| Winsock & .NET | Winsock | < Client-Server Multicast Example | Linux Socket Index | TCP/IP Stack & RAW Socket > |

LINUX SOCKET PART 14
Advanced TCP/IP - The TCP/IP Stack & OSI Layer

My Training Period: xx hours

This is a continuation from Part III series, TCP & UDP Client-server program examples. Working program examples if any compiled using gcc, tested using the public IPs, run on Linux / Fedora 3, with several times of update, as root or SUID 0. The Fedora machine used for the testing having the "No Stack Execute" disabled and the SELinux set to default configuration.
This Module will concentrate on the TCP/IP stack and will try to dig deeper till the packet level.

The skills that supposed to be acquired:

- Able to understand the 7 layers OSI stack.
- Able to understand the 4 layers TCP/IP stack/suite/layer.
- Able to understand protocols in TCP/IP stack.
- Able to find and appreciate the RFCs and Standards.
- Able to understand and use the RAW socket (vs cooked socket).
- Able to understand and use for good purposes of the useful network tools that can be developed using RAW socket.

Menu

Network Story 1
Network Story 2
Network Story 3
Network Story 4
Network Story 5
Network Story 6
Socket Example 1
Socket Example 2
Socket Example 3
Socket Example 4
Socket Example 5
Socket Example 6
Socket Example

Introduction

In the previous Modules we just dealt with the generic TCP/UDP programs. It is not so usable without implementing other protocols at other layers of the TCP/IP suite. In this Module we will investigate deeper into the TCP/IP suite, their protocols, header formats and at the end we will try to construct our own packet using RAW packet. Let recall some of the information that we have already covered in the previous Modules and then proceed on to the details. The following figure shows various TCP/IP and other protocols reside in the original OSI model.

7	**Application**	e.g. HTTP, SMTP, SNMP, FTP, Telnet, SSH and Scp, NFS, RTSP etc.
6	**Presentation**	e.g. XDR, ASN.1, SMB, AFP etc.
5	**Session**	e.g. TLS, SSH, ISO 8327 / CCITT X.225, RPC, NetBIOS, ASP etc.
4	**Transport**	e.g. TCP, UDP, RTP, SCTP, SPX, ATP etc.
3	**Network**	e.g. IP/IPv6, ICMP, IGMP, X.25, CLNP, ARP, RARP, BGP, OSPF, RIP, IPX, DDP etc.
2	**Data Link**	e.g. Ethernet, Token ring, PPP, HDLC, Frame relay, ISDN, ATM, 802.11 Wi-Fi, FDDI etc.
1	**Physical**	e.g. wire, radio, fiber optic etc.

Figure 1: OSI layer.

As discussed in the previous Module, in implementation, the de facto standard used is the TCP/IP. This TCP/IP term should be general and here we will study the detail of the TCP/IP.

TCP/IP

The TCP/IP suite attempts to create a heterogeneous network with open protocols that are independent of operating system and architectural difference. TCP/IP protocols are available to everyone, and are developed and changed by consensus, not by one manufacturer. Everyone is free to develop products to meet these open protocol specifications. Most information about TCP/IP is published as **Request For Comments (**RFC**)**, which contain the latest version of the specifications of all TCP/IP protocols standard. The following figure shows the 4 layers of TCP/IP suite.

4	**Application layer**	BGP, FTP, HTTP, HTTPS, IMAP, IRC, NNTP, POP3, RTP, SIP, SMTP, SNMP, SSH, SSL, Telnet, UUCP, Finger, Gopher, DNS, RIP, Traceroute, Whois, IMAP/IMAP4, Ping, RADIUS, BGP etc.
3	**Transport layer**	DCCP, OSPF, SCTP, TCP, UDP, ICMP etc.
2	**Network/Internet layer**	IPv4, IPv6, ICMP, ARP, IGMP etc
1	**Physical/ Data Link layer**	Ethernet, Wireless (WAP, CDPD, 802.11, Wi-Fi), Token ring, FDDI, PPP, ISDN, Frame Relay, ATM, SONET/SDH, xDSL, SLIP etc. RS-232, EIA-422, RS-449, EIA-485 etc.

