:0:3

Acronyms

TRAINING

General Acronyms

ACSE	Association Control Service Elements	OSI
AE	Application Entity	
ANSI	American National Standards Institute	
APDU	Applications Protocol Data Unit	OSI
ASCII	American Standard Character Interchange	ANSI
ASE	Application Service Elements	OSI
ASN.1	Abstract Syntax Notation Number 1	OSI
ATM	Asynchronous Transfer Mode	SNA
	Automatic Teller Machine	Banking
BAS	Basic Activity Subset	OSI
BCC	Block Check Character	SNA
BCS	Basic Combined Subset	OSI
BER	Bit Error Range	
BISDN	Broadband Integrated Services Digital Network	ITU-T
BISYNC	Binary Synchronous Control	IBM
BIU	Bus Interface Unit	IEEE
BLU	Basic Link Unit	IEEE
BSC	Binary Synchronous Control	IBM
BSI	British Standards Institute	
BSS	Basic Synchronised Subset	OSI
BTU	Basic Transmission Unit	BTU
CASE	Common Application Service Elements	MAP
CATV	Community Antenna TV (cable TV)	
CBX	Computerised Branch Exchange	
CCIR	International Radio Consultative Committee	
CCITT	International Consultative Committee on Telegraph & Telephony	
CCR	Commitment, Concurrency and Recovery	OSI
CDM	Code Division Multiplexing	ITU-T
CDMA	Code Division Multiple Access	ITU-T
CGI	Computer Graphics Interface	
CGM	Computer Graphics Metafile	
CIM	Computer Integrated Manufacturing	
CLNP	Connectionless Network Protocol	OSI
CLNS	Connectionless Network Service	OSI
CLTS	Connectionless Transport Service	OSI
CONS	Connection Orientated Networks Service	OSI
COTS	Connection Orientated Transport Service	OSI
CRC	Cyclic Redundancy Check	

CSA	Client Server Agent	
CSDN	Circuit Switched Data Network	ITU-T
CSE	Circuit Switched Exchange	ITU-T
CSMA	Carrier Sense Multiple Access	
CSMA/CD	Carrier Sense Multiple Access with Collision Avoidance	IEEE
CT	Cellular Telephone	
	Cordless Telephone	ITU-T
DAMA	Demand Assigned Multiple Access	
DARPA	Defence Advanced Research Projects Agency	TCP/IP
DBX	Digital Branch Exchange	ITU-T
DCE	Date Circuit-terminating Equipment	ITU-T
	Data Communications Equipment	ITU-T
DDCMP	Digital Data Communications Message Protocol	DEC
DEC	Digital Electronics Corporation *(digital)*	DEC
DECT	Digital Exchange Cellular Telephone	
DES	Data Encryption Standard	US DoD
DIB	Directory Information Base	
DLSDU	Data Link Service Data Unit	OSI
DNA	Distributed Network Architecture	DEC
DoD	US Department of Defence	
DPA	DoD Protocol Architecture	US DoD
DQDB	Distributed Queue Dual Bus	
DS	Directory Service	OSI
DSA	Directory Service Agent	OSI
DSMDU	Directory Service Message Data Unit	OSI
DTAM	Document Transfer and Access Management	
DTE	Data Terminal Equipment	ITU-T
DXE	DCE/DTE	
EBCDIC	Extended Binary-Coded Decimal Interchange Code	IBM
ECMA	European Computer Manufacturers Association	
EDIF	Electronic Design Interface Format	
EFT	Electronic Fund Transfer	Banking
EFTPOS	Electronic Fund Transfer at the Point Of Sale	Banking
EIA	Electronic Industries Association	
EPA	Enhanced Performance Architecture	MAP
FADU	File Access Data Unit	

FCS	Frame Check Sequence	OSI
FDDI	Fibre Distributed Data Interface	IEEE
FDM	Frequency Division Multiplexing	ITU-T
FDMA	Frequency Division Multiple Access	ITU-T
FDX	Full Duplex	
FEP	Front End Processor	IBM
FTAM	File Transfer, Access and Management	OSI
FTP	File Transfer Protocol	TCP/IP
GGP	Gateway to Gateway Protocol	TCP/IP
GKS	Graphics Kernel System	
GoS	Grade of Service	
GSM	Special Group Mobile	
HDLC	High Level Data Link Control	OSI
HDX	Half Duplex	
IAN	Integrated Analogue Network	ITU-T
ICI	Interface Control Information	
IDN	Integrated Digital Network	ITU-T
IDSE	International Data Switching Exchange	BT
IDU	Interface Data Unit	
IEE	Institute of Electrical Engineers (UK)	
IEEE	Institute of Electrical and Electronics Engineers (US)	
IGES	Initial Graphics Exchange Specification	
IMP	Interface Message Processor	
IP	Internet Protocol	TCP/IP
IPDU	Internet Protocol Data Unit	OSI
IRDS	Information Resource Dictionary System	
ISDN	Integrated Services Digital Network	ITU-T
ISO	International Standards Organisation	
ITU	International Telecommunications Union	
JANET	Joint Academic NETwork	
JTM	Job Transfer and Manipulation	OSI
LAN	Local Area Network	
LAN/RM	Local Area Network Reference Model	IEEE
LAP	Link Access Protocol	ITU-T
LAPB	Link Access Protocol Balanced	ITU-T
LAPD	Link Access Protocol - ISDN Channel D	ITU-T
LAPX	Link Access Protocol Extended	ITU-T

MAN	Metropolitan Area Network	
MAP	Manufacturing Automation Protocol	
MHS	Message Handling System	ITU-T/OSI
MTBF	Mean Time Between Failures	
MTBSO	Mean Time Between Service Outrage	
MTTR	Mean Time To Repair	
MTTSR	Mean Time To Service Restoration	
NBS	National Bureau of Standards *(now NIST)*	
NCC	National Computer Centre	
NCP	Network Control Protocol	SNA
NISDN	Narrowband Integrated Services Digital Network	ITU-T
NIST	National Institute of Standards and Technology *(was NBS)*	
NMAP	Network Manager Application Process	
NMS	Network Management System	
NPAI	Network Protocol Address Information	
NSAP	Network Service Access Point	OSI
NSDU	Network Service Data Unit	OSI
ODA	Office Document Architecture	
ODIF	Office Document Interchange Format	
OEM	Original Equipment Manufacturer	
OLTP	On-Line Transaction Processing	
OSI	Open System Interconnection	ISO
OSI/RM	Open System Interconnection Reference Model	ISO
PABX	Private Automatic Branch Exchange	
PAD	Packet Assembler/Disassembler	ITU-T
PBX	Private Branch Exchange	
PCI	Protocol Control Information	
PDN	Public Data Network	
PDU	Protocol Data Unit	OSI
PhSDU	Physical Service Data Unit	
PKC	Public Key Cryptography	
POS	Point Of Sale	Banking
PPDU	Presentation Protocol Data Unit	OSI
PPSDN	Public Packet Switched Data Network	
PSAP	Presentation Service Access Point	OSI
PSDU	Presentation Service Data Unit	OSI
PSDN	Public *(packet)* Switched Data Network	

