

| [Winsock & .NET](#) | [Winsock](#) | [< Linux Socket Index](#) | [TCP/IP Client Server Model](#) > |

---

# NETWORK PROGRAMMING LINUX SOCKET PART I: THE FUNDAMENTALS

My Training Period: xx hours

## 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](#)

**Note:** Program examples if any, compiled using [gcc](#) on **Linux Fedora Core 3** machine with several update, as normal user. The Fedora machine used for the testing having the "No Stack Execute" disabled and the **SELinux** set to default configuration.

### The abilities that supposed to be acquired:

- Able to understand the basic of client-server model.
- Able to understand the TCP/IP suite/stack/layer.
- Able to understand important protocols such as TCP and UDP.
- Able to understand and use the Unix/Linux C language socket APIs.
- Able to understand and implement several simple Client and server basic designs.

This Tutorial introduces a network programming using sockets. Some of the information is implementation specific but all the program examples run on Fedora 3 and compiled using gcc. The following are topics that will be covered briefly.

- Some Background Story.
- The Client-Server Model.
- Concurrent Processing.
- Programming Interface.
- The Socket Interface.
- Client Design.
- Example Clients.
- Server Design.
- Iterative, Connectionless Servers (UDP).

[Socket Example 6](#)  
[Socket Example 7](#)  
[Advanced TCP/IP 1](#)  
[Advanced TCP/IP 2](#)  
[Advanced TCP/IP 3](#)  
[Advanced TCP/IP 4](#)  
[Advanced TCP/IP 5](#)

[Advanced Winsock2 Tutorial](#)

- Iterative, Connection-Oriented Servers (TCP).
- Concurrent, Connection-Oriented Servers (TCP).
- Single-Process, Concurrent Servers (TCP).
- Multi protocol Servers.
- Multi service Servers.
- Managing Server Concurrency.
- Client Concurrency.

### Some Background Story

- This background story tries to introduce the terms used in network programming and also to give you the big picture.
- The following figure is a typical physical network devices connection.

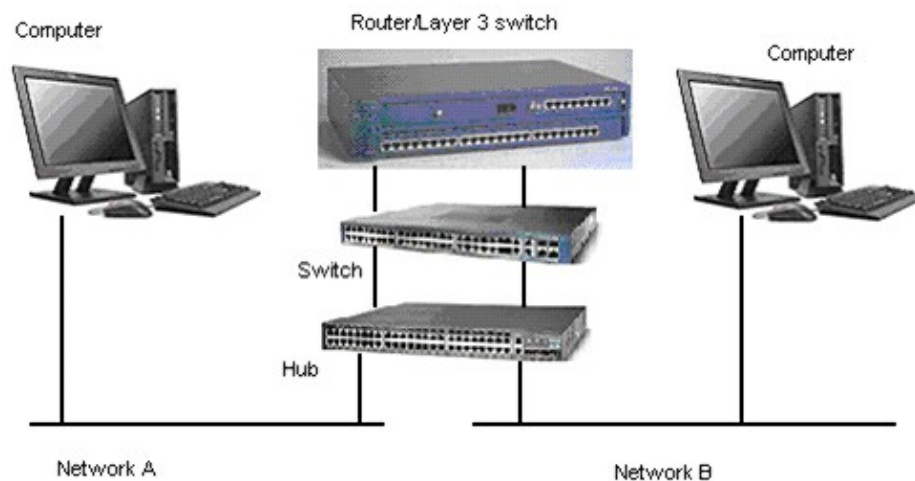


Figure 1

- Using a simple network shown in the above Figure, let trace the data stream flow from Network A to Network B, by assuming that Network A is company A's network and Network B is company B's network.
- Physically, the flow of the data stream is from a computer in Network A (source) will go through the hub, switch and router.
- Then the stream travel through the carrier such as Public Switch Telephone Network (PSTN) and leased line (copper, fiber or wireless – satellite) and finally reach Network B's router, go through the switch, hub and finally reach at the

computer in company B (destination).

- And from the previous network devices layout, the OSI (Open System Interconnection) 7 layer stack mapping is shown below.