Figure 2: TCP/IP stack/layer/suite.

7
Advanced TCP/IP 1
Advanced TCP/IP 2
Advanced TCP/IP 3
Advanced TCP/IP 4
Advanced TCP/IP 5

Commonly, the top three layers of the OSI model (Application, Presentation and Session) are considered as a single Application layer in the TCP/IP suite and the bottom two layers as well considered as a single Network Access layer. Because the TCP/IP suite has no unified session layer on which higher layers are built, these functions are typically carried out (or ignored) by individual applications. The most notable difference between TCP/IP and OSI models is the Application layer, as TCP/IP integrates a few steps of the OSI model into its Application layer. A simplified TCP/IP interpretation of the stack is shown below:

Application	e.g. HTTP, FTP, DNS. (Routing protocols like BGP and RIP, which for a variety of reasons run over TCP and UDP respectively, may also be considered part of the Network layer)
Transport	e.g. TCP, UDP, RTP, SCTP. (Routing protocols like OSPF, which run over IP, may also be considered part of the Network layer)
Network/Internet	For TCP/IP this is the Internet Protocol (IP). (Required protocols like ICMP and IGMP run over IP, but may still be considered part of the network layer; ARP does not run over IP).
Physical/ Data Link/Network Access	e.g. Ethernet, Token ring, etc. e.g. physical media, and encoding techniques, T1, E1 etc.

Figure 3: Brief of the TCP/IP stack functions.

The basic function each of the TCP/IP layer is illustrated in the following figure.

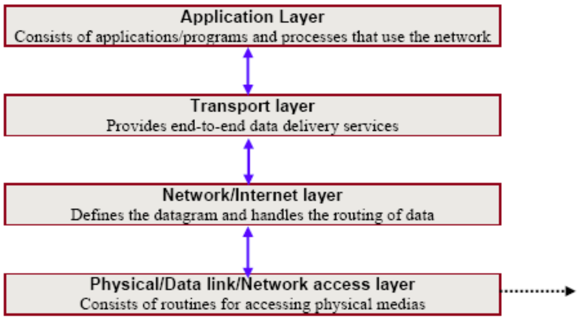

Application Layer
Consists of applications/programs and processes that use the network
Transport layer
Provides end-to-end data delivery services
Network/Internet layer
Defines the datagram and handles the routing of data
Physical/Data link/Network access layer
Consists of routines for accessing physical medias

Figure 4: Another TCP/IP basic stack functionalities.

A shown in figure 5, the four-layered structure of TCP/IP is seen in the way data handled as it passes down the protocol stack from the Application layer to the underlying physical network. Each layer in the stack adds control information to ensure proper delivery. This control information is called a **header** because it is placed in front of the data to be transmitted. Each layer treats all of the information it receives from the layer above as **data** and places its own header in front of that information. The addition of delivery information at every layer is called **encapsulation**. Note that the **real data** that will be transmitted, seen or used at Application layer just a small portion of the whole packet. When data is received, the opposite process happens. Each layer strips off its header before passing the real data on the layer above. As information flows back up the stack, information received from a lower layer is interpreted as both a header and data.

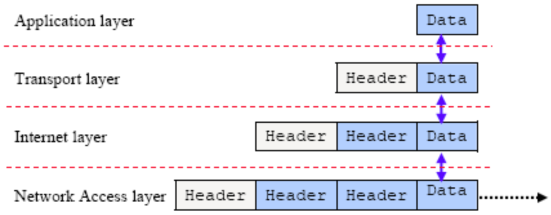
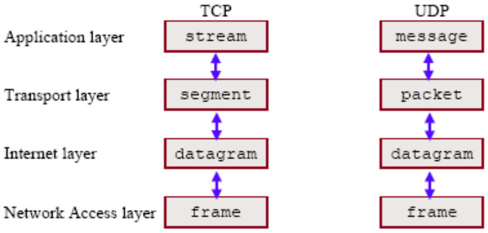


Figure 5: TCP/IP header encapsulation.

