Network Protocols

A study of various network protocols

Pratap Vikram Singh

July 4th, 1997

Contents

1. Transmission Control Protocol / Internet Protocol

1.1 Introduction

The TCP/IP protocol was introduced by the ARPA in US for communication between research organizations and universities. It grew very popular and became one of the de-facto standards of the networking industry. Most of the Unix systems incorporate this protocol. BSD 4.3 introduced it first followed by other Unix systems. The Unix OS which supports this protocol implements the Transport layer and below completely in the kernel. The application programmer uses the interfaces provided to him to use the TCP/IP model.

This protocol suite not only consists of TCP and IP, but other members also like *User Datagram Protocol* (UDP), *Internet Control Message Protocol* (ICMP), *Address Resolution Protocol* (ARP) and the *Reverse Address Resolution Protocol* (RARP).

Transmission Control Protocol is a connection-oriented protocol. It provides a reliable, full-duplex byte stream for the user process. Since TCP uses IP, the entire suite is called TCP/IP. It is in the transport layer.

User Datagram Protocol is a connectionless protocol for user processes. Unlike TCP, which is a reliable protocol, there is no guarantee that UDP datagrams ever reach their intended destinations. It is in the transport layer.

Internet Control Message Protocol is a protocol, which handles error and control information between Gateways and Hosts. It is used by the TCP/IP network software and not directly by the user process. It is in the transport layer.

The *Internet Protocol* provides the packet delivery for the TCP, UDP and the ICMP. User processes may not be directly concerned with this layer. It is in the network layer.

Address Resolution Protocol is used to map the IP addresses of a host to its local network address. E.g. host can be on an Ethernet and the gateway uses ARP to map the IP address of the host to its local Ethernet address.

Reverse Address Resolution Protocol maps the hardware address of a host to its IP address. It is the opposite of the ARP.

There is a variety of Data Link connections used by the TCP/IP suite like satellite links and packet radio.

1.2 Network Layer

Each IP datagram is considered as an independent packet. There is no association provided between the packets and has to be done by the upper layers. Each packet contains the source and destination address so each datagram can be routed and delivered independently. It does not guarantee the correct delivery of each packet and must be handled by the upper layers.

The IP layer verifies the checksum and encapsulates the data by a 20-byte header. If an error is found in the packet by the IP layer, it simply discards it and the recovery must be maintained by the upper layers.

The functions of the IP layer are :

1. Error checking.
2. Routing.
3. Fragmentation.
4. Flow control.

1.3 Internet Addresses

An Internet address occupies 32 bits and encodes both the network id and the host id. Every host on a TCP/IP Internet must have a unique 32-bit address and is assigned by an authority called *Network Information Center* (NIC).

We generally write the IP address as four numbers separated by dots, e.g. 164.164.30.22. These four numbers are represented by an unsigned character (8 bits) thus making a total of 32 bits. When the IP address is decoded into its binary representation, it has one of the four formats :

CLASS A

0	7 bits for network id	24 bits for host id

This class of addressing is used in networks where there is a large number of hosts.

CLASS B

1	0	14 bits for network id	16 bits for host id

This class of addressing is used for an intermediate mix of networks and host in the network.

CLASS C

1	1	0	21 bits for network id	8 bits for host id

For networks having more network and less hosts per network.

CLASS D

1	1	1	0	28 bits for multicast address

The NIC takes care of giving a unique network id in cases of classes A to C. The requesting organization has the responsibility of providing the host ids.

Gateways generally extract the network id of the IP address and use it for routing purposes. Thus the gateways need only know the location of the networks and not the location of each host. This forms a distributed form of addressing. An organization with an Internet address of any class can subdivide the host address part in to more classes. One very common one is the subnet address.

1	0	14 bits for network id	16 bits for host id

1	0	14 bits for network id	8 bits for subnet id	8 bits for host id

1.4 Address Resolution Protocol

In an Ethernet Local area network (LAN) consisting of hosts using the TCP/IP protocol, we have two types of addresses. One is the 32-bit IP address of the host on the Internet and the other is the 48 bit local Ethernet address.

When a gateway gets a packet for a host with an IP address which is also in the Ethernet LAN it has to route it using the Ethernet address and not the IP address. Thus it broadcasts a special packet on the Ethernet which asks a specified host with the IP addres to respond. This specified host is the one to whom the packet has to be routed. When this host responds, the gateway notes down its Ethernet address and uses it to route any more incoming packets to the same IP address.