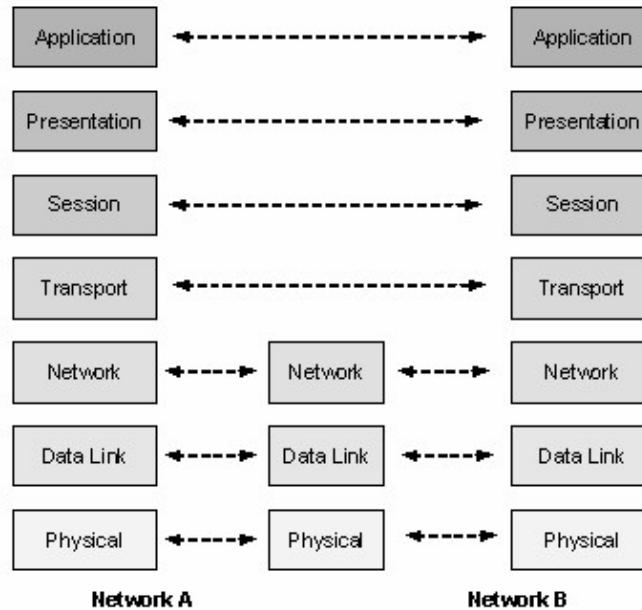


Figure 2

- From the Application layer of a computer at company A go downward the layer until the Physical (medium such as Cat 5 cable) layer, then exit Network A through the Network (router) layer in the middle of the diagram.
- After traveling through the carrier, reaches at the Network (router) layer of company B, travels through the Physical layer, goes upward until reaching at the Application layer of the computer at company B. Actually, at company B (the destination), the data flows through the network devices in the reverse manner compared to what happened at company A (the source).
- In contrast to TCP/IP, the OSI approach started from a clean slate and defined standards, adhering tightly to their own model, using a formal committee process without requiring implementations.
- Internet protocols use a less formal but more practical engineering approach, where anybody can propose and comment on Request For Comment (RFC) documents, and implementations are required to verify feasibility.
- The OSI protocols developed slowly, and because running the full protocol stack is resource intensive, they have not been widely deployed, especially in the desktop and small computer market.
- In the meantime, TCP/IP and the internet were developing rapidly, with deployment occurring at a very high rate, which is why the TCP/IP suite becomes a de facto standard.
- The OSI layer and their brief functionalities are listed in the following Table.

OSI Layer	Function provided
<b>Application</b>	Network application such as file transfer and terminal emulation
<b>Presentation</b>	Data formatting and encryption.
<b>Session</b>	Establishment and maintenance of sessions.

<b>Transport</b>	Provision for end-to-end reliable and unreliable delivery.
<b>Network</b>	Delivery of packets of information, which includes routing.
<b>Data Link</b>	Transfer of units of information, framing and error checking.
<b>Physical</b>	Transmission of binary data of a medium.

Table 1

- In the practical implementation, the standard used is based on TCP/IP stack. This TCP/IP stack is a de facto standard, not a pure standard but it is widely used and adopted.
- The equivalent or mapping of the OSI and TCP/IP stack is shown below. It is divided into 4 layers. The Session, Presentation and Application layers of OSI have been combined into one layer, Application layer.
- Physical and data link layers also become one layer. Different books or documentations might use different terms, but the 4 layers of TCP/IP are usually referred.

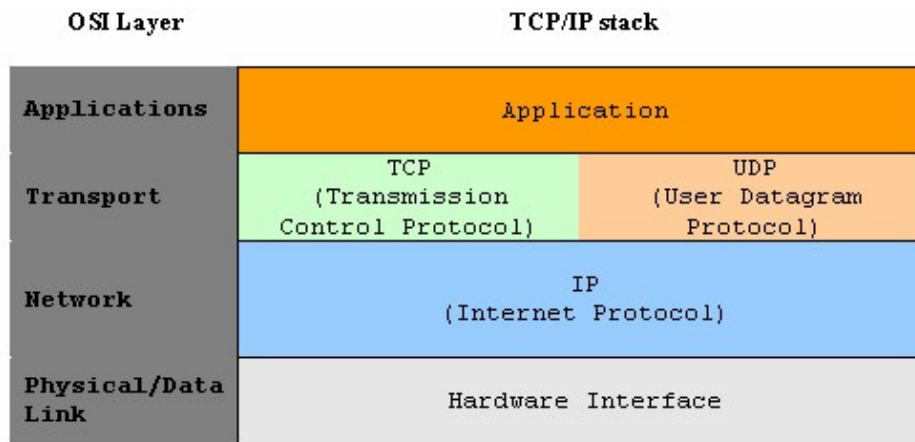


Figure 3

- In this Tutorial we will concentrate more on the Transport and Network layer of the TCP/IP stack.
- More detail TCP/IP stack with typical applications is shown below.