Each layer has its own independent data structures. Conceptually a layer is unaware of the data structure used by the layers above and below it. In reality, the data structures of a layer are designed to be compatible with the structures used by the surrounding layers for the sake of more efficient data transmission. Still, each layer has its own data structure and its own terminology to describe that structure. Figure 6 shows the terms used by different layers of TCP/IP to refer to the data being transmitted. As a general term, most networks refer to a transmitted data as **packets** of **frames**.



	TCP	UDP
Application layer	stream	message
Transport layer	segment	packet
Internet layer	datagram	datagram
Network Access layer	frame	frame

Figure 6: Different term of packet at different TCP/IP layers.

TCP/IP: The Detail

The following figure tries to give a big picture of what actually happen when a host (network device) communicates with another host in TCP/IP stack. The flow of the packets is two ways representing the terms send and receive. Using figure 7 as our reference, let investigate more detail of every layer starting from the bottom layer of the TCP/IP stack.

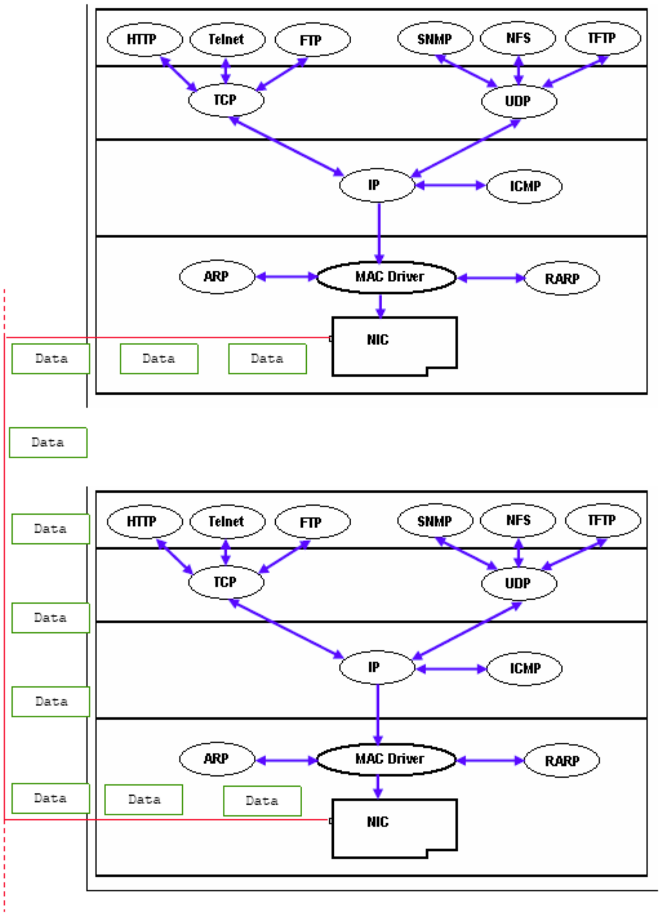


Figure 7: Quite a complete structure of protocols diagram used for communication.

Network Access Layer

The Network Access layer it is the lowest layer of the TCP/IP protocol hierarchy. The protocols in this layer provide the means for the system to deliver data to the other device on a directly attached network. It defines how to use the network to transmit an IP diagram. Unlike higher-level protocols, it must know the details of the underlying network to correctly format the data being transmitted to comply with the network constraints. The TCP/IP Network Access layer can encompass the function of all three lower layers of the OSI

reference model Network layer, Data Link layer, and Physical layer.
Functions performed at this level include encapsulation of IP datagrams into the frames transmitted by the network and mapping of IP addresses to the physical addresses used by the network (provided by ARP protocol). The network access layer is responsible for exchanging data between a host and the network and for delivering data between two devices on the same network. Node physical addresses (MAC address) are used to accomplish delivery on the local network.
TCP/IP has been adapted to a wide variety of network types, including switching, such as X.21, packet switching, such as X.25, Ethernet, the IEEE 802.x protocols, frame relay, wireless etc. For example, data in the network access layer encode EtherType (Ethernet) information that is used to demultiplex data associated with specific upper-layer protocol stacks.