PSE	Packet Switching Exchange	BT
PSS	Packet Switch Stream	BT
PSTN	Public Switched Telephone Network	
PTT	Post, Telegraph, Telephone	
QoS	Quality of Service	
ROSE	Remote Operations Service Element	OSI
RPC	Remote Procedure Call	TCP/IP
RTSE	Reliable Transfer Service Element	
SAP	Service Access Point	OSI
SASE	Specific Service Access Element	OSI
SDL	Specification and Description Language	
SDLC	Synchronous Data Link Control	SNA
SDH	Synchronous Digital Hierarchy	
SDM	Space Division Multiplexing	
SDMA	Space Division Multiple Access	
SDU	Service Data Unit	OSI
SNA	Systems Network Architecture	IBM
SNDCF	SubNetwork Dependent Convergence Facility	
SNDCP	SubNetwork Dependent Convergence Protocol	
SNICF	SubNetwork Independent Convergence Facility	
SNICP	SubNetwork Independent Convergence Protocol	
SNPA	SubNetwork Point of Attachment	
SNR	Signal to Noise Ratio	
SPDU	Session Protocol Data Unit	OSI
SSAP	Session Service Access Point	OSI
SSDU	Session Service Data Unit	OSI
SSM	Spread Spectrum Multiplexing	
SSMA	Spread Spectrum Multiple Access	
SS/TDMA	Satellite Switched Time Division Multiple Access	
STE	Signalling Terminal Exchange	
TCP	Transmission Control Protocol	TCP/IP
TCP/IP	Transmission Control Protocol/Internet Protocol	
TDM	Time Division Multiplexing	ITU-T
TDMA	Time Division Multiple Access	ITU-T
TP	Transport Protocol	

TOP	Technical and Office Protocol	
TPDU	Transport Protocol Data Unit	OSI
TSAP	Transport Service Access Point	OSI
TSDU	Transport Service Data Unit	OSI
UA	User Agent	OSI
UI	User Interface	
VAN	Value Added Network	
VDI	Virtual Device Interface	
VSAT	Very Small Aperture Terminal	
VT	Virtual Terminal	OSI
VTP	Virtual Terminal Protocol	
WAN	Wide Area Network	

TCP/IP Specific Acronyms

ARP	Address Resolution Protocol
ARPA	Advanced Projects Research Agency
ARQ	Automatic Repeat Request
BER	Basic Encoding Rules
BOOTP	Bootstrap Protocol
BSD	Berkeley System Distribution
CAE	Common Application Environment
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service (definition)
CMOT	Common Management information service Over TCP/IP
CMISE	Common Management Information Service Element
CRC	Cyclic Redundancy Check
DARPA	Defence Advanced Research Project Agency
DCE	Distributed Computing Environment
DF	Don't Fragment
DNS	Domain Name Service
EGP	Exterior Gateway Protocol
EUNET	European UNIX Network
FIN	Final
FTP	File Transfer Protocol
GGP	Gateway to Gateway Protocol
GOSIP	Government OSI Protocols
IAB	Internet Activities Board
ICMP	Internet Control Message Protocol
IEN	Internet Engineering Notes
IGP	Interior Gateway Protocol
IMP	Internet Message Processor
IP	Internet Protocol
LLC	Logical Link Control
LPP	Lightweight Presentation Protocol
LU6.2	Logical Unit 6.2
MIB	Management Information Base
MSL	Maximum Segment Lifetime
MTU	Maximum Transfer Unit
NCP	Network Control Processor
NCS	Network Computing System

NETBLT	NETwork Block Transfer
NFS	Network File System
NIC	Network Information Centre
NIS	Network Information Service
NLM	Network Lock Manager
OSF	Open Software Foundation
OSI	Open Systems Interconnection
POSIX	Portable Operating System Interface on UNIX
PSH	PuSH
PSN	Packet Switched Node
RARP	Reverse Address Resolution Protocol
REX	Remote Execution Service
RFC	Request For Comments
RFS	Remote File Sharing
RFT	Request For Technology
RIP	Routing Information Protocol
RJE	Remote Job Entry
RPC	Remote Procedure Call
RST	ReSeT
RTO	Resource Time Out
RTT	Round Trip Time
SLIP	Serial Line Internet Protocol
SMB	Server Message Block
SMI	Structure of Management Information
SMTP	Simple Mail Transfer Protocol
SNAP	SubNetwork Access Protocol
SNMP	Simple Network Management Protocol
SRI	Stanford Research Institute
SRTT	Smoothed Round Trip Time
SVID	System V Interface Definition
SYN	SYNchronise
TLI	Transport Layer Interface
TP4	Transport Protocol 4
TPI	Transport Provider Interface
TTL	Time To Live
UCB	University of California, Berkeley
UCLA	University of California, Los Angeles
UCSB	University of California, Santa Barbara

UDP	User Datagram Protocol
URG	URGent
UUCP	Unix to Unix Copy Program
VFS	Virtual File System
VMTP	Versatile Message Transfer Protocol
XDR	eXternal Data Representation
XNS	Xerox Network System
XPG	X/Open Portability Guide
XTI	X/Open Transport Interface
YP	Yellow Pages

DHCP Process

Client Server

DHCP DISCOVER →

DHCP OFFER ←

DHCP REQUEST →

DHCP ACK ←

DHCP IP addresses

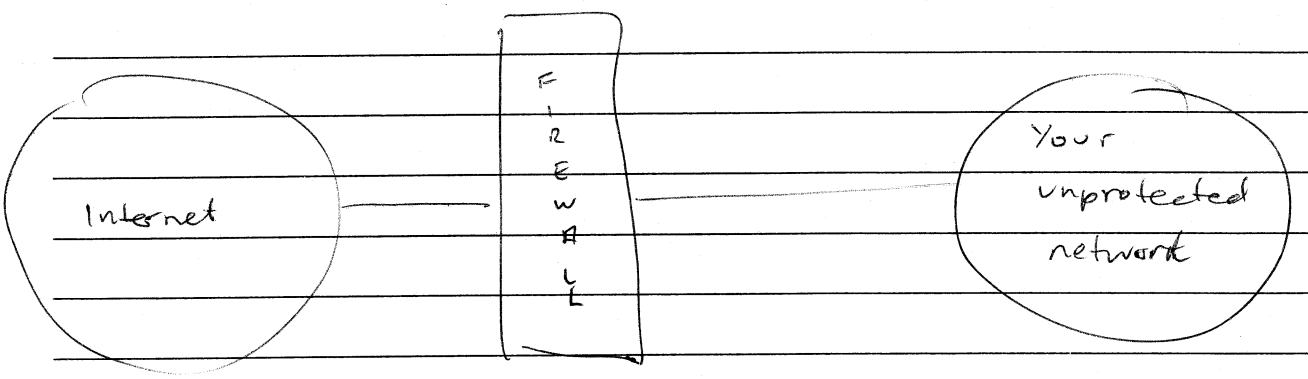
Source & Dest IP addr not yet set

Uses

Entity	Stage	Source	Dest
Client	DISCOVER	0.0.0.0	255.255.255.255
Server	OFFER	Server	✓
Client	REQ	0.0.0.0	✓
Server	ACK	Ser	✓

