The Requirements of Internetworking

Networks come in differing topologies and speeds and (of course) no single network configuration suits everyone.

The technology known as *internetworking* draws the multitudes of networking technologies into a common framework that combines networks into *internets*. In general an internetwork must :

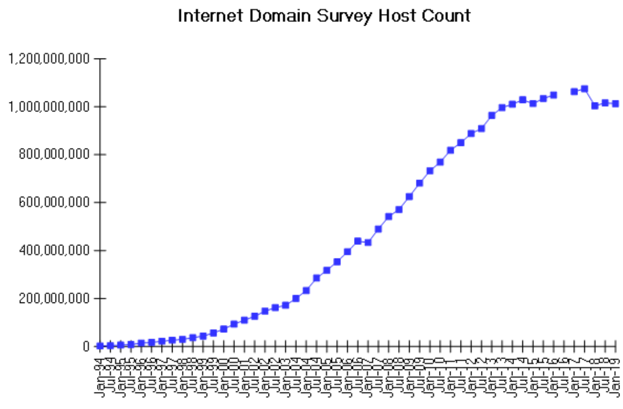
- Provide a link *between networks* (at minimum a physical and link control connection).
- Provide routing and delivery of data *between processes* on different networks (implies operating systems' support).
- Provide an accounting service, keeping track of the status of *networks and gateways*.
- Accommodate the differences between different sub-networks, including -
    - different physical addressing schemes,
    - different maximum packet sizes,
    - different network access mechanisms,
    - different timeout and retry schemes,
    - quality of error recovery,
    - quality of status reporting,
    - different routing, fault detection and congestion control techniques,
    - different data representations,
    - different user authentication practices, and
    - whether or not a connectionless or connection-based service is required.

Easy really!

A Brief History of the Internet

In 1969, the US Defense Advanced Research Projects Agency (DARPA) funded a project in computing resource sharing termed the ARPANET. ARPANET consisted of multiply connected, high-bandwidth (56 Kbps) links between government, academic and industrial laboratories.

During the 1970s DARPA was the chief funding body for packet switching networks using diverse technologies, such as mobile radio transmitters and satellite links. By the mid 1970s research concentrated on a framework for the ARPANET. By 1979 the Transmission Control Protocol/Internet Protocol (TCP/IP) was running exclusively over ARPANET.



Source: Internet Systems Consortium (www.isc.org)

Ref: ISC Internet Domain Survey
(and now for amusement purposes only).

As early as 2007, a white-paper from the *Pew Internet & American Life Project* reported that US-based Internet growth in *use*, is slowly slowing. More recently they report that growth in *value* is slowly slowing.

Other interesting readings (from our Resources page):

- *A Short History of the Internet*,
- *IP: 10 choices that were critical to the Net's success*,
- *Counting on the Internet (summary of findings)*.

The Initial Internetting Concepts

The *Internet Society* hosts a monograph named A Brief History of the Internet, by Vint Cerf *etal.*. Quoting directly from that paper (**READ IT!**):

Ground rules:

- Each distinct network would have to stand on its own and **no internal changes** could be required to any such network **to connect it to the Internet**.
- Communications would be on **a best effort basis**. If a packet didn't make it to the final destination, it would shortly be **retransmitted** from the source.
- Black boxes would be used to connect the networks; these would later be called **gateways and routers**. There would be no information retained by the gateways about the individual flows of packets passing through them, thereby **keeping them simple and avoiding complicated** adaptation and recovery from various failure modes.
- There would be **no global control** at the operations level.

Key issues:

- **Algorithms to prevent lost packets from permanently disabling communications** and enabling them to be successfully retransmitted from the source.
- Providing for **host-to-host pipelining** so that multiple packets could be enroute from source to destination at the discretion of the participating hosts, if the intermediate networks allowed it.
- Gateway functions to allow it to forward packets appropriately. This included interpreting **IP headers for routing, handling interfaces, breaking packets into smaller pieces** if necessary, etc.
- The need for **end-end checksums, reassembly of packets from fragments and detection of duplicates**, if any.
- The need for **global addressing**.
- Techniques for **host-to-host flow control**.
- Interfacing with the **various operating systems**.
- There were also other concerns, such as implementation efficiency, internetwork performance, but these were **secondary considerations** at first.

The Initial Internetting Concepts, *continued*

Emergent basic approaches:

- Communication between two processes would logically consist of **a very long stream of bytes** (they called them octets). The position of any octet in the stream would be used to identify it.
- **Flow control would be done by using sliding windows and acknowledgments**. The destination could select when to acknowledge and each acknowledgment returned would be cumulative for all packets received to that point.
- It was left open as to exactly how the source and destination would agree on the parameters of the windowing to be used. Defaults were used initially.
- Although Ethernet was under development at Xerox PARC at that time, the proliferation of LANs were not envisioned at the time, much less PCs and workstations. The original model was national level networks like ARPANET of which only a relatively small number were expected to exist. Thus **a 32 bit IP address** was used of which the first 8 bits signified the network and the remaining 24 bits designated the host on that network.