Network/Internet Layer

The Internet layer is the heart of TCP/IP and the most important protocol. This layer provides the basic packet delivery service on which TCP/IP networks are built. The TCP/IP protocol at this layer is the **Internet Protocol** (IP- RFC 791). **All** protocols, in the layers above and below Internet layer, use the **Internet Protocol** to deliver data. **All** TCP/IP data flows through IP, incoming and outgoing, regardless of its final destination.
The Internet layer is responsible for **routing** messages through **internetworks**. Devices responsible for routing messages between networks are called **gateways** in TCP/IP terminology, although the term **router** is also used with increasing frequency. In addition to the physical node addresses utilized at the network access layer, the IP protocol implements a system of **logical host** addresses called **IP addresses**. The IP addresses are used by the **internet** and **higher layers** to identify devices and to perform internetwork routing. As discussed in the previous Module the IP address may be a class or classless type. The **Address Resolution Protocol (ARP)** enables IP to identify the physical address (Media Access Control, MAC) that matches a given IP address. The physical address has been burnt on every NIC. To make it readable for human being, the (domain) name is used instead of the IP address in normal operation. The IP address and name resolution is done by Domain Name System (DNS). In the implementation, UNIX/Linux uses BIND and Windows uses Domain Name Service (also DNS acronym). The relationship is shown in the following figure.

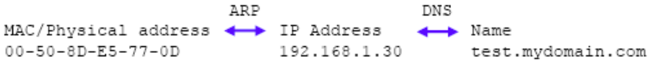


Figure 8: IP/Name resolution.

The IP provides services that are roughly equivalent to the OSI Network layer. IP provides a datagram (connectionless) transport service across the network. This service is sometimes referred to as **unreliable** because the network does not guarantee delivery nor notify the end host system about packets lost due to errors or network congestion. IP datagrams contain a message, or one fragment of a message, that may be up to 65,535 bytes (octets/bytes) in length. IP does not provide a mechanism for flow control. Let dig deeper about the protocols in this layer.

Internet Protocol (IP)

The IP protocol functionalities include:

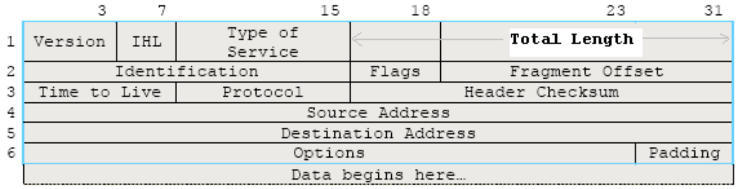
1. Defining the datagram, which is the basic unit of transmission in the Internet.
2. Defining the Internet addressing scheme, moving data between the Network Access layer and the Transport layer.
3. Routing datagrams to remote hosts.
4. Performing fragmentation and reassembly of datagrams.

The Datagram

Is a packet format defined by Internet Protocol. The internet protocol delivers the datagram by checking the **Destination Address (DA)**. This is an IP address that identifies the destination network and the specific host on that network. If the destination address is the address of a host on the local network, the packet is delivered directly to the destination; otherwise the packet is passed to a gateway/router for delivery. Gateways are devices that switch packets between the different physical networks. Deciding which gateway to use is called **routing**. IP makes the routing decision for each individual packet. IP deals with data in chunks called **datagrams**. The terms packet and datagram are often used interchangeably, although a packet is a data link-layer object and a datagram is a network layer object. In many cases, particularly when using IP on Ethernet, a datagram and packet refer to the same chunk of data. There's no guarantee that the physical link layer can handle a packet of the network layer's size. If the media's MTU is smaller than the network's packet size, then the network layer has to break large datagrams down into packed-sized chunks that the data link layer and physical layer can digest. This process is called **fragmentation**. The host receiving a fragmented datagram reassembles the pieces in the correct order.