Firewalls

- Packet filters or screening routers
- Application level gateways or proxies



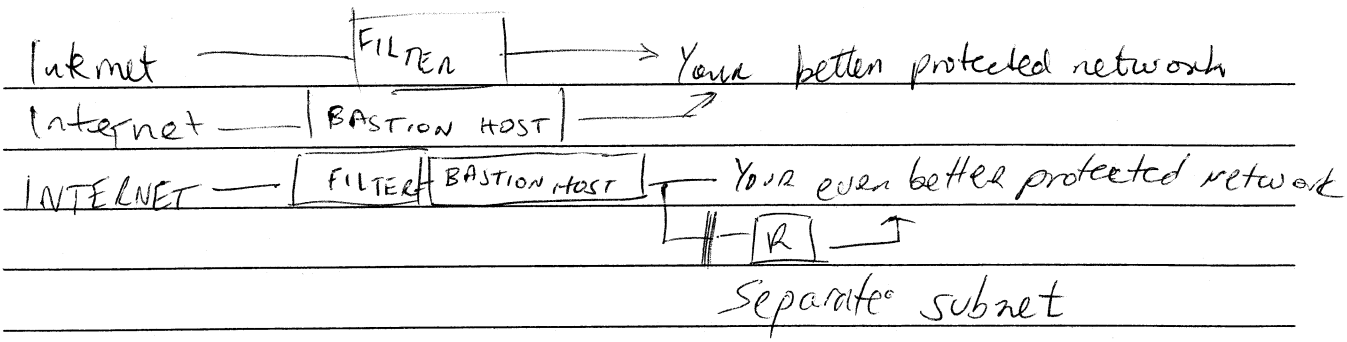
Stops outside people getting into your network
Stops inside people wasting time on Internet
Can cache data
Can prevent access to certain sights

Screening router - not very flexible - only operates at filter level

Application gateway (bastion host) only - Not very flexible - only operates at application level

Screening router & bastion host - better

Separate subnet & screening router & bastion - defence in depth

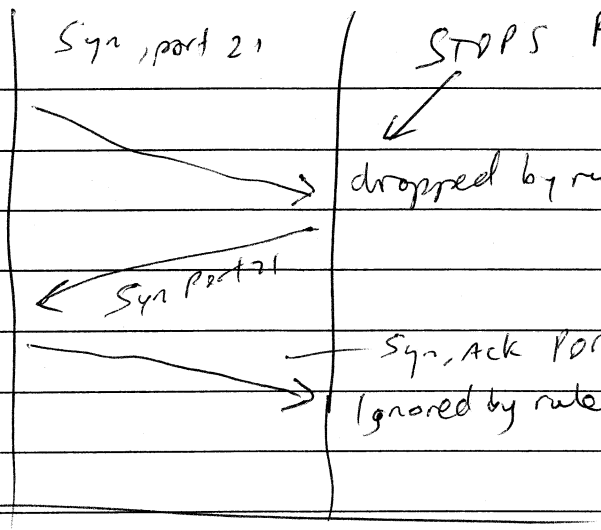


Packet filters
- IP address
- ports
- protocol
- etc

Example
- TCP coming from outside to port 21 (FTP) containing SYN but no ACK flag are dropped.

What packets should get through
By default, don't let it through
IPFW under Linux (IP Firewall)
Filter rules in router boxes
TIS FWTK (Trusted Information System Firewall toolkit)
Commercial products
Get the source if you can

www.greatcircle.com for firewall info



Bastions

Can do more make better decisions
Can use passwords or encryption
More processing required

TIS FWTK
Squid
FTP proxies
Telnet proxies
OPIE and S/Key (one-time password) ← password can only be used once
SSH (Secure Shell) for encryption & authorisation

TCP Wrappers
- controls & logs access to services launched through inetd
  Domain level or machine level only, not user level.

Split DNS

You may need DNS to query outside your network for lookup
But you dont want incoming DNS queries to see inside your network

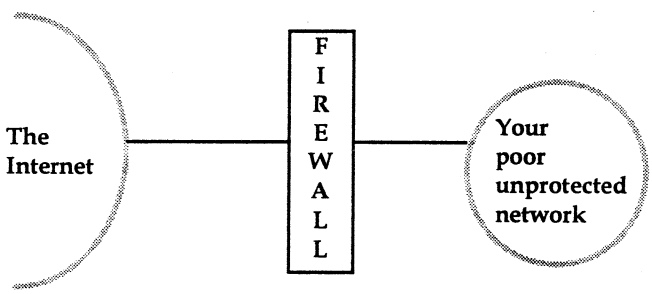
External DNS requests never passed through → DNS Server firewall

IP V4 4 octets (the usual)
IP V6 128 ~~octets~~ bits (16 bytes)

Firewalls

A firewall is ...

- **something between you and the internet**
  - **controls incoming access**
  - **controls outgoing access**



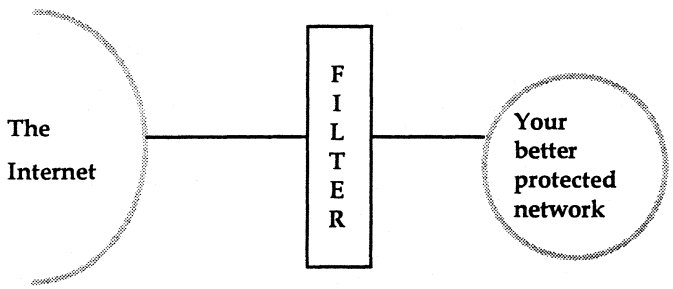
Types of firewalls

- **Packet filters or screening routers**
  - **filter packets using particular criteria**
  - **original was screend**
- **Application level gateways or proxies**
  - **Accept connections and then boost them through if all OK**
  - **squid, the HTTP proxy cache is an example**
  - **TIS FWTK contains FTP, Telnet, etc ...**

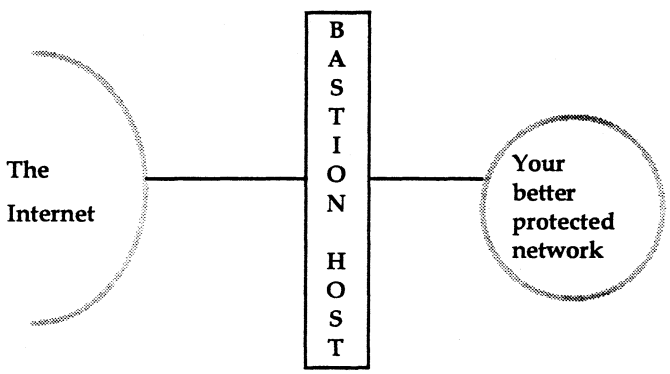
Types of firewalls (cont ...)