  This **assumption, that 256 networks would be sufficient for the foreseeable future, was clearly in need of reconsideration** when LANs began to appear in the late 1970s.

(Earlier) Milestones in Internet History

Cerf's paper notes these key milestones in Internet history:

- 1961: Leonard Kleinrock writes the first paper on packet switched networks.
- 1962: J.C.R. Licklider of MIT writes a paper describing a globally connected "Galactic Network" of computers.
- 1966: Larry Roberts proposes the ARPANET to the Defense Department's Advanced Research Projects Agency (ARPA).
- 1968: ARPA issues Request for Quotations for the Interface Message Processors (IMPs), which became the first routers.
- 1969: First IMP is installed at UCLA. Early 1970s: Universities and defense agencies and contractors begin to connect to ARPANET.
- 1973: Bob Kahn and Vint Cerf begin research into what eventually becomes IP - the Internet Protocol and its companion, TCP - the Transmission Control Protocol.
- 1973: Bob Metcalfe develops Ethernet, which had been the subject of his PhD thesis, while working at Xerox.
- Early 1980s: The Personal Computer revolution begins.
- Mid 1980s: Local Area Networks (LANs) begin to flourish in business and university environments. Campus area networks soon follow.
- January 1, 1983: All "old-style" traffic on the ARPANET ceases, as TCP/IP becomes the only protocol used. [Arguably, this is the date of the birth of the Internet as a functioning, practical, production network.]
- 1985: Dennis Jennings chooses TCP/IP as the protocol for the planned National Science Foundation Network (NSFnet).
- 1988: NSF sponsors a series of workshops at Harvard on the commercialization and privatization of the Internet.
- 1988: Kahn et al. write a paper "Towards a National Research Network." According to the Brief History, "This report was influential on then Senator Al Gore, and ushered in high speed networks that laid the networking foundation for the future information superhighway."
- 1991: Mark McCahill et al. (University of Minnesota) release the Internet Gopher, the first widely-adopted menu-based system for browsing and retrieving Internet-based documents.
- 1991: Tim Berners-Lee et al. at the European Center for High-Energy Physics (CERN) describe the World Wide Web. The first browser is a line-mode tool.
- March 1993: Mark Andreessen et al. at the National Center for Supercomputing Applications (NCSA) at the University of Illinois release Mosaic, the first widely-adopted graphical browser for the Web

Discussion

What advances were made over the **next 30+ years?** Would you classify them as *technical* or *societal*?

Where It All Began (?)

> *When I took office, only high energy physicists had ever heard of what is called the World Wide Web... Now even my cat has its own page.*
>
> *— Bill Clinton, announcing the Next Generation Internet Initiative, 1996*

> *During my service in the United States Congress, I took the initiative in creating the Internet.*
>
> *— Al Gore, 1999*

Internet RFCs (Requests For Comments)

The Internet is not (yet) a commerical product - instead a large, active research project.

Reports of work, proposals for protocols, and protocol standards appear as a series of nearly 8800 technical reports termed *RFCs*.

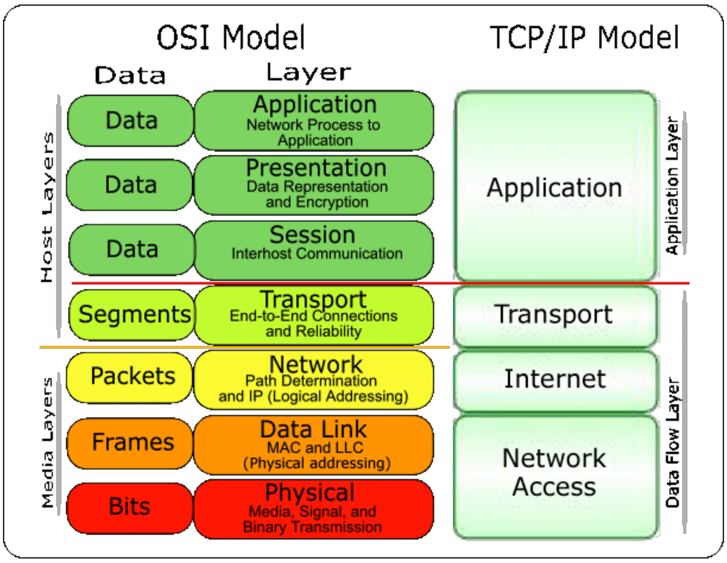
Recently: RFC 9233: Internationalized Domain Names for Applications 2008 (IDNA2008) and Unicode 12.0.0

The whole RFC collection may be searched and read via:

https://www.rfc-editor.org.