Reverse Address Resolution Protocol (RARP) is implemented in another way. It is meant for diskless workstations that cannot remember their IP addresses. Thus in a LAN there is a RARP server which keeps the mapping of the local address with the IP address of each host. Thus when a host comes up, it simply asks the RARP server for its IP address. The RARP server recognizes the requesting host by its local address and gives it its IP address.

1.5 Transport Layer

Users interact with the TCP/IP protocol by sending or receiving TCP or UDP data. Since the IP layer provides unreliable connection less protocol, it is the responsibility of the TCP module to contain the required logic to provide a reliable virtual circuit for the user process.

The functions of the TCP module are :

1. Establishing and terminating connections.
2. Sequencing of data.
3. End to end reliability (checksums, time-outs and acknowledgments).
4. Flow control.

1.6 Port Numbers

There can be more than one process running on one host. To identify each process in the host, TCP and UDP both use 16 bit port numbers for each process. Therefore if a client wants to contact a host and a server process running on the host, it contacts the host by using the IP address and the server process by specifying the 16 bit port number assigned to it.

Some servers like the File transfer protocol (FTP) server always use a particular port in a host, e.g.: port 21. These ports are known as *well known ports*. When a server gets a request from the client, it has to reply to it. It gets the 32-bit IP address of the client host from the IP header. Also from the TCP header, it extracts the port to which the client has bound itself to the client host. The combination of the IP address and the 16 bit port address completes the association.

When TCP and UDP assign unique port numbers for the user processes, they are called *ephemeral port numbers* (short-lived). Port numbers from 1 through 255 are reserved ports. They are used as *well known ports* by kernel programs. Some operating systems reserve additional ports for privileged programs, e.g. 4.3BSD reserves ports 1-1023 for superuser programs. The ephemeral ports are above the reserved ports.

1.7 Addressing

The IP datagram contains the source and destination addresses in its IP header. These two 32 bit values uniquely identify two host systems that are communicating. Also contained in the IP header is a protocol identifier. This is for the IP module to determine that a datagram is for TCP, UDP or for some other protocol module that uses IP like ICMP.

The UDP and the TCP header contain the source port number and the destination port number. These two 16 bit values are used by the protocol modules to identify the user process. Note that port 1000 for UDP is independent from port 1000 for TCP because they use different protocols.

The 5-tuple that identifies an association in the Internet suite is :

1. The protocol (TCP or UDP).
2. The local host's IP address.
3. The local port number.
4. The foreign host's IP address.
5. The foreign port number.

1.8 Other features of TCP/IP protocol

UDP does not support buffering and out of band data as sequencing is not done. TCP provides both these features. It supports out of band data by classifying a data as *urgent* data. Not all implementations support more than one byte of urgent data.

If the UDP data plus the header exceeds the Message transfer unit (MTU) then fragmentation occurs. In TCP, both the end points can decide on a data size called a *segment*. This segment is so decided as to avoid fragmentation.

2. NetBIOS

2.1 Introduction

IBM introduced NetBIOS for providing interfaces for its LAN adapters. It is currently implemented as a set of device drivers. It is not a protocol but a software interface. It was designed for a group of personal computers, which share a common broadcast medium. It provides both connection-oriented and connection-less service and supports multicast and broadcast.

The four types of services provided by NetBIOS are:

1. Name service.
2. Session service.
3. Datagram service.
4. General commands.

In many PC environments, the application NetBIOS is being used for is file sharing. In this case, another protocol interface exists above it called *Server Message Block Protocol* (SMB).

2.2 Name Service

Names are used to identify resources in NetBIOS. The name space is flat, meaning there is no hierarchy, and each name consists of from 1 to 16 alphanumeric characters. Uppercase is different from lowercase and the first three letters should not be IBM.

There are two types of names :

Unique names	Must be unique across the network.
Group names	Need not be unique, all processes that have a common group name belong to the same group.

There are four commands pertaining to name service:

COMMAND	DESCRIPTION
ADD_NAME	Add a unique name
ADD_GROUP_NAME	Add a group name
DELETE_NAME	Delete a name
FIND_NAME	Determine if a name is registered

To obtain a unique name or a group name, a process must bid for the use of the name. This is done by broadcasting a notice that the process wants to use the name as a *unique name* or a *group name*. If no objections are received by any other NetBIOS node, the name is automatically registered.

Each NetBIOS node maintains a table of all names that processes on that node currently own. These names are owned by the processes till they are specifically deleted or till the node is powered off to reset. Both ADD_NAME and ADD_GROUP_NAME return an integer identifying the name.