- **A screening router**
  - **Not very flexible**
  - **Only operates at the filter level**
- **An application gateway (bastion host) only**
  - **Again, not very flexible**
  - **Only operates at the application level**
- **A screening router and bastion host**
  - **better**
- **A separate subnet with screening router and bastion**
  - **defence in depth**

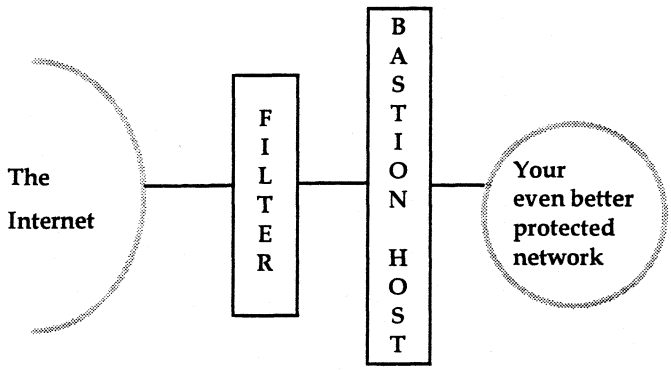
Screening router



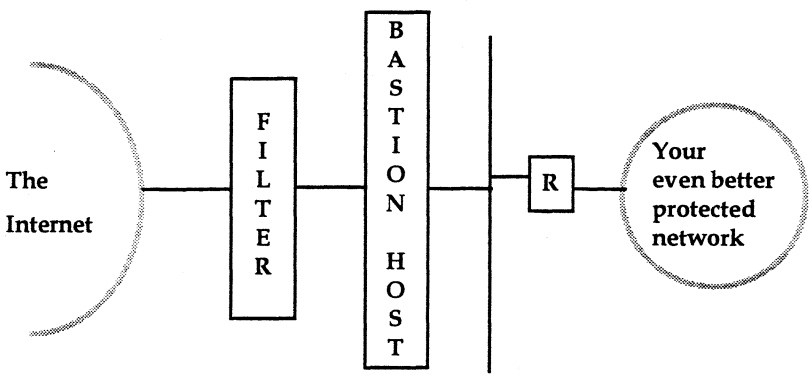
Application gateways



Screening router and bastion host



Screened subnet with bastion host

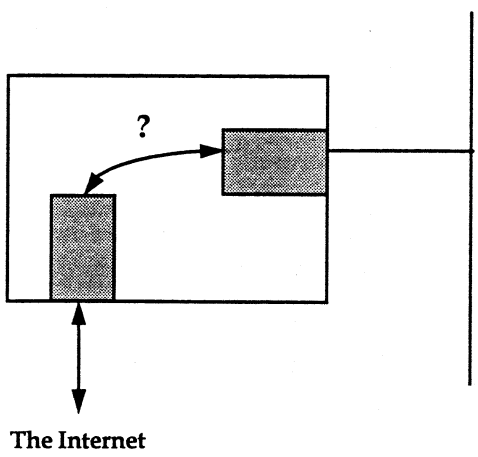


Packet filters

- Filter packets based on packet level info
  - IP addresses
  - ports
  - protocol
  - etc
- Example
  - TCP segments coming from outside, destined to port 21 (FTP) and containing a SYN flag but no ACK flag are dropped

Packet filters

- What packets should get through?
- Be paranoid, don't allow packets through by default
- IPFW under Linux
- Filter rules in router boxes
- TIS FWTK
- Commercial products?
- Get source if you can!



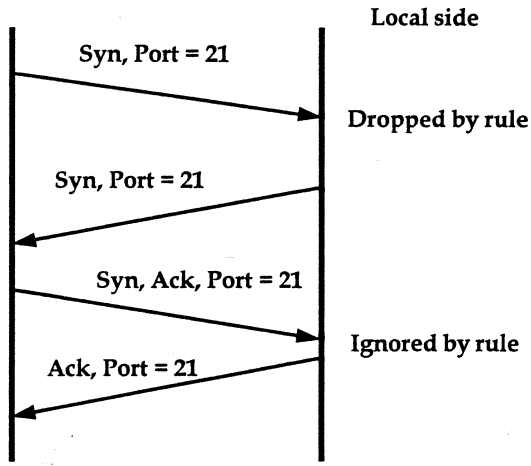
Filter example

Rule:

Don't allow incoming FTP

Filter:

TCP Segments from outside, to port 21, with a Syn but no Ack flag, are dropped



Application level gateways

- Can do more and make finer grained decisions
- Can use one time password systems or public key encryption for authentication
- More processing required

Application level gateways

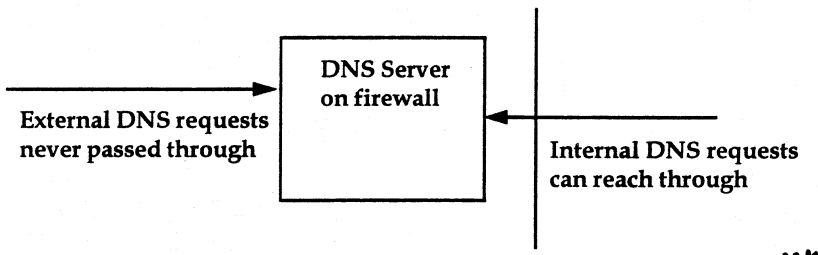
- Trusted Information Systems Firewall Toolkit
- squid
- FTP proxies
  - might ask for one time password via SNK
- Telnet proxies
  - might ask for one time password via SNK
- OPIE and S/Key (One time password systems)
- SSH for encryption and authentication

Other facilities

- TCP wrappers
  - controls and logs access to services launched through inetd

Split DNS

- You may need DNS to query outside your network for lookups
- But not want incoming DNS queries to see inside your network



1. Hands-on Session 1

1.1. Objective

The purpose of this session is to undertake the necessary hardware configuration for hands-on sessions that will be used through-out the course.