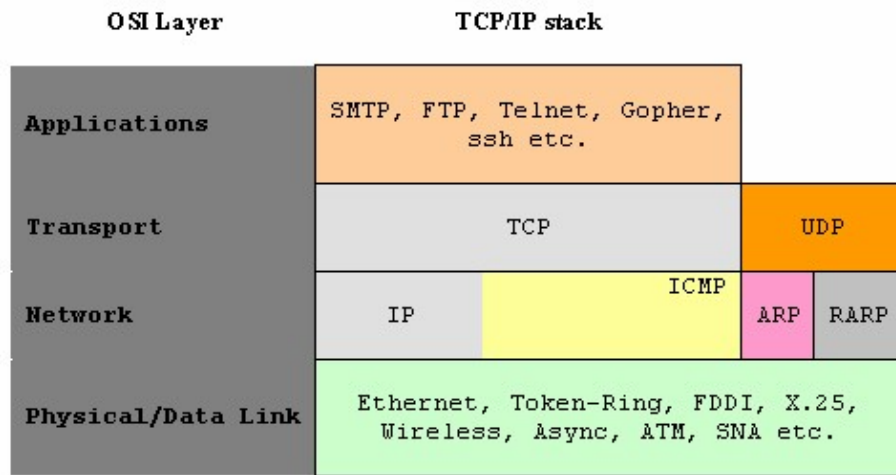


Figure 4

- The following figure is a TCP/IP architectural model. Frame, packet and message are same entity but called differently at the different layer because there are data encapsulations at every layer.

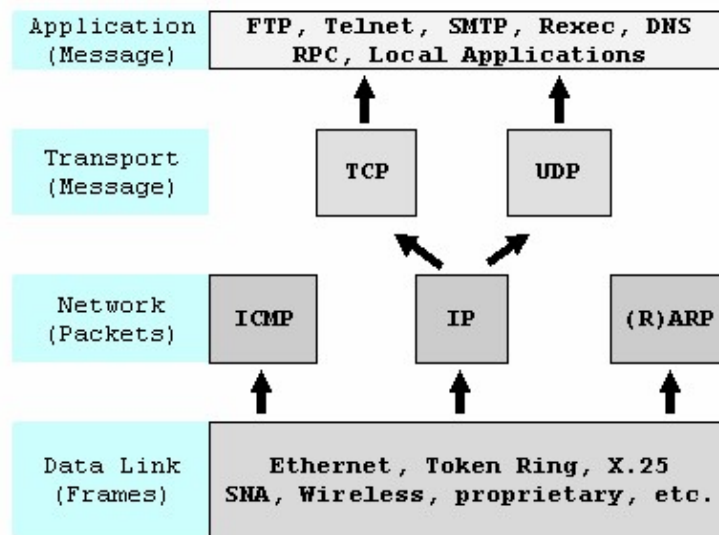


Figure 5

- The common applications that you encounter in your everyday use are:
  1. FTP (file transfer protocol).
  2. SMTP (simple mail transfer protocol).
  3. telnet (remote logins).
  4. rlogin (simple remote login between UNIX machines).
  5. World Wide Web (built on http) and https (secure http).
  6. NFS (network filing system – originally for Sun Microsystems).
  7. TFTP (trivial file transfer protocol – used for booting).
  8. SNMP (simple network management protocol).
- The user interfaces developed (programs) for the communication should depend

on the platform.