2.3 Session Service

The NetBIOS session service provides a connection-oriented, reliable, full-duplex message service to a user process. Data is organized into messages and each message can be between 0 and 131,071 bytes. NetBIOS does not provide any form of out-of-band data. The following commands provide session service :

COMMAND	DESCRIPTION
CALL	Call - Active open
LISTEN	Listen - Passive open
SEND	Send session data
SEND_NO_ACK	Send session data, no acknowledgment
RECEIVE	Receive session data
RECEIVE_ANY	Receive session data
HANG_UP	Terminate session
SESSION_STATUS	Retrieve session status

NetBIOS requires one process to be the server and another to be the client. The server first issues a passive open using LISTEN. The client then connects with the server with the CALL command.

The LISTEN command requires the caller, to specify both the local name and the remote name. The local name is generally a *well-known port*. The remote name is the name of the client to which the server wants to connect. This can be specified as an *asterisk* (*) to specify that the server can connect to any client.

Both, the LISTEN and CALL, commands return the *local session number* to the calling program. This can be used in the SEND and RECEIVE command to specify the relevant session, as a process can have more than one session open.

This number is also used by the HANG_UP command to specify, which session is to be terminated. When a session is terminated, all pending data is first transferred.

When using the SEND command, NetBIOS waits for the positive acknowledgment to come before returning to the caller. Similarly, RECEIVE first sends the acknowledgment and then returns.

RECEIVE_ANY allows a process to receive a message from any of the sessions currently active.

2.4 Datagram Service

NetBIOS supports datagrams of upto 512 bytes in length. Datagrams can be sent to a unique name, multicast to a group or broadcast to the entire network. These datagrams are connectionless and unreliable. The four datagram commands are:

COMMAND	DESCRIPTION
SEND_DATAGRAM	Send datagram to a process
SEND_BROADCAST_DATAGRAM	Broadcast datagram to all
RECEIVE_DATAGRAM	Receive datagram from a process
RECEIVE_BROADCAST_DATAGRAM	Receive a broadcast datagram

2.5 General Commands

There are four general commands

COMMAND	DESCRIPTION
RESET	Reset NetBIOS
CANCEL	Cancel an asynchronous command
ADAPTER_STATUS	Fetch adapter status
UNLINK	Unlink from bootstrap server

3. Open Systems Interconnection (OSI) Protocols

3.1 Introduction

The OSI model provides a framework within which standards can be developed for protocols at each layer. Most of the popular network protocols predate the OSI model. The Data Link layer uses various standards for providing the logical link. The Network layer provides both a connection-oriented and connection-less service.

3.2 Transport Layer

The task of the transport layer is to provide, reliable, end-to-end data transfer for uses of the transport layer. Expedited data or out-of-band data is supported by the transport layer. Upto 16 bytes of expedited data can be sent in a single operation. The protocol says that normal data sent after expedited data must not reach the peer end before the expedited data. The definition of the transport layer also includes features such as connection establishment and negotiation of parameters between the end points. There are three types of network services provided by this layer :

CLASS	DESCRIPTION
Type A	A reliable network service. The network layer and the data link layer handle all error conditions.
Type B	A reliable network service with error notification. Although most of the error is again handled by the network and data link layer, some indication can be passed on to the transport layer. For example a reset for the network layer means that the transport layer should resynchronize with its peer.
Type C	An unreliable network service. This is the type provided by datagram-oriented networks.

The ISO connection-less transport protocol is similar to UDP. The packet contains the source address, the destination address, an optional checksum and the user data.

3.3 Session Layer

Two services provided by the session layer are:

1. Session establishment.
2. Session release.

A single transport connection can be used for one session or multiple transport connections can be used for one session connection. This is transparent to the user. Multiple sessions, however, cannot use a single transport connection.

Another service provided by this layer is *Dialog Management*. In this, an imaginary token is maintained between the two session layers and whoever holds the token transmits.

The other services provided by this layer are :

1. Synchronization.
2. Activity management.
3. Exception reporting.

3.4 Presentation Layer

This layer is concerned with the representation of data that is being exchanged. This includes :

1. Conversion of data from one format to another (ASCII to EBCDIC).

2. Encryption.
3. Compression.

One task of this layer is to convert the data to be transferred to a standard format. It uses *Abstract Syntax Notation* (ASN).