(and read the collection of humorous April Fools' Day Request for Comments).

The TCP/IP Protocol Architecture



The TCP/IP protocol suite is based on the view that communication involves three agents: *networks* (which contain hosts), *hosts* (which run processes), and *processes* (which generate and consume data).

Therefore, a network need only be concerned about routing data between hosts as long as the hosts agree how to get data to individual processes.

With this in mind, the TCP/IP architecture organizes protocols into four layers (some texts argue for a five-layer model, in which the Process/Application layer is split in two, with well-understood applications forming the fifth layer).

- Protocols in the *network access layer* route data between hosts (physically) attached in the same network.
- Protocols in the *internet layer* route data across multiple (possibly dissimilar) networks and hosts.

Internet layer protocols are implemented in both hosts and gateways - where a gateway is understood to connect two networks and must relay data between both networks, both using an Internet protocol.

The TCP/IP Protocol Architecture, *continued*

The *host/host layer* contains protocols able to deliver data between processes on the different hosts. Depending on the quality of service and the length of connections required (if any) at this layer, four host-host layer protocols are in frequent use -

- A reliable connection-oriented data protocol providing reliable, sequenced delivery (the Transmission Control Protocol, TCP, akin to the ISO Class 4 transport protocol) [RFC-793].

- A low overhead, minimum functionality datagram protocol (the User Datagram Protocol, UDP) [RFC-768].

- Reliable datagram protocols, providing low overhead communication for 'bursty' delivery between 'random' endpoints - good for speech and low-definition video streams [RFC-908].

- A real-time streaming protocol, characterized by the need for handling a steady stream with minimum delay variance [RFC-2326].

The *process/application layer* protocols define resource sharing (e.g. host-to-host) and remote access (terminal-to-host).

Well understood process/application layer protocols include the File Transfer Protocol (FTP), [RFC-959], HyperText Transport Protocol (HTTP), Simple Mail Transfer Protocol (SMTP) and the TELNET Protocol (for terminal emulation connections).

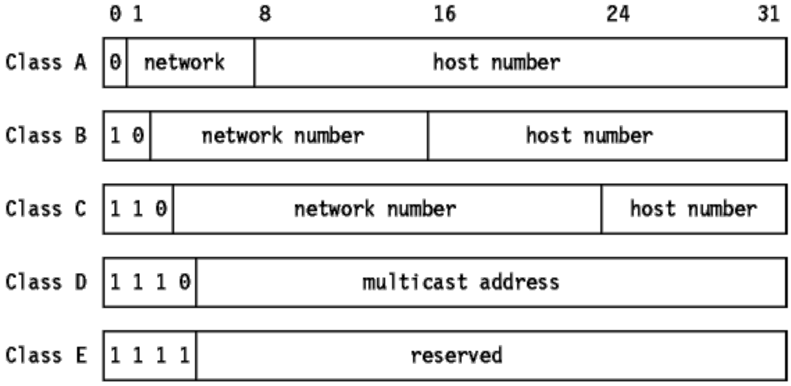
Traditional Class-based IP Version 4 Addressing

Each computer or device accessible using Internet (IP) protocols has at least one unique address within its subnet.

If every device was accessible over the global Internet, each device would require its own unique address (worldwide); each address consists of a *network* 'portion' and a *local* 'portion'. The network 'portions' are assigned by the central DARPA authority.

The original Internet designers were unsure as to how the Internet would grow - either a large number of networks each with a small number of hosts or a small number of networks each with a large number of hosts.

As a compromise, Internet addressing schemes accommodate both large and small network topologies.



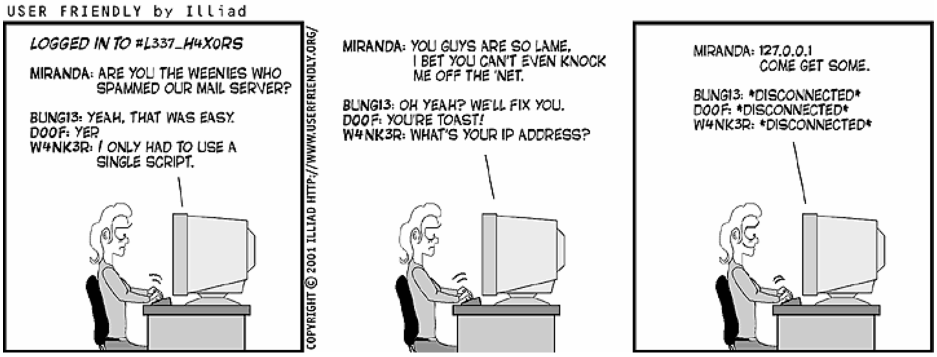
Moreover, class-based addresses are *self-describing*.