## Protocol

- In computing field, a protocol is a convention or standard rules that enables and controls the connection, communication and data transfer between two computing endpoints.
- Protocols may be implemented by hardware, software, or a combination of the two. At the lowest level, a protocol defines the behavior of a hardware connection.
- In term of controls, protocol may provide data transfer reliability, resiliency and integrity.
- An actual communication is defined by various communication protocols. In the context of data communication, a network protocol is a formal set of rules, conventions and data structure that governs how computers and other network devices exchange information over a network.
- In other words, protocol is a standard procedure and format that two data communication devices must understand, accept and use to be able to talk to each other.
- A wide variety of network protocols exist, which are defined by many standard organizations worldwide and technology vendors over years of technology evolution and developments. One of the most popular network protocol suites is TCP/IP, which is the heart of internetworking communications.

## TCP and UDP Protocols

- TCP and UDP protocols were built on top of the IP protocol.
- Basic for the TCP:
  1. Transmission Control Protocol is defined by [RFC-793](#).
  2. TCP provides connection-oriented transport service and reliable.
  3. End-to-end transparent byte-stream.
  4. E.g.: FTP, telnet, http, SMTP.
- While the UDP:
  1. User Datagram Protocol is defined by [RFC-768](#).
  2. UDP provides datagram service that is a packet based.
  3. Connectionless.
  4. Unreliable.
  5. E.g.: NFS, TFTP.

## Port numbers and services

- It is 16 bit integers. So we have  $2^{16} = 65536$  ports maximum.
- It is unique within a machine/IP address. Every service/application/daemon will have their own port number.
- To make a connection we need an IP address and port number of the protocol.
- The connection defined by:

IP address & port of server + IP address & port of client

- Normally, server port numbers are low numbers in the range 1 – 1023, normally called **well known port number** and normally assign for root (Administrator) only.
- It is used for authentication e.g. rlogin.
- And normally, client port numbers are higher numbers starting at 1024.
- A server running on a well-known port lets the OS know what port it wants to listen on.
- Whereas a client normally simply lets the operating system picks a new port that isn't already in use.

### Numeric IP Addresses

- Ipv4 (Internet Protocol version 4) Internet address is 32 bit integers. The IP stand for Internet Protocol.
- For convenience they are displayed in "dotted decimal" format.
- Each byte is presented as a decimal number.
- Dots separate the bytes, for example:

131.95.115.204

### IP Address Classes

- To simplify packet routing, internet addresses are divided into classes.
- An IP address has two parts: The network portion and the host portion.
- The network portion is unique to each company/organization/domain/group /network, and the host portion is unique to each network device (host) in the network.
- Where the network portion ends and the host portion begin is different for each class of IP address.
- You can determine this by looking at the two high-order bits in the IP address.

192.168.1.100

XXXXXXXX.XXXXXXXXXX.XXXXXXXXXX.XXXXXXXXXX

Byte 1.Byte 2.Byte 3.Byte 4

Class and Network size	Range (decimal)	Network ID	Host ID
Class A (Large)	1 -127	Byte 1	Bytes 2, 3, 4
Class B (Medium)	128 - 191	Bytes 1, 2	Bytes 3, 4
Class C (Small)	192 - 223	Bytes 1, 2, 3	Bytes 4

Table 2

- The first four bits (bits 0-3) of an address determine its class:

0xxx = class A - bits 1-7 define a network. bits 8-31 define a host on that network. So we've 128 networks and 2<sup>24</sup> hosts.

10xx = class B - bits 2-15 define a network. bits 16-31 define a host on that network. So we've 16384 networks and 2<sup>16</sup> hosts.

**110x = class C** - bits 3-23 define a network. bits 24-31 define a host on that network. So we've hosts.

Table 3

- The IP network portion can represent a very large network that may spans multiple geographic sites.
- To make this situation easier to manage, you can use subnetworks. Subnetworks use the two parts of the address to define a set of IP addresses that are treated as group. The subnetting divides the address into smaller networks.
- You configure a subnetwork by defining a mask, which is a series of bits. Then, the system performs a logical AND operation on these bits and the IP address.
- The 1 bit defines the subnetwork portion of the IP address (which must include at least the network portion). The 0 bits define the host portion.
- Class D is a multicast address and class E is reserved.
- As a summary:

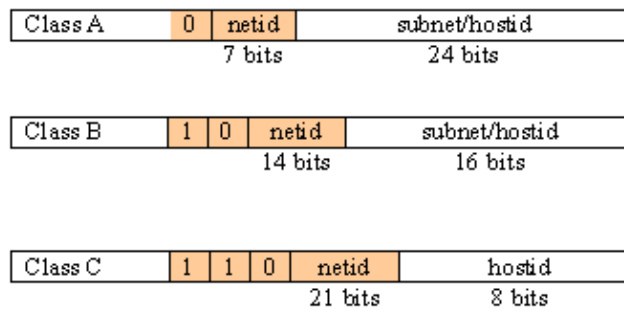


Figure 6

- Nowadays we use classless IP address. That means we subnet the class type IP into smaller subnet or smaller group of IP addresses creating smaller networks. The example can be found in [Classless Inter-Domain Routing \(CIDR\)](#) and the private domain normally uses the [private IP](#) range. The private IP range cannot be routed in the Internet domain.
- A network technology that deploy the private IP is [Virtual Private Network \(VPN\)](#) while the advanced subnetting technology using private IP can be found in [Virtual LAN \(VLAN\)](#).
- Before the IPV4 run out of the IP addresses, now we have IPV6 with 128 bits.

### Host Names and DNS

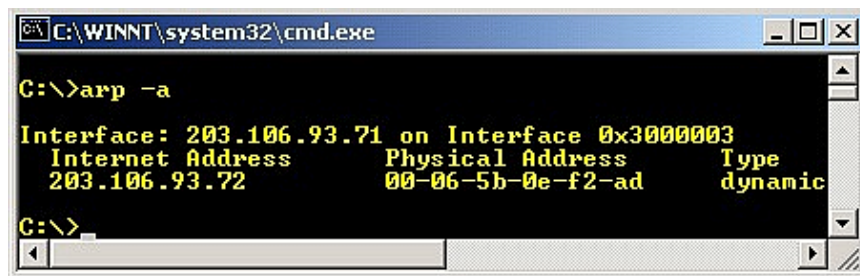
- People need names to make it simpler to use the Internet instead of the dotted decimal.
- The Domain Name System (DNS) can translate from name (domain name) to number (IP address) or from number to name. This is called name resolution.
- Name resolution done by Domain Name Service (DNS – although the term is same as the Domain Name System and same acronym, this is Microsoft implementation of the Domain Name System) in Windows and in Unices/Linux it is implemented using Berkeley Internet Name Domain ([BIND](#)).