IPv4 Datagram Format

The following figure shows the IPv4 datagram header format. It is 6 x 32 bits (word size) wide.



	3	7	15	18	23	31
1	Version	IHL	Type of Service	Total Length		
2	Identification			Flags	Fragment Offset	
3	Time to Live		Protocol	Header Checksum		
4	Source Address					
5	Destination Address					
6	Options					Padding
	Data begins here…					

Figure 9: The IP Datagram Format.

A brief field description:

Field	Description
Version	The version of IP currently used.
IHL	IP Header Length (IHL) - datagram header length. Points to the beginning of the data. The minimum value for a correct header is 5.
Type of Service	Data in this field indicate the quality of service desired. The effects of values in the precedence fields depend on the network technology employed, and values must be configured accordingly. Format of the Type of Service field: ▪ Bits 0-2: Precedence 111 = Normal Control. 110 = Internetwork Control. 101 = CRITIC/ECP. 100 = Flash Override. 011 = Flash. 010 = Immediate. 001 = Priority. 000 = Routine. ▪ Bit 3 : Delay 0 = normal delay, 1 = low delay. ▪ Bit 4 : Throughput 0 = normal throughput, 1 = high throughput. ▪ Bit 5 : Reliability 0 = normal reliability, 1 = high reliability. ▪ Bits 6-7: Reserved
Total Length	The length of the datagram in byte, including the IP header and data. This field enables datagrams to consist of up to 65,535 bytes. The standard recommends that all hosts be prepared to receive datagrams of at least 576 bytes in length.
Identification	An identification field used to aid reassembles of the fragments of a datagram.
Flags	If a datagram is fragmented, the MB bit is 1 in all fragments except the last. This field contains three control bits: ▪ Bit 0: Reserved, must be 0. ▪ Bit 1 (**DF**): 1 = **D**o not **f**ragment and 0 = May fragment. ▪ Bit 2 (**MF**): 1 = **M**ore **f**ragments and 0 = Last fragment.
Fragment Offset	For fragmented datagrams, indicates the position in the datagram of this fragment.
Time-to-live	Indicates the maximum time the datagram may remain on the network.
Protocol	The 8 bits field of the upper layer protocol associated with the data portion of the datagram. For a complete information please refer to RFC 1700 and the following is some of the protocol numbers: **Decimal** **Protocol** 1 ICMP (Internet Control Message) 2 IGMP (Internet Group Management) 4 IP (IP in IP -encapsulation) 5 ST (Stream) 6 TCP (Transmission Control) 17 UDP (User Datagram) 27 RDP (Reliable Data Protocol)
Header Checksum	A checksum for the header only. This value must be recalculated each time the header is modified.
Source Address	The IP address of the originated the datagram.
Destination Address	The IP address of the host that is the final destination of the datagram.
Options	May contain 0 or more options.
Padding	Filled with bits to ensure that the size of the header is a 32-bit multiple.

Table 1: IP datagram fields description.

Note that in the IP packet we just have the source and destination IP addresses. There is no source and destination port numbers here which is set in UDP or TCP header.

Internet Control Message Protocol (ICMP and ICMPv6)

Is part of the **Internet layer** and **uses the IP datagram delivery facility** to sends its messages. ICMP sends messages that perform control, error reporting, and informational functions for TCP/IP. The RFC document for ICMP is RFC 792. The following figure is the ICMP header format.