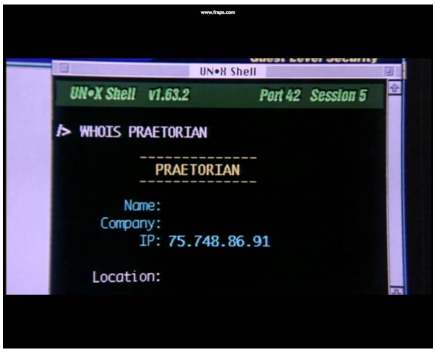
Class-based IP Version 4 Addressing, *continued*

When describing Internet addresses (verbally or in literature) a *dotted decimal notation* is used to describe the 32-bit addresses.

```
#
130.95.1.10      www        www.csse.uwa.edu.au
130.95.1.8       budgie     budgie.csse.uwa.edu.au
130.95.116.32    laser25    laser25.csse.uwa.edu.au
#
130.95.252.64    ecm-csse2101-l.uniwa.uwa.edu.au
130.95.252.112   ecm-csse2101-x.uniwa.uwa.edu.au
```

The different (hardware) encoding of 32-bit integers between architectures demands a standard representation for Internet addresses. The Internet standard for byte order specifies that integers are sent most significant byte first (*big-endian*).


USER FRIENDLY by Illiad
LOGGED IN TO #L337_H4XORS
MIRANDA: ARE YOU THE WEENIES WHO SPAMMED OUR MAIL SERVER?
BUNG13: YEAH, THAT WAS EASY.
DOOF: YEP.
W4NK3R: I ONLY HAD TO USE A SINGLE SCRIPT.
COPYRIGHT © 2001 ILLIAD HTTP://WWW.USERFRIENDLY.ORG/
MIRANDA: YOU GUYS ARE SO LAME. I BET YOU CAN'T EVEN KNOCK ME OFF THE 'NET.
BUNG13: OH YEAH? WE'LL FIX YOU.
DOOF: YOU'RE TOAST!
W4NK3R: WHAT'S YOUR IP ADDRESS?
MIRANDA: 127.0.0.1 COME GET SOME.
BUNG13: *DISCONNECTED*
DOOF: *DISCONNECTED*
W4NK3R: *DISCONNECTED*



```
▷ WHOIS PRAETORIAN
   --------------
    PRAETORIAN
   --------------
     Name:
   Company:
        IP: 75.748.86.91
 Location:
```

A number of standard C, Python, and Java library functions (methods, classes...) map Internet addresses to and from network standard ordering, Ethernet addresses and character strings (see Unix/Linux manuals for *inet*, *ethers* and *ntohl*).

Another significant problem with the Internet addressing scheme is that if a host moves from one *network* to another, it must change Internet addresses (consider the impact on *mobile-computing*).

Jonathan Swift's Gulliver's Travels, published in 1726, provided the earliest literary reference to computers, in which a machine would write books. This early attempt at artificial intelligence was characteristically marked by its inventor's call for public funding and the employment of student operators. Gulliver's diagram of the machine actually contained errors, these being either an attempt to protect his invention or the first computer hardware glitch.

The term endian is used because of an analogy with the story Gulliver's Travels, in which Swift imagined a never-ending fight between the kingdoms of the Big-Endians and the Little-Endians (whether you were Lilliputian or Brobdignagian), whose only difference is in where they crack open a hard-boiled egg.

Classless Inter-Domain Routing (CIDR)

With the advent of CIDR, the classful restrictions no longer exist. Address space may be allocated and assigned on bit boundaries, and routers may use one aggregated route (like 194.145.96.0/20) instead of advertising 16 class C addresses [RFC-1518].

addrs	bits	prefix	class	mask
1	0	/32	-	255.255.255.255
2	1	/31	-	255.255.255.254
4	2	/30	-	255.255.255.252
8	3	/29	-	255.255.255.248
16	4	/28	-	255.255.255.240
..	..	..	..	..
512	9	/23	2C	255.255.254.0
1K	10	/22	4C	255.255.252.0
..	..	..	..	..
64K	16	/16	1B	255.255.0.0
128K	17	/15	2B	255.254.0.0
256K	18	/14	4B	255.252.0.0
512K	19	/13	8B	255.248.0.0
1M	20	/12	16B	255.240.0.0
2M	21	/11	32B	255.224.0.0
4M	22	/10	64B	255.192.0.0
..	..	..	..	..
128M	27	/5	8A	248.0.0.0
256M	28	/4	16A	240.0.0.0
512M	29	/3	32A	224.0.0.0
1024M	30	/2	64A	192.0.0.0