In addition to the Pentium 75's that you and your partner will be using the following components need to be identified and setup:

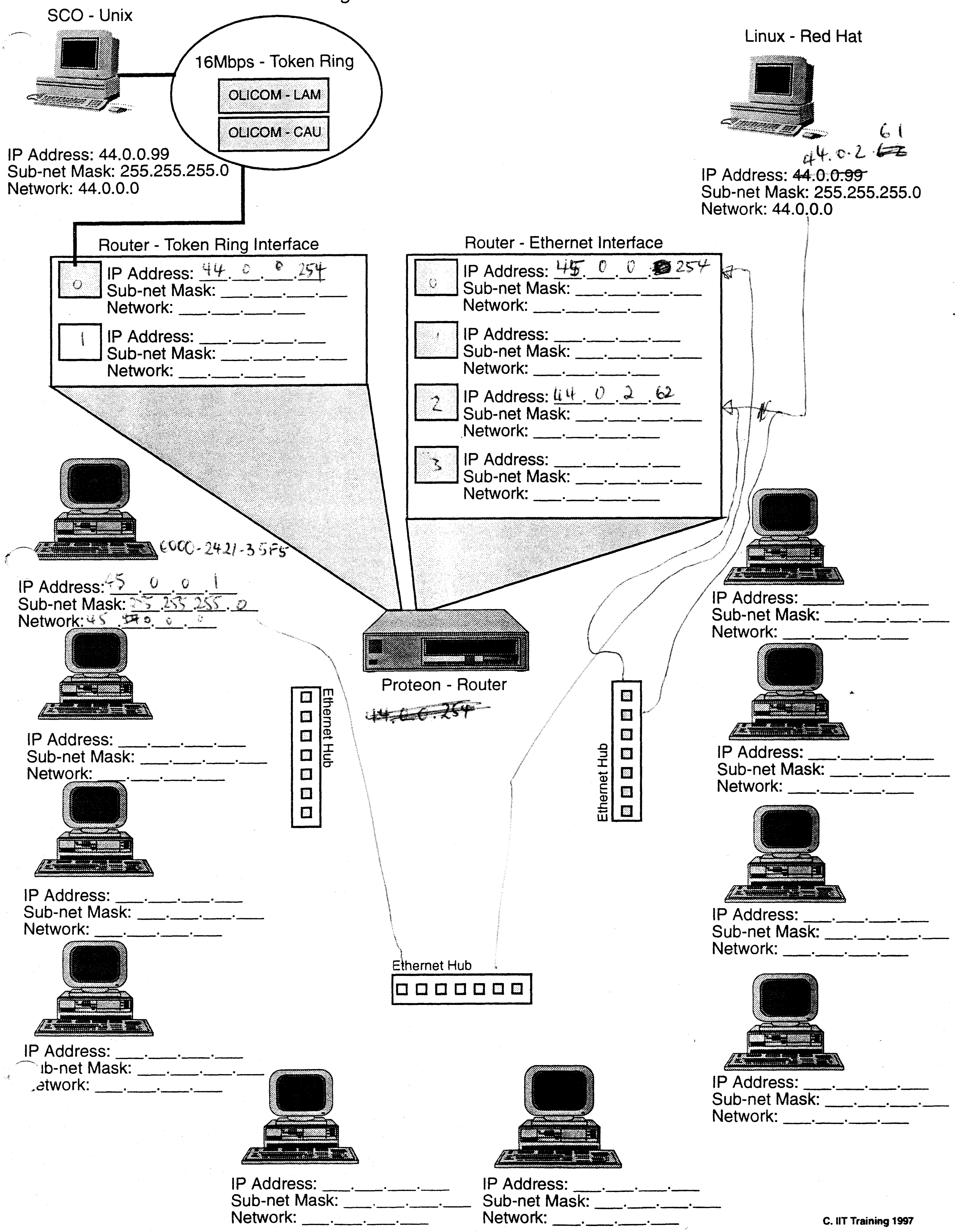
- **Proteon CNX500 router**
  Two (2) token ring and four (4) Ethernet interfaces
- **10BaseT hubs**
  Three (3) 10BaseT Cabletron hubs
- **Token Ring MSAU**
  One (1) Oilicom CAM and associated LAM
- **SCO Unix**
  One (1) SCO Unix host
- **Linux Unix**
  One (1) Linux Unix host
- **Assorted 10BaseT cables at 3m and 5m lengths**

1.2. Network Diagram

Please complete the first working digram on the attached sheet. Please note the IP addresses as presented by the instructor. Please make sure that you make careful note of the IP addresses and sub-net masks for all the interfaces.

IIT - Hands-on TCP/IP Networks - Working Network Diagram

Diagram No: Date:



C. IIT Training 1997

2. Hands-on Session 2

2.1. Objective

The purpose of this session is to configure and load a DOS based TCP/IP protocol stack. At the conclusion of this session you will have configured your workstation with an IP address and tested the protocol implementation with simple TCP/IP utilities.

The PCs that you are using are Pentium 75's with 16Mb RAM. They are configured with DOS v6.22 and Windows v3.11. The network adaptor card is an Olicom 10BaseT adaptor which we will be using in an NE2000 mode. The I/O address of the card is x300 and the interrupt is set to 10.

2.2. Hands-on Steps

1. Power-on your local PC and go to the C:\PCTCP sub-directory. The version of the TCP/IP software is PCTCP, which is a protocol stack from FTP Software Corp.

2. From this subdirectory execute the program CONFEDIT.EXE. This program allows you to edit the TCP/IP configuration file. This configuration information is stored in a text file PCTCP.INI. The standard contents are detailed below. You will need to make some changes to the fields, and the fields that you need to change are tabled here. These fields are also flagged, with an asterisk, in the detailed contents so that you can readily identify them whilst you are editing the file.

   Ask you instructor now for the values of the following fields: lower-case

   ** host-name = PC1

   ** domain = iit.com

   ** user = train1

   ** broadcast-address = 45.0.0.255

   ** ip-address = 45.0.0.1

   ** subnet-mask = 255.255.255.0

   ** router = 45.0.0.254

2.3. Sample PCTCP Configuration File

(pctcp general)
*** host-name = Train1*
*** domain = IIT.COM*
*** user = PC??*
time-zone = EST
time-zone-offset = 300
use-old-init-scheme = no

(pctcp kernel)
interface = ifcust 0
window = 2048
low-window = 0
use-emm = no
kernel-int = 0x61
ip-ttl = 64
ip-precedence = routine
ip-precedence-matching = lax
ip-security = none
ip-delay = high
ip-reliability = low
ip-throughput = low
large-packets = 5
small-packets = 5
tcp-connections = 6
udp-connections = 6
router-discovery = no
mtu-discovery = yes
host-table = c:\pctcp\hosts.txt

(pctcp ifcust 0)
*** broadcast-address = 255.255.255.255*
*** ip-address = 46.0.0.3*
*** subnet-mask = 255.0.0.0*
*** router = 46.0.0.98*

(pctcp lpr)
banner = yes

(pctcp idprint printer)
when = timeout
timer = 30
mine = no
number = 0

(pctcp ip-security)
basic-classification = unclassified
matching = lax

(pctcp pd-pctcp)
default = ipx

(pctcp pd-pctcp ipx)
retain-header = 0
protocol = UDP
type-port = 0x8137,213

(pctcp netbios)
scope =

(pctcp tn)
ftpsrv = off
status = on
back-arrow-key = del
kerberos4-auth = on
kerberos4-only = off

(pctcp vt)
allow-vt220-8-bit = off
8-bit = off
wrap-line = on
answerback = FTP

(pctcp host)
umask = 644

(pctcp wmsg)
protocol = UDP
save-protocol = yes
beep-on-msg = yes

(pctcp serial n)
port = 1
io-addr = 0x03f8
irq = 4
baud = 2400
hardware-flow-control = on

(pctcp comscrpt hostname)
dialup = c:\pctcp\dialup.scr
hangup = c:\pctcp\hangup.scr
mru = 1500
accm = 0x000A0000
addr-ctrl-field-comp = on
prot-field-comp = on
local-ip-address = 0.0.0.0
remote-ip-address = 0.0.0.0

(pctcp time)
dst-begins = 94
dst-ends = 299

(pctcp install)
bindir = C:\PCTCP
etcdir = C:\PCTCP
kernel-name = ETHDRV.EXE
major-version = 2
minor-version = 2
patch-level = 0
do-kappconf = yes
configure-driver = yes

3. Once you have made the changes to the PCTCP.INI file you must save the changes by using the F5 key and then press ESC to exit the CONFEDIT utility. You can now test the protocol stack.