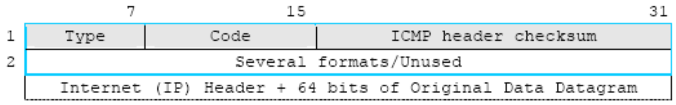


Figure 10: The ICMP Header Format.

A brief description:

Field	Description
Type	Messages can be error or informational messages. Error messages can be Destination unreachable, Packet too big, Time exceed, Parameter problem. The possible informational messages are, Echo Request, Echo Reply, Group Membership Query, Group Membership Report and Group Membership Reduction. A summary of message Types are listed below. 0: Echo Reply. 3: Destination Unreachable. 4: Source Quench. 5: Redirect. 8: Echo. 11: Time Exceeded. 12: Parameter Problem. 13: Timestamp. 14: Timestamp Reply. 15: Information Request. 16: Information Reply.
Code	For each type of message as listed above, several different codes are defined. An example of this is the Destination Unreachable message, where possible messages are: no route to destination, communication with destination administratively prohibited, not a neighbor, address unreachable, port unreachable. The code and its means for Destination Unreachable message is listed below. 0 = net unreachable. 1 = host unreachable. 2 = protocol unreachable. 3 = port unreachable. 4 = fragmentation needed and DF set. 5 = source route failed.
Checksum	The 16-bit one's complement of the one's complement sum of the ICMP message starting with the ICMP Type. For computing the checksum, the checksum field should be zero.
Second word (Several formats/unused	Several formats that match with certain IP header fields/depend on the Type and Code fields.

Table 2: ICMP datagram fields description.

The usage examples of the ICMP (together with IP) are listed below:

1. Flow control: When datagrams arrive too fast for processing, the destination host or intermediate gateway sends an ICMP Source Quench Message back to the sender. This tells the source to temporarily stop sending datagrams.
2. Detecting unreachable destinations: When a destination is unreachable, the system detecting the problem sends an ICMP Destination Unreachable Message to the datagrams source. If the unreachable destination is a network or host, the message is sent by an intermediate gateway. But if the destination is an unreachable port, the destination host sends the message.
3. Redirecting routes: A gateway sends the ICMP Redirect Message to tell a host to use another gateway, presumably because the other gateway is a better choice. This message can only be used when the source host is on the same network as both gateways.
4. Checking remote hosts: A host can send the ICMP Echo Message to see if a remote system's internet protocol is up and operational. When a system receives an echo message, it sends the same packet back to the source host (e.g. PING command).

Other message types include:

1. Information Request or Information Reply Message.
2. Timestamp or Timestamp Reply Message.
3. Parameter Problem Message.
4. Time Exceeded Message.

Unless otherwise noted under the individual format descriptions as explained above, the values of the Internet Protocol (IP) header fields for the ICMP are as follows:

IP Field	Description
Version	4.
IHL	Internet header length in 32-bit words.
Type of Service	0.
Total Length	Length of internet header and data in octets.
Identification, Flags, Fragment Offset	Used in fragmentation.
Time to Live	Time to live in seconds; as this field is decremented at each machine in which the datagram is processed, the value in this field should be at least as great as the number of gateways which this datagram will traverse.
Protocol	ICMP = 1.
Header Checksum	The 16 bit one's complement of the one's complement sum of all 16 bit words in the header. For computing the checksum, the checksum field should be zero. This checksum may be replaced in the future.

Source Address	The address of the gateway or host that composes the ICMP message. Unless otherwise noted, this can be any of a gateway's addresses.
Destination Address	The address of the gateway or host to which the message should be sent.

Table 3: IP fields description when used with ICMP.

Continue on next Module…TCP/IP and RAW socket, more program examples.

Further reading and digging:

1. Check the best selling C / C++, Networking, Linux and Open Source books at Amazon.com.
2. GCC, GDB and other related tools.

| Winsock & .NET | Winsock | < Client-Server Multicast Example | Linux Socket Index | TCP/IP Stack & RAW Socket > |