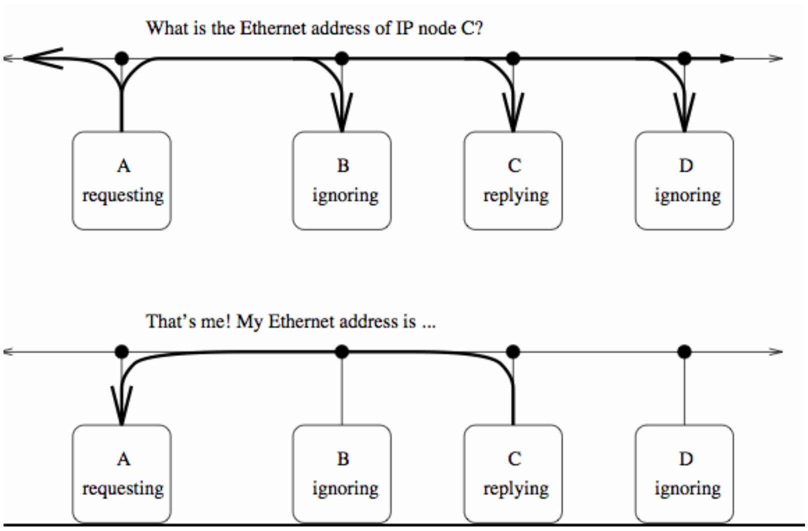
where:

- 'addrs' represents the number of addresses available; note that the number of addressable hosts normally is 2 less than this number because the host parts with all equal bits (all 0s and all 1s) are reserved.
- 'bits' represents the size of the allocation/assignment in bits of address space.
- 'prefix' represents the length of the route prefix covering this address space. This is sometimes used to indicate the size of an allocation/assignment.
- 'class' represents the size of the address space in terms of classful network numbers.
- 'mask' represents the network mask defining the routing prefix in dotted decimal notation.

Mapping Internet Addresses to Physical Addresses

An obvious question is *'What physical address should the sender use to send Internet datagrams to a specific Internet site?'*

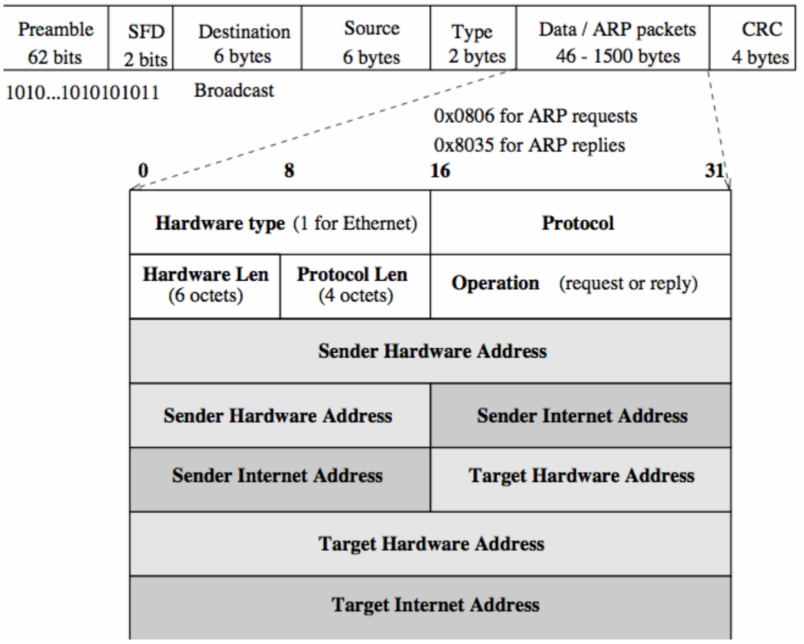
In some cases a physical address may fit into, say, a Class A Internet address, but more typically an Ethernet address (48 bits) will not fit in the Internet addressing schema.



The Address Resolution Protocol (ARP) is a special protocol designed to map Internet to physical addresses. When a gateway needs to know the physical address for an Internet address of a host known to be on its network, it broadcasts an ARP frame requesting the physical address. The required host replies; the gateway caches the address for future reference.

The Address Resolution Protocol (ARP)

ARP is a low-level protocol that hides the underlying physical addressing, permitting *one or more* Internet addresses to be assigned to each machine. ARP is considered part of the physical network system, not strictly part of the Internet protocols.



Preamble 62 bits	SFD 2 bits	Destination 6 bytes	Source 6 bytes	Type 2 bytes	Data / ARP packets 46 - 1500 bytes	CRC 4 bytes
1010...1010101011		Broadcast				

0x0806 for ARP requests
0x8035 for ARP replies

0	8	16	31
Hardware type (1 for Ethernet)		**Protocol**	
Hardware Len (6 octets)	**Protocol Len** (4 octets)	**Operation** (request or reply)	
Sender Hardware Address			
Sender Hardware Address		**Sender Internet Address**	
Sender Internet Address		**Target Hardware Address**	
Target Hardware Address			
Target Internet Address			

Unlike most protocols, ARP does not have a fixed format. Its design permits it to indicate how big its own fields will be, in this case that Ethernet addresses are 6 bytes (octets) long, and Internet addresses 4 bytes long. This permits ARP to be used with arbitrary network addressing schemes.