4. This implementation of PCTCP uses the Novell Open Data Link Interface (ODLI or ODI for short). To load the protocol stack we need to first set up ODI environment and then load the MAC layer NIC driver. This is achieved by executing the following programs:

   LSL

   NE2000

   When the program NE2000 is executed it also refers to a file called NET.CFG for configuration information. This configuration information includes the frame format that will be used by the adaptor. We have configured the MAC driver to use the standard Ethernet_II frame type format. ie. Your IP packets will be sent in Ethernet v2.0 frame format.

5. Next we load the two key pieces of software the packet driver and the TCP/IP protocol. The packet driver provides the connection to your NIC in terms of software interface for IP. Load the packet driver by executing the following program:

   ODIPKT

What is the MAC address of your NIC that is being reported by the packet driver? Record your Ethernet MAC address here: 0000 · 2421-351F

6. Next load the TCP/IP kernel software. This will go looking for the packet driver and report back a number of pieces of configuration information. Load the kernel by executing the following program:

   ETHDRV

7. You can now test the installation by using by using the PING program which utilises the Internet Control Message Protocol (ICMP) echo functions. The purpose of the PING program is to test the reachability of other hosts in the IP network. Perform the following tasks and observe the results.

a) Issue the **ping -?** command to review the options available with the ping command. Note: The options available with the **ping** command vary between different implementations of the IP protocol.

b) What is the format of the command that you would use to send 100 pings to another host?

ping -n 100 ~~10~~ ~~45~~. 0. 0. 254

c) Why would sending more than one ping to a host be of use?

Test reliability of a route

d) What is the format of the command that you would use to increase the wait time to 20 seconds?

Ping -w 20 45. 0. 0. 254

e) Which command would you use to trace the IP route to a target host?

ping -r 45. 0. 0. 254

f) Ping the SCO Unix host at address 44.0.0.99. What was the host response time? 25 ms

g) Record the value of:

arps transmitted 3

arps received 7

h) What is the meaning of the **arps transmitted** and **arps received** values?

Number of address requests

Number of arp's detected on the segment

i) Issue a single ping to 44.0.0.99, however this time use the trace route option.

What is the IP address of the first hop? 44.0.0.254

Whose address is this? Router - Token ring interface

Where have you seen this address before? _____

What is the IP address of the last hop in the PING's route? 45.0.0.254

Under what circumstances is an IP address added to the route ?

If an alternate path was used or another router added into the wiring

j) Issue a single ping to 44.0.2.61, with the trace route option. What is different about this?

The PC is shown twice

8. We can now also take a look at the statistics and other IP information being recorded by the TCP/IP stack. We can investigate these statistics by using the **INET** command. Note: Different protocol stack implementations use different commands to get this information.

a) Issue the **inet** command and review the available options.

b) What are the four key protocols that the **inet stats** command keeps track of?

TCP

IP

UDP

ICMP

c) If you have only been using the PING command ie. ICMP echo requests and responses to this juncture, why is there no protocol packet send/receive statistics for UDP and TCP?

ICMP does not use UDP/TCP, not part of transport layer

d) Issue the **inet arp** command. What information does this command provide and of what use is this information?

Local subnet only (not via a router)

Shows IP addresses, MAC addresses, and time-out status

3. Hands-on Session 3

3.1. Objective

The purpose of this session is to test and use the basic TCP/IP protocols by using an application to communicate with another host in the network. The application we are going to use is the File Transfer Protocol (FTP) which uses both TCP and IP to transfer files between hosts.

FTP will be used to copy a protocol analyser utility called LAN Decoder (LDE) from the SCO Unix server to your PC and to configure it for use. At the conclusion of this session, you will have used FTP to copy files from one machine to another via the network, and will have briefly monitored the network.

LDE is a network packet monitoring and decoding package from Triticom. Similar packaged are available from Microsoft (NetMon), Novell (Lanalyzer), etc.

3.2. Hands-on Steps

1. Create the directory C:\LDE on your local PC, and then change to this directory.
2. FTP is an executable program that resides in the C:\PCTCP directory. Execute the FTP program and connect to 44.0.0.99 (the SCO UNIX server) by typing from the C:\LDE prompt:

   c:\pctcp\ftp 44.0.0.99

3. FTP will have now been initialised and your prompt should now be **FTP 44.0.0.99**. You will also be challenged to enter a userid. Enter **lanscope** as the userid and press, followed by **lanscope** as the password.
4. You are now in the lanscope account on the SCO UNIX machine.

a) Issue a print working directory command (PWD) to see your current path. Note the path should read /usr/lanscope:

   pwd

b) Get a directory listing by typing:

   dir
   or use the Unix derivative
   ls -l

5. Now go to the **/usr/lanscope/prac2** directory on the SCO UNIX server by entering:

   cd prac2

Issue the **pwd** command to confirm the current directory, and then do an **dir** to review the contents of the prac2 directory. Your should find four files in the prac2 directory:

lde.exe
names.~~dat~~ lan
lde.cfg
read.me
ethrvend.id

6. FTP allows the files to be brought back in binary (image) mode or ascii (text) mode. Enter the **binary** command to bring back all the files in binary mode.

   binary

   Note that you should have received the reply 'Type set to I' in response to this command.

7. Tell FTP not to prompt you to copy each file with the command:

   option ask off

8. Copy all the files with:

   mget *.*

   Note: MGET is 'multiple get' and the full stop asks for all files with or without extensions. This command will copy all the files without you having to enter their names separately.

9. The **lde.cfg** file is however a text file and needs to be brought back in ascii mode. Issue the following command to change FTP's transfer method to ascii:

   ascii

   You should receive a prompt 'Type set to ascii' in response to this command.

10. Issue the following command to bring back the lde.cfg file:

    get lde.cfg

11. Before finishing with FTP, please answer the following question:

a) What did FTP tell you about each file that it copied from the SCO UNIX server for you?

Transferred xxx bytes in x seconds (x bits/sec, x bytes/sec).