### Ethernet Addresses



## MAC and ARP protocol

- In Local Area Network (LAN), based on the architecture, we have several network types such as Ethernet and Token Ring.
- The most widely used is Ethernet.
- Each Ethernet interface (Network Interface Card -NIC) has a unique Ethernet address provided by the manufacturer, hard coded into the NIC, normally called Media Access Control (MAC) or physical address.
- Ethernet addresses are 6 bytes shown as 6 hexadecimal values separated by colons.
- For example: **00:C0:F0:1F:3C:27**.
- You can see this MAC address by issuing the **arp** or **ipconfig** or **ifconfig** (Linux) command as shown below:



```

C:\WINNT\system32\cmd.exe

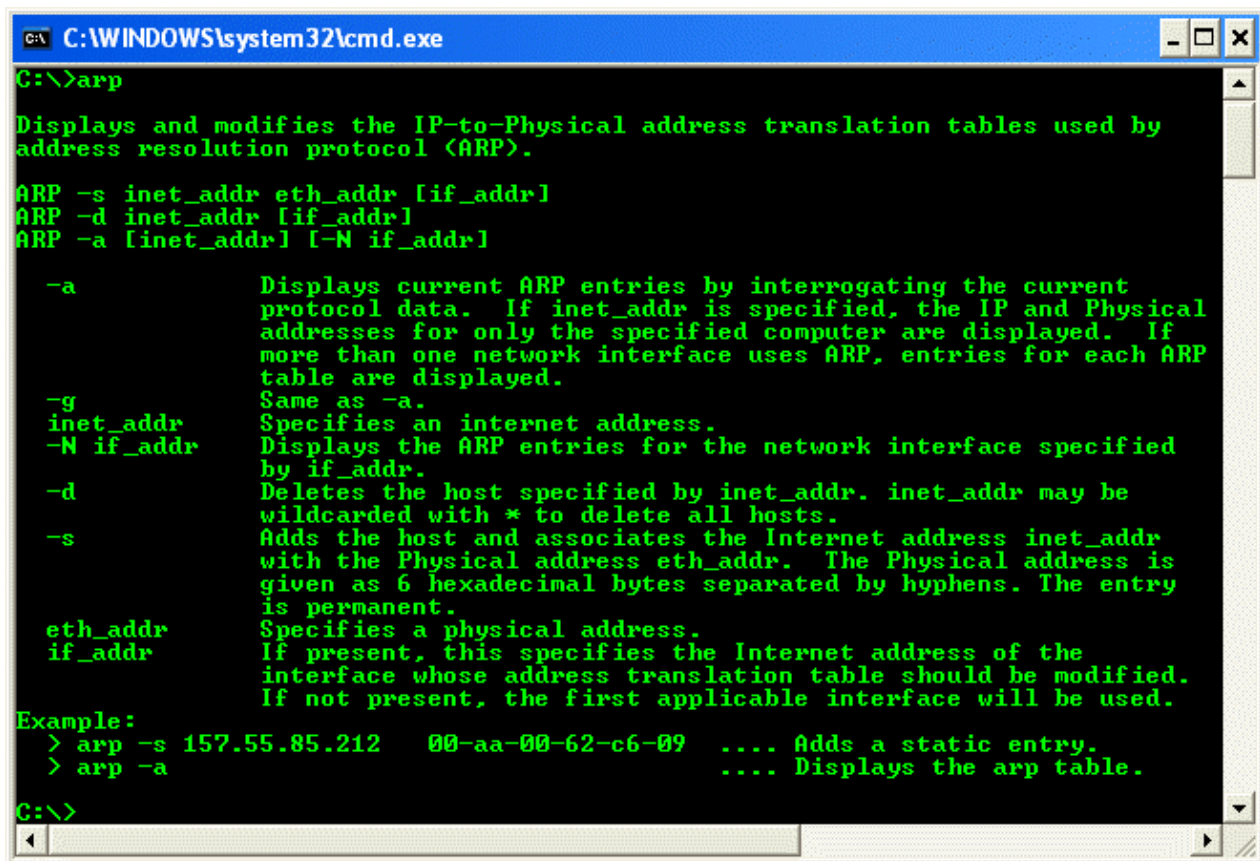
C:\>arp -a

Interface: 203.106.93.71 on Interface 0x30000003
Internet Address      Physical Address      Type
203.106.93.72        00-06-5b-0e-f2-ad    dynamic

C:\>

```

Figure 7



```

C:\WINDOWS\system32\cmd.exe

C:\>arp

Displays and modifies the IP-to-Physical address translation tables used by
address resolution protocol (ARP).

ARP -s inet_addr eth_addr [if_addr]
ARP -d inet_addr [if_addr]
ARP -a [inet_addr] [-N if_addr]

-a          Displays current ARP entries by interrogating the current
            protocol data. If inet_addr is specified, the IP and Physical
            addresses for only the specified computer are displayed. If
            more than one network interface uses ARP, entries for each ARP
            table are displayed.

-g          Same as -a.

inet_addr  Specifies an internet address.

-N if_addr Displays the ARP entries for the network interface specified
            by if_addr.

-d          Deletes the host specified by inet_addr. inet_addr may be
            wildcarded with * to delete all hosts.

-s          Adds the host and associates the Internet address inet_addr
            with the Physical address eth_addr. The Physical address is
            given as 6 hexadecimal bytes separated by hyphens. The entry
            is permanent.

eth_addr   Specifies a physical address.

if_addr    If present, this specifies the Internet address of the
            interface whose address translation table should be modified.
            If not present, the first applicable interface will be used.

Example:
> arp -s 157.55.85.212 00-aa-00-62-c6-09 .... Adds a static entry.
> arp -a          .... Displays the arp table.

C:\>

```

Figure 8

- Ethernet packets have header and data sections.
- The header contains the source and destination of the Ethernet addresses (MAC) and a 2 byte packet type.
- For IP packets the data area contains the IP fields which hold the IP source and destination addresses that are readable and more suitable for human.
- To send to an IP address, a computer uses the Address Resolution Protocol (**arp**) to determine a MAC address.