3.5 Application Layer

The services provided by this layer are:

1. Electronic Mail.
2. Directory Services.
3. Virtual Terminal.
4. File Transfer Access and Management.

4. Systems Network Architecture

4.1 Introduction

Introduced by IBM and now the predominant method of forming internets for IBM computers. SNA has a different architecture than that of the TCP/IP or the XNS protocols. The user of a network is defined as an *end user*. This is either a user at a terminal or a process. The user interacts with the *logical unit*. This provides some features like communication between processes. The LU interacts with the *physical unit*. This is the operating system of the node. It is also called the *path control layer*. The PU interacts with the *data link*. This is the module that takes care of the low level data transmission. This is generally the device driver section of the operating system. Two forms of SNA networks have evolved :

Subarea SNA networks	These networks are built around mainframes and maintain a centralized control over the network.
Advanced Peer to Peer Networks	APPN do not have a centralized component and can be connected using SNA subarea networks.

4.2 Path Control Layer and Logical Unit

It provides a *virtual circuit* to its upper layer (LU). That means, the PCL and the layers below provide error control, flow control and sequencing. Every LU in a given SNA network must have a unique name, from 1 to 8 characters. In an SNA Internet, every network must have a unique name, again form 1 to 8 characters. Thus a particular LU can be identified by

1. Network name.
2. LU name within the network.

The network qualified LU name is written as *netname.LUname*. The mapping from this network qualified LU name to the physical address is handled by a portion of the LU called *directory services.* Path control also does packet fragmentation and reassembly. This is called *segmenting* in SNA terminology. The LU appends a header to the data passed to it that is called *request header*. The data and the request header are together known as *Basic Information Unit* (BIU). The PCL appends its own header to the BIU called the *transmission header*. The complete unit is called *Path Information Unit*(PIU).

4.3 LU 6.2 - Advanced Program to Program Communication

This layer provides a connection oriented, reliable, half-duplex service to an application program. It also provides positive acknowledgments, time-out and retransmission, duplicate detection and sequencing for reliability.

<u>*Sessions and Conversations*</u>: The peer to peer connection between two user processes is called a *conversation*. The peer to peer connection between two LUs is called a *session*. A session is generally a long-term connection between two LUs and a conversation is generally short term. Sessions are expensive to establish so a typical LU establishes a certain number of connections with its partner LUs. This forms a pool of active sessions for a LU to manage. When a user process wants a conversation, the LU picks an available session from the pool and dedicates for the conversation. When it finishes, the LU returns the session to the pool.

Each LU associates a conversation with a unique identifier called the *conversation id*. It is a 16-bit integer. The id is carried in the transmission header and is used by APPN to specify the user process.

The association consists of :

1. The protocol.
2. The local network qualified name.
3. The local conversation id.
4. The foreign network qualified name.
5. The foreign conversation id.

LU 6.2 provides two interfaces:

1. Mapped conversations also called *basic conversation*.
2. Unmapped conversations.

The major difference between the two types is the format of data that is exchanged between the process and the LU. The interfaces between the user process and the LU are called *verbs*. These are invoked by the process to request a service from the LU. All user process data consists of a two byte length field(LL) and zero or more bytes of data.

LL 2 bytes	User data (0 - 32765 bytes)

The length field includes the two bytes of the LL field so it is always greater than or equal to 2. While using mapped conversation, the user data is transformed to the mapped version called *mapped conversation record*. This is then appended to the id and LL to form a logical record. This logical record is also called a *Generalized Data Stream*(GDS) variable. LU 6.2 internally buffers everything that one process sends to another.

4.4 Application layer

IBM differentiates between two type of application processes :

1. Application transaction programs (ATP).
2. Service transaction programs (STP).

STPs are supplied by IBM and can provide services for ATPs. ATPs are typically user processes.

5. Xerox Network Systems

5.1 Introduction

Introduced by Xerox Corporation for integrating their office products and computer systems. It is an open system and has been published and the protocols made available. Most systems support the XNS protocol e.g. 4.3BSD. XNS is similar to TCP/IP in structure. The protocols supported by XNS are:

Echo protocol	This protocol causes the host to echo the packet that it receives.
Routing Information Protocol	A protocol used to maintain a routing database for use on a host for the forwarding of IDP packets.
Packet Exchange Protocol	An unreliable, connectionless, datagram protocol for user processes. It supports retransmission but does not do duplicate detection.
Sequenced Packet Protocol	A connection oriented, reliable protocol for user processes. It provides a byte stream for the user process with optional message boundaries. It is similar to the TCP and is widely used.
Error Protocol	A protocol that can be used by any process to report that it has discovered an error and therefore discard a packet.
Internet Datagram Protocol	IDP is a connectionless, unreliable datagram protocol that provides the packet delivery service for all the above protocols. It is like the IP in the TCP/IP protocol suite.