When making a request, the sender (making the request) also supplies its own Ethernet/Internet address mapping. As all hosts on the Ethernet monitor the broadcast, they can update their mapping tables for future reference (ARP snooping).

Configuration of Network Devices

All previous discussion on internetworking has assumed that our computers have been 'up and running', and possessed full knowledge about their networking environment. Where did this knowledge come from?

The standard booting sequence for most operating systems involves the computer's hard-disk (or CD-ROM, flash-memory, ...) providing a short *bootstrap* program of several hundred bytes, which in turn reads the true operating system code from nominated blocks on permanent media. To configure its network connection, a client host requires (at least):

- one unique IP address for each of its network interfaces,
- the client's *hostname*
- the address of its default router - where to send all packets that we don't explicitly know how to deliver,
- each interface's *subnet mask* to determine how many bits of the IP address provide the network and host ids,
- the IP address of an initial *domain name server*, to resolve host names to their IP addresses,
- (maybe) the time, or at least timezone.

A reasonable first approach to defining this information is in a configuration file (see our labs' */etc/sysconfig/network**):

```
DEVICE=eth0
BROADCAST=130.95.1.255
NETMASK=255.255.255.0
IPADDR=130.95.1.8
BOOTPROTO=none
GATEWAY=130.95.1.41
GATEWAYDEV=eth0
HOSTNAME=budgie.csse.uwa.edu.au
```

Problems With Static Configuration

There are a number of clear problems with static configuration of network attributes:

- System administrators may have to oversee hundreds of machines on a network. Manual maintenance of distinct files becomes intractable.
- A single (political) network domain may service many more (dialup or mobile) computers than it has registered IP addresses. Of course, this scenario hopes that not *all* computers wish to be connected at once.
- Mobile computers may wish to frequently connect to different (unrelated) networks, and
- Some previously trusted computers may become untrusted, and system administrators may have lost access to modify their configuration files.

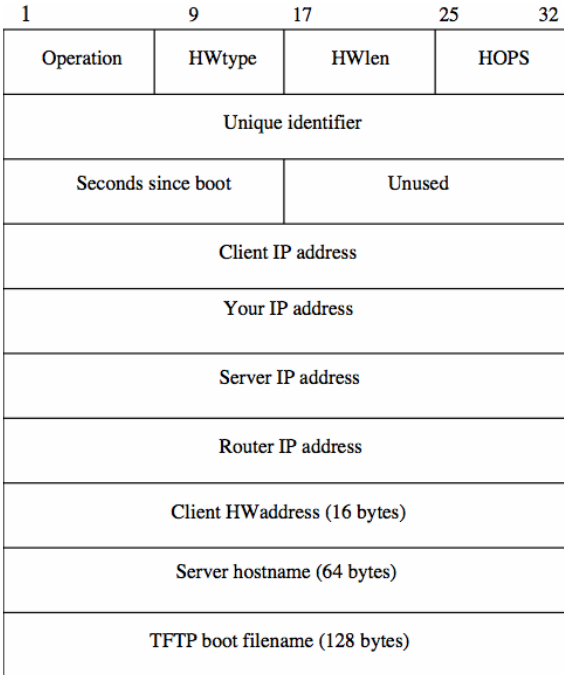
A partial solution

The most 'stable' attribute in most networking configurations is the network interface card's *MAC address*, such as a card's 48-bit Ethernet address. (many new Ethernet cards can change their MAC addresses programmatically - a mixed blessing!)

As with the ARP protocol described earlier, a newly booted client computer can broadcast its Ethernet address via the *Reverse Address-Resolution Protocol* (RARP).

The client broadcasts its RARP request, and any host acting as a RARP server may reply with the client's allocated IP address.

The Bootstrap Protocol (BOOTP)



1	9	17	25 32
Operation	HWtype	HWlen	HOPS
Unique identifier			
Seconds since boot		Unused	
Client IP address			
Your IP address			
Server IP address			
Router IP address			
Client HWaddress (16 bytes)			
Server hostname (64 bytes)			
TFTP boot filename (128 bytes)			

The *Bootstrap Protocol* (BOOTP) is a UDP/IP-based protocol that allows a booting host to configure itself dynamically, and more significantly, without user supervision. BOOTP is fully defined in RFC-1497. A server's BOOTP *response* includes several configuration items, and fits in a single (Ethernet) packet.

It provides a means to assign a host its IP address, a file from which to download a boot program from some server, that server's address, and (if present) the address of an Internet gateway.