### IPv6 - Internet Protocol version 6

- Current IP is IPv4 (Internet Protocol version 4).
- IPv4 has 32 bit addresses.
- Due to splitting addresses, 32 bits is not enough.
- IPv6 will have 128 bit addresses.
- Addresses will be shown in a colon hexadecimal format with internal strings of 0s omitted. For example:

69DC:88F4:FFFF:0:ABCD:DBAC:1234:FBCD:A12B::F6

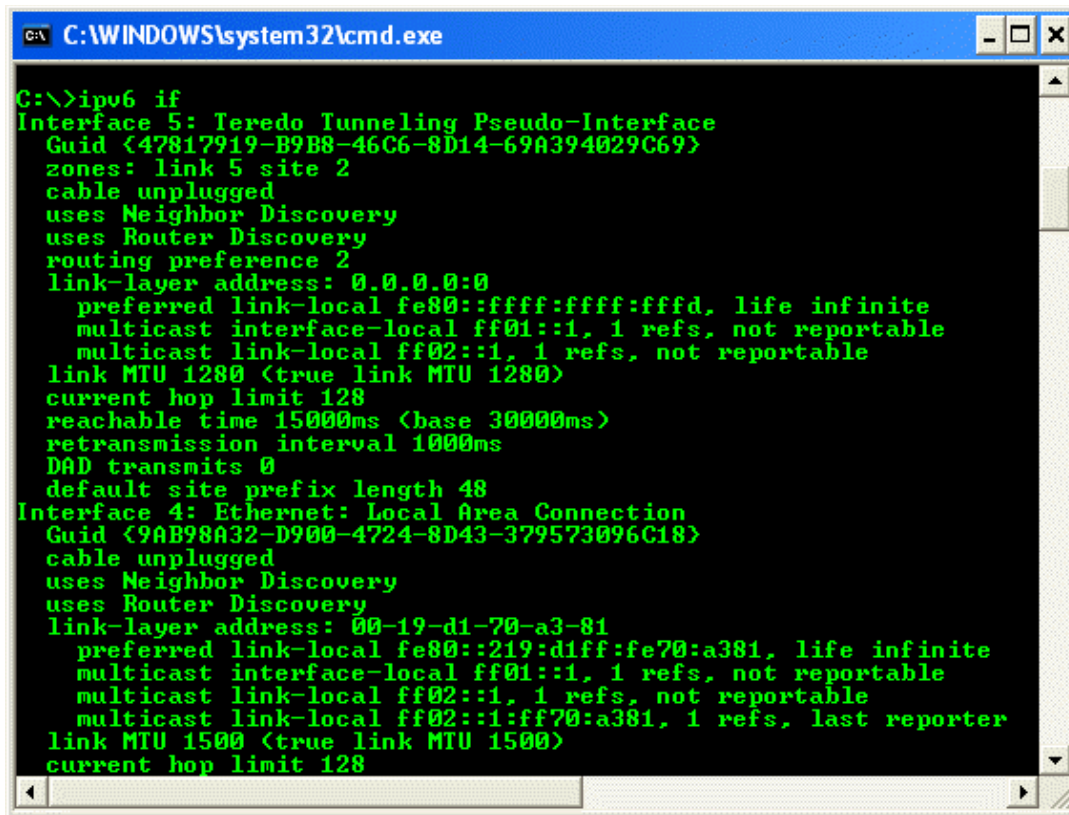
- New service types exist to accommodate IPv6 such as in multimedia and wireless fields.
- In Windows Xp and above, you can try the **ipv6** command at the prompt to view and/or set the IPv6 configuration. For example: **ipv6 if**.

```

C:\WINDOWS\system32\cmd.exe

C:\>ipv6
usage: ipv6 [-p] [-v] if [ifindex]
        ipv6 [-p] ifcr v6v4 v4src v4dst [nd] [pml]
        ipv6 [-p] ifcr 6over4 v4src
        ipv6 [-p] ifc ifindex [forwards] [-forwards] [advertises] [-advertises]
        ipv6 rlu ifindex v4dst
        ipv6 [-p] ifd ifindex
        ipv6 [-p] adu ifindex/address [life validlifetime[/preflifetime]]
        ipv6 nc [ifindex [address]]
        ipv6 ncf [ifindex [address]]
        ipv6 rc [ifindex address]
        ipv6 rcf [ifindex [address]]
        ipv6 bc
        ipv6 [-p] [-v] rt
        ipv6 [-p] rtu prefix ifindex[/address] [life valid[/pref]] [pref]
        ipv6 spt
        ipv6 spu prefix ifindex [life L]
        ipv6 [-p] gp
        ipv6 [-p] gpu [parameter value] ... <try -?>
        ipv6 renew [ifindex]
        ipv6 [-p] ppt
        ipv6 [-p] ppu prefix precedence P srclabel SL [dstlabel DL]
        ipv6 [-p] ppd prefix
        ipv6 [-p] reset
        ipv6 install
        ipv6 uninstall
Some subcommands require local Administrator privileges.
C:\>_
  