The typical XNS network is an Ethernet, although leased lines or switched telephone lines can be used.

5.2 Network Layer

An XNS address occupies 12 bytes and is comprised of :

1. 32-bit network id.
2. 48-bit host id.
3. 16-bit port number called a *socket*.

The host id is an absolute number that must be unique across all Xerox networks. Typically the 48-bit host id is set to the 48-bit Ethernet address as most of the XNS networks are built using Ethernets. It is written as six hexadecimal digits separated by periods.

With unique host ids, the network id is redundant but it is still required for routing purposes. The network id must also be unique across all XNS Internets. The network id is typically written as a decimal integer. In this way XNS addresses form a flat address space.

5.3 IDP packets

Everything in XNS is eventually transmitted using IDP packets. IDP provides a connectionless and unreliable delivery service similar to the IP layer in the TCP/IP suite. Every IDP packet contains the following fields:

1. Source XNS address (host id, network id, port).
2. Destination XNS address (host id, network id, port).
3. Checksum.
4. Length of data (typically 0 - 546 bytes).
5. Higher layer packet type (SPP, PEX, etc.).

Differences between IDP and IP are:

IDP	IP
Contains a checksum that includes the entire IDP packet.	Checksum only for the IP header and not the data.
Contains 16 bit port number	Port number defined in TCP or UDP header.
IDP demultiplexes the incoming datagrams based on the port numbers.	IP demultiplexes the incoming datagram based on the protocol they use.

User can directly interact with the IDP protocol. Since IDP demultiplexes the datagrams on the port number, it is possible to receive more than one type of protocol packets on a port. e.g. An SPP protocol can receive ERROR packets directed on the same SPP port.

5.4 Transport Layer

Sequenced Packet Protocol: SPP, the predominant transport layer protocol in XNS, is very much similar to TCP. Unlike TCP which provides only a byte stream interface, there is a three level hierarchy in SPP:

1. Bytes are the basic entity.
2. A packet is composed of zero or more bytes.
3. A message is composed of one or more packets.

Using theses data forms, SPP presents three interfaces to the user :

Byte stream	The bytes are delivered to the user in order. Message boundaries are preserved. No packet boundaries.
Packet stream	Packets are delivered to the user in order. User process reads or writes entire packets. The user has to know how to decode the data from the packet by knowing the SPP header format.
Reliable packet	Packets are delivered to the user, but they might be out of order. Duplicate packets are discarded by the SPP software.

The two end users can use different SPP interfaces, e.g. The client can use the byte stream and the server can use the packet stream.

Packet Exchange Protocol: PEX is a datagram-oriented protocol similar to UDP. PEX retransmits the data when required. It does not do duplicate detection. PEX has a 6-byte header that includes a 32 bit id field and a 16 bit client type.

PEX operates as follows:

1. Client sets the id field to any value it wants and the client type field to a value that specifies the type of service requested. This is sent to the server.
2. The server performs the service specified by the client.
3. When the server forms the response packet, it must return the same id field specified by the client. This way the client knows that it has go the response for the request. The client must use another id for the next request.

The purpose of the client type field in the PEX header is to allow a single server to handle multiple requests, each service corresponding to a different client type. PEX module at the client retransmits the request if it does not get the response within a particular time frame.

5.5 XNS Bridge Fields

The XNS protocols define a *bridge field* in the header that is encapsulated by each layer. These fields are not used by the current layer but are interpreted by the next higher layer. There are three bridge fields:

Packet type	This is present in the IDP header and specifies the protocol of the data in the IDP packet. e.g. ERROR, etc.
Datastream type	This is present in the SPP header and is available to the user process using SPP. e.g. Used for shutting down the SPP connection between two SPP processes.
Client type	Present in the PEX header. Used by the user process to specify the type of service required.

There are some interfaces between the protocol module and the user process to inspect and set these bridge fields.

5.6 Other Features of XNS protocol

PEX is a datagram service so the concept of buffering and out of band data do not apply as no sequencing is done.

SPP provides a single byte of out of band data. The Xerox specification says that this byte of data should be made available to the user process as soon as possible. Also the SPP layer on getting this data copies it in a special buffer for the user process, and also puts it in its normal sequential position.

The IDP packets are not fragmented as IP datagrams. This is because they are limited to a very small size (576 bytes).