12. Now exit from FTP with the quit command. Note: You might find that FTP has timed you out if there has not been activity on the connection.
13. Reboot the PC to remove all the network drivers and protocols from memory.
14. From your local PC check the contents of the C:\LDE directory and ensure that the ~~four~~ five files that were on the SCO Unix server are now present.
15. Run LAN Decoder by executing the LDE command from the LDE directory. From the LDE main menu arrow select **'Capture Traffic'** which will start monitoring of the local segment. The capture traffic process will display the source and destination MAC addresses of the packets being caught. You will note that the function key options at the bottom of the capture screen allows you to change items such as the address being seen ie: IP, full MAC, or interpreted vendor code MAC address etc. and other options such as frame counts or byte counts.
16. Wait until you have at least captured one broadcast frame, identified by the MAC address FFFFFFFFFFFF in the destination field, and press ESC to stop the capture. After you have captured a few frames, exit from capture mode by entering an ESC, and then go to decode mode to review the details of the frames. Find the broadcast frame in the capture buffer and review the IP header details.

4. Hands-on Session 4

4.1. Objective

The purpose of this session is to use LDE to monitor network activity and to discover what packets are actually transmitted when you 'ping' another TCP/IP host.

You will operate in teams of two groups, each with a PC, where one group will monitor the network, while the other group performs the PING request. After the monitoring group has answered all the questions below, swap around so that each group has an opportunity to monitor the traffic.

4.2. Hands-on Steps

4.2.1 Monitoring group:

1. Go to the directory on your local PC, C:\LDE. Start LDE typing:

   C:\LDE\LDE ___

2. Commence capturing packets and ask your 'PINGing' group to PING the SCO Unix host and other active PC's, as noted in there script below.
3. When the other group has finished PINGing, stop capturing, go to decode mode, and answer the questions below.

4.2.2 PINGing group:

1. Reboot your PC and bring up the PCTCP TCP/IP stack
2. Ping 44.0.0.99 with the command:

   ping 44.0.0.99

3. Issue the following PING also

   ping -r 44.0.0.99

4. Ping the routers local Ethernet segment interface with

   ping 45.0.0.254

4.3. Questions

1. Find the ICMP echo request and response commands in the stream of packets captured. You might see several, since several teams will be issuing a ping command at once. Try to find your teams PING request by checking either the MAC or IP address of the PC for the group in your team that performed the PING.

   What data does the packet carry? Sample data 256 bytes

   How long is the data? 256

What is the maximum length of the ICMP Echo packet? 1518

2. Find the response to the ICMP Echo packet. It should be after the request. Use the ethernet MAC address to ensure that it is the one for your team.

a) Does the data differ in any way to what was in the request?

No

b) What is the value of the TTL field in the ICMP response?

254

c) Where is this TTL value set?

Source

d) Go back to check the TTL value of the ICMP request?

64

e) Why would having a long or short value of the TTL be of benefit?

Short one cuts network looping but increases chances of errors

3. Go backwards in the packet stream and find an ARP request and ARP response for 45.0.0.254. There must be one, since the PC that did the PING was just rebooted, and its ARP cache would be empty.

a) Where did the response to the ARP request come from (check the ethernet MAC address)

Router

b) What MAC address did the ARP response claim 45.0.0.254 is at?

0000 - 9318 - 881D

c) Why does the PC need to know the MAC address of 45.0.0.254?

So it can communicate directly with the router and not have to do continuous broadcasts.

5. Hands-on Session 5

5.1. Objective

The purpose of this hands-on session is to review and familiarise you with the structure and format of the TCP and UDP headers. After this review you will be comfortable in describing and identifying the contents of the headers.

1. Working in your teams organise a trace capture of an FTP session followed immediately by a TFTP session. By using both applications you will be able to compare in a single trace file the use TCP (FTP) and UDP (TFTP) headers.

   Using the SCO Unix box as in hands-on session #3, retrieve first by FTP and then by TFTP the text file **rfc854.txt** /usr/lanscope/prac5 directory. Start with FTP and transfer the file, **quit** from FTP and then immediately start TFTP. TFTP is located in the C:\PCTCP directory on your local machine.

2. Once you have captured the frames for both FTP and TFTP answer the following questions:

a) What source and destination ports were used in:

i) FTP session PC->SCO 3615 SCO<-PC 21

ii) TFTP session PC->SCO 1010 SCO<-PC ~~20~~ 69

b) Can you identify any difference in the packet size used in FTP and TFTP in the trace? If so what?

512 TFTP

512 FTP

c) How many packets were sent between the SCO and PC hosts used in the session with FTP and TFTP?

20 77

3. Go back to the TCP session trace and find the three way handshake that initiated the FTP control session. say FTP(no data)

a) Which flags were set in the first packet in the exchange?

SYNC

b) What was the requested Initial Sequence Number (ISN)?

c) Which flags were set in the second packet of the exchange?

SYN ACK

d) What is the sequence number of the first byte of data sent, and how does it relate to the ISN?

e) What was the window offered by the server?

4096

4. You are provided with the summary of a IP packet and the hexadecimal interpretation of the entire packet. With this information answer the following questions:

5.2. Summary information

Use LAN ~~analyzer~~ decoder.

Source	Destination
191.254.2.5	44.0.0.99
Frame length	82 bytes
CRC	89A3C5B3

Data Link Control says it's a TCP/IP packet

Version

Header Length

Type of Service

Hexadecimal data																
Offset	**Hex data**												DLC Control	Header		
0000	00	00	93	18	88	1D	00	00	24	21	30	1E	08	00	45	10
0010	00	40	00	B7	00	00	40	06	8B	8B	BF	FE	02	05	2C	00
0020	00	63	0B	31	00	15	00	A0	8F	21	4D	04	CC	75	50	18
0030	07	8D	AF	D4	00	00	50	4F	52	54	20	31	39	31	2C	32
0040	35	34	2C	32	2C	35	2C	31	31	2C	35	30	0D	0A	89	A3
0050	C5	B3														

Message 18 bytes

a) What is the value of the IP fragment ID?

00 B7

b) What is the value of the TTL?

40

c) What is the value of the IP field that identifies this as a TCP packet?

06

d) What are the source and destination ports for the FTP information?

0B31 / 0015

e) Identify the TCP flags field?

18

f) Identify the TCP sequence number field?

00 A0 8F 21

g) Identify the TCP acknowledgement number?

40 04 CC 75

Sequence Numbers

SYN flag counts as 1

Actual data in subsequent packets add to sequence numbers

Control data in packets not included

FIN flag counts as 1 more

So to send 6132 bytes
Sequence is 1 + 6132 + 1.

Other end uses acknowledgement number to ack receipt of data

As transmission could be duplex, sequence & ack numbers are swapped about for each direction.

6. Hands-on Session 6

6.1. Objective

The purpose of this hands-on session is reinforce the understanding of the routing protocol RIP. Using a protocol analyser the RIP packets on the network will be investigated. RIP packet is a broadcast ie all FFFF

6.2. Hands-on Steps

1. Start LDE and capture several RIP packets. After you have several broadcasts from the router, you should have several RIP packets.