```

Figure 9



```

C:\>ipconfig /all

Interface 5: Teredo Tunneling Pseudo-Interface
    Guid {47817919-B9B8-46C6-8D14-69A394029C69}
    zones: link 5 site 2
    cable unplugged
    uses Neighbor Discovery
    uses Router Discovery
    routing preference 2
    link-layer address: 0.0.0.0:0
        preferred link-local fe80::ffff:ffff:ffff, life infinite
    multicast interface-local ff01::1, 1 refs, not reportable
    multicast link-local ff02::1, 1 refs, not reportable
    link MTU 1280 (true link MTU 1280)
    current hop limit 128
    reachable time 15000ms (base 30000ms)
    retransmission interval 1000ms
    DAD transmits 0
    default site prefix length 48
Interface 4: Ethernet: Local Area Connection
    Guid {9AB98A32-D900-4724-8D43-379573096C18}
    cable unplugged
    uses Neighbor Discovery
    uses Router Discovery
    link-layer address: 00-19-d1-70-a3-81
        preferred link-local fe80::219:d1ff:fe70:a381, life infinite
    multicast interface-local ff01::1, 1 refs, not reportable
    multicast link-local ff02::1, 1 refs, not reportable
    multicast link-local ff02::1:ff70:a381, 1 refs, last reporter
    link MTU 1500 (true link MTU 1500)
    current hop limit 128
  
```

Figure 10

## Distributed Applications

- The goal is to hide the fact that the application is distributed other than to provide the redundancy for reliability.
- User interfaces can look identical.
- Typically data resides on remote systems.
- In many instances, remote users interact with each other.

## Application Protocols

- Protocol is a set of rules defining how to communicate.
- Application protocol: communication rules for an application.
- Standard protocols: documented in RFCs such as ftp, telnet, http.
- Non-standard protocols: programmers write a distributed application = new protocol.
- Programmers choose standard protocols where they apply.
- E.g. telnet:

```

telnet computer.some.where [port_number]
telnet www.yahoo.com 23
  
```

- Port is a number defining which service to connect to.
- For example, port 23 is the default for telnet services.
- In Linux, ports, protocols and service names are specified in [/etc/services](#).
- We will learn more detail regarding the port, protocol and service another Modules.

### Providing **Concurrent** Access to Services

- Users expect almost immediate response.
- Network servers must handle multiple clients "apparently simultaneously".
- The CPU resources and network must be shared.
- Normally in the form of multiple server's **processes** or multiple server's **threads** in one process.

*...Continue on next Module...* More in-depth discussion about TCP/IP suite is given in [Advanced TCP/IP Tutorials](#).

### Further reading and digging:

1. Check the [best selling C / C++, Networking, Linux and Open Source books at Amazon.com](#).
2. [Protocol sequence diagram examples](#).
3. [Another site for protocols information](#).
4. [RFCs](#).
5. [GCC, GDB and other related tools](#).

---

| [Winsock & .NET](#) | [Winsock](#) | [< Linux Socket Index](#) | [TCP/IP Client Server Model](#) > |