One problem appears to have been introduced - how can BOOTP use a 'higher' protocol from the TCP/IP suite (in this case UDP) if it is trying to determine IP-level information? The answer lies in the use of special IP addresses within the BOOTP request packet:

- To send the BOOTP datagram, the client machine sets the destination IP address to the global broadcast (255.255.255.255), and its own source IP address to 0.0.0.0.
- The responding BOOTP server either replies with an IP broadcast (heard by the requesting client), or responds directly to the client's MAC address.

Booting over a Network

An additional feature of BOOTP is its support of providing a computer's (or any 'dumber' device's) operating system's image:

- The client may provide a generic string, such as *UNIX* or *NCD-XTERM-1320*, in the *TFTP-boot-filename* field of its request,
- The BOOTP server may accept this part of the request, and respond with the pathname of an operating system's kernel image available from the server, such as */tftp/bootimages/vmlinuz-4.14*
- The client may then use this pathname in a subsequent request using the *Trivial File Transfer Protocol* (TFTP, [RFC-783]) to fetch its boot image (loading it directly into memory).
- This feature is excellent for maintaining identical operating system kernels, or booting diskless machines.

Dynamic Host Configuration Protocol (DHCP)

DHCP's purpose is to enable individual computers on an IP network to extract their configurations from a server (the 'DHCP server').

In general, the servers will have no static information about the individual client computers until information is requested. Responses to each client will then be generated dynamically.

The overall purpose of this is to reduce the work necessary to administer a large (often dynamic) IP-based network.

The most significant piece of information distributed in this manner is the IP address.

DHCP is based on BOOTP and maintains some backward compatibility. The main difference is that BOOTP was designed for manual pre-configuration of the host information in a server database, while DHCP allows for dynamic allocation of network addresses and configurations to newly attached hosts. Additionally, DHCP allows for recovery and reallocation of network addresses through a *leasing* mechanism.

DHCP, like BOOTP, runs over the user-datagram protocol (UDP), using ports 67 and 68, and is defined in RFC-1534 and RFC-2131.

DHCP Configuration

The DHCP daemon (process) is typically provided with the name of a network interface (such as *eth0*) so that it knows from where to accept broadcast requests.

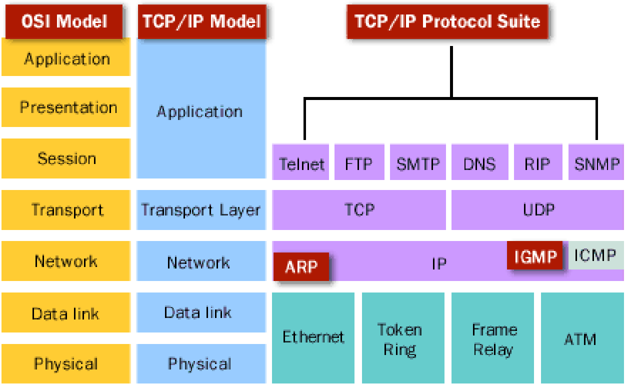
The DHCP daemon reads information from a configuration file storing the 'public' information for clients. To provide a truly *dynamic* configuration, we also need to provide a*range of IP addresses* given to clients.

We can also use DHCP to provide fixed configuration information, based on the Ethernet (MAC) address of the arriving request:

The TCP/IP Protocol Dependencies

The many Internet protocols naturally depend on each other, that is they demand the services provided by other protocols.

For example, the file-transfer protocol (*FTP*) demands that it operates over a reliable stream protocol (TCP), delivered to a host on a network (IP) which provides flow control (ICMP).

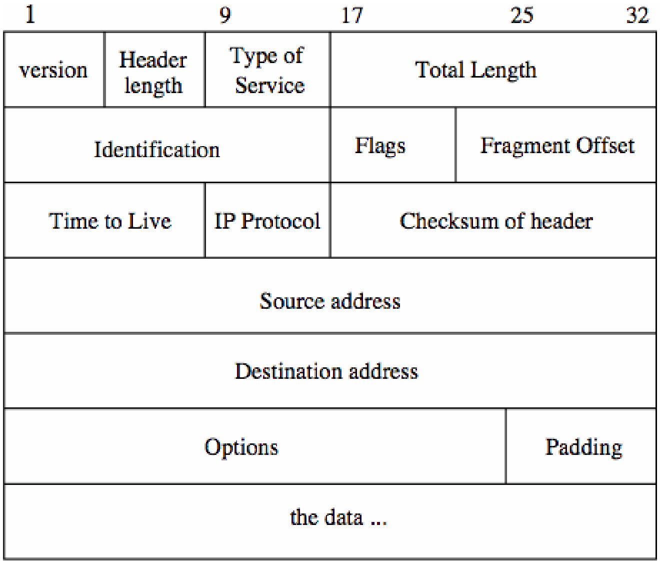


Internet Protocol (IP) Datagrams

The Internet Protocol (IP) provides an unreliable, best-effort, connectionless, packet delivery system.