2. What is the time difference between them?

   Approx 30 seconds

3. What networks are the RIP packets announcing and what are the distances to them?

   44.0.0.0 Distance 1

   46.0.0.0 Distance 1

4. What is the TTL of each RIP packet?

   60

5. What is the source MAC address of each packet?

   45.0.0.254 0000.9318.881D

7. Hands-on Session #7

7.1. Objective

The purpose of this hands-on session is to explore further the use of the default route configuration options and the use of trace route to determine IP packet paths.

The PCTCP.INI has been modified to remove the default gateway. This will affect the operation of the stack.

7.2. Hands-on Steps

1. Ping 45.0.0.254. Is it reachable?

   YES

2. Ping 44.0.0.99. Is it reachable?

   YES

3. Examine the pctcp.ini. What is causing the problem?

   IP_ROUTER was missing

4. Now telnet to the Linux system (44.0.2.61) and log in as pcn when prompted (where n is the number of your pc). Your password is the same as your user account. TN

5. Get super-user privilege with the command:

   su -

   and enter the password 'batcavin' when prompted.

6. Ping 44.0.0.99. If it does not respond for a while, type Control-C. Have any packets been dropped?

7. Go to the SCO console and enter the following command:

   netstat -r

   Can you see what the problem is?

8. After the instructor adds a default route to the SCO system, try to ping 44.0.0.99 again from Linux. Does it work now?

9. What exactly was the problem in 6 above?

10. At what point were packets being dropped?

a) On their way to the router from Linux?

b) On their way from the router to SCO?

c) On their way from SCO to the router?

d) On their way from the router to Linux?

8. Hands-on Session 8

8.1. Objective

The purpose of this hands-on session is to explore the Telnet application as it interoperates over the TCP/IP protocol stack. Specifics of the application's operation will be investigated.

However, we will first install the Windows for Workgroups 32-bit TCP/IP stack from Microsoft (TCP32B).

8.2. Hands-on Steps

1. Reboot the PC and Start WFW
2. Open the Network Group and start Network Setup
3. Select "Install Microsoft Windows Network" with "No Additional Network" selected. Click OK
4. From Network Drivers click on Add Adaptor and select "NE2000 Compatible"
5. Set interrupt to 10
6. Click on Add Protocol and select "Unlisted or Updated Protocol"
7. Click Browse, select D:\TCP32W and answer OK
8. When prompted for Disk 7, Click Browse and select D:\WINDOWS\DISK_7
9. When prompted for Disk 8, Click Browse and select D:\WINDOWS\DISK_8

   When presented with a TCP/IP configuration dialogue box, enter the appropriate details as per the PCTCP configuration, eg:

   IP Address subnet
   ~~Broadcast~~ address
   Gateway address

10. Restart the PC when prompted
11. Working in your team, capture a login sequence to the Linux host. One group should login to 44.0.2.61 via Telnet from Windows for Workgroups, while the other group should monitor the frames using LDE.

    After logging in, obtain a directory listing with the command:

    ls -al /home/public

Userid PC1
Password PC1

12. Record the following information for each of the frames that relate to the session:

Direction (tick)		TCP Seq No	TCP Ack#	Window Size	Summary description of Telnet data
PC->Lnx	Lnx->PC				
✓		67418	-1456333341	31744	

13. Record all the IAC commands listed at the beginning of the session and determine their meaning from the RFC loaded in hands-on session #5 or your class notes.

9. Hands-on Session 9

9.1. Objective

The purpose of this hands-on session is to explore the use of a Domain Name Server. Using DNS we can now move beyond the use of identifying hosts and services via IP addresses and move towards identification by name.

Initially we will need to configure your PC to use DNS.

9.2. Hands-on Steps

Network TCP/IP DNS ↓ Host name = pc1 Domain Name = iit.com

LINUX1

1. Configure you local PC to use DNS. Follow the instructors guidelines on this as options and configuration vary between the protocol stacks that you now have configured. The DNS server will be the Linux host at 44.0.2.61.
2. In your teams capture a Telnet session to ~~linix1iit.com~~ linux1.iit.com
3. What protocol does DNS seem to use?

   UDP

4. How long does it take to timeout/return if you use a name that is not in DNS? Why is this?

   1.5 ms

5. Telnet to **linux1**, login with the userid and password assigned by the instructor and look at the following files:

   (1) /etc/named.boot
   (2) /~~etc~~/local/adm/named/iit.zone
   USR

   What is the purpose of these files?

   (1) files for each domain being serviced
   (2) translating from name to IP-address

~~2423~~ 2423: 04FA

10. Hands-on Session 10

10.1. Objective

The purpose of this hands-on session is to observe the operation of the Simple Mail Transfer Protocol (SMTP) in operation.

Initially we will need to configure your PC to use the SMTP components.

10.2. Hands-on Steps

1. Configure your local PC to use SMTP. Follow the instructors guidelines on this as options and configuration vary between the protocol stacks that you now have configured.
2. In your teams capture and monitor an SMTP session. One team should monitor with LDE and the other compose and send mail.
3. What port does SMTP use?

   25

4. What type of commands does SMTP use?

   MAIL TO

   RECEIVE FROM

5. How does the mail program terminate the body of the message?

   QUIT

6. Is there any checking of who the sender really is?

   No

11. Hands-on Session 11

11.1. Objective

The purpose of this hands-on session is to observe the operation of a World Wide Web (WWW) browser session.

Initially we will need to configure your PC to setup the Hyper Text Transfer Protocol (HTTP) support on your local machines.

11.2. Hands-on Steps

1. Configure your local PC to use Information Explorer 2.1. Follow your instructor guidelines.
2. In your teams capture and monitor with LDE an HTTP connection from one PC to **//44.0.2.61/**.
3. How many connections were established between the PC and the Web server?

   1063 -> 80

   1064

   1065

4. Who terminates each of the connections?
5. What command does the browser use to fetch a page?

   GET
6. What is the first part of each response from the server?

12. Hands-on Session 12

12.1. Objective

The purpose of this hands-on session is to configure your local machines for dynamic assignment of IP addresses using the Dynamic Host Configuration Protocol (DHCP).

Initially we will need to configure your PC to use DHCP.

12.2. Hands-on Steps

1. Configure your local PC Windows TCP/IP stack to use DHCP. The instructor guidelines will take you through making the changes in the Windows Network Setup dialogues. After you have made the necessary modifications restart Windows.

2. What IP address does your local machine now have after Windows has been restarted? (Use **ipconfig** to display the information)

   IPCONFIG/ALL /RENEW

   45.0.0.25

3. The instructor will change the DHCP server configuration to provide a DNS Server address and a WINS Server address. Restart Windows and verify that these are now provided. Record your details below.

   44.0.2.61 DNS

   44.0.2.61 WINS

4. Using LDE determine what packets were exchanged to get the DHCP information into your PC's configuration. Record a summary of these packets below.

Copyright IIT Training Pty Ltd