In this unit we will initially be discussing Internet Protocol version 4 (IP v4) RFC-791.

Internet datagrams resemble 'standard' physical-layer frames, but are designed to be *encapsulated* within the normal network framing schema. Hence, Internet datagrams are said to run *on top of* traditional networks.



version	Header length	Type of Service	Total Length
Identification		Flags	Fragment Offset
Time to Live	IP Protocol	Checksum of header	
Source address			
Destination address			
Options			Padding
the data ...			

- Checksum: 16 bit checksum of the *header only*. The checksum is the one's complement of the one's complement sum of the header.
- TTL: Time to live - the hop count.
- Length: The datagram length (16 bits) in *octets*.
- Type: 8 bits, identifying protocol types.

```
# Internet (IP) protocols
#
ip      0     IP       # internet protocol, pseudo protocol number
icmp    1     ICMP     # internet control message protocol
igmp    2     IGMP     # internet group multicast protocol
ggp     3     GGP      # gateway-gateway protocol
tcp     6     TCP      # transmission control protocol
pup     12    PUP      # PARC universal packet protocol
udp     17    UDP      # user datagram protocol
```

Internet Control Message Protocol (ICMP)

ICMP allows gateways and hosts to exchange bootstrap and error information. Gateways send ICMP datagrams when they cannot deliver a datagram, or to direct hosts to use another gateway. Hosts send ICMP datagrams to test the 'liveness' of their network.

As an example, the Unix program *ping* sends ICMP *echo messages* to a specified machine. Upon receipt of the echo request, the destination returns an ICMP *echo reply*. *ping* hence both checks that a host is up and that the path to a host is viable.

```
prompt> /bin/ping elvis
elvis is alive

prompt> time /bin/ping sophia.inria.fr
sophia.inria.fr is alive
3.006s real 0.040s usr 0.060s sys

prompt> time /bin/ping sophia.inria.fr
sophia.inria.fr is alive
0.591s real 0.020s usr 0.090s sys
```

If a gateway must discard a datagram due to lack of resources it sends a *source quench* to the datagram's sender. If a datagram cannot be delivered because a host is down or no route exists, a ICMP *destination unreachable* datagram is generated.

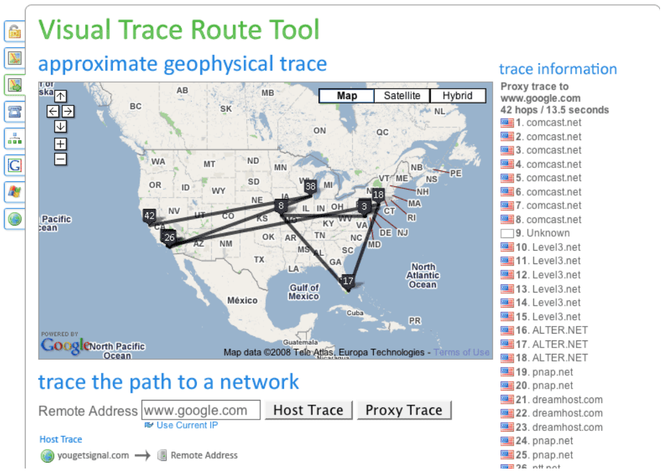
The TCP/IP Protocol suite defines over 25 (in-use) ICMP error message types, including:

> destination unreachable, time exceeded, parameter problems, source quench, redirection, echo, echo reply, timestamp, timestamp reply, information request and information reply.

Interesting Uses For ICMP - Traceroute

Traceroute utilizes the IP protocol `time to live' field and attempts to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host:

```
prompt>    traceroute www.cs.berkeley.edu
traceroute to hyperion.CS.Berkeley.EDU (208.1.75.105),
30 hops max, 40 byte packets
 1  cs-gate (130.95.1.41)  1.200 ms  1.602 ms ....
 2  parnet-uwa.parnet.edu.au (203.19.110.17)  7.567 ms ...
 3  atm11-0-7.ia3.optus.net.au (192.65.88.189)  54.287 ms ...
 4  atm91-4.ia1.optus.net.au (202.139.7.174)  63.938 ms ...
 5  h21.la1.optus.net.au (202.139.7.129)  370.803 ms  ...
 6  906.Hssi8-0.GW1.LAX2.ALTER.NET (157.130.224.137)  381.297 ms ...
 ...
17  f1-0-0.inr-107-eva.Berkeley.EDU (128.32.2.1)  869.557 ms ...
18  f1-0.inr-181-soda.Berkeley.EDU (128.32.120.181)  845.670 ms....
19  128.32.40.202 (128.32.40.202)  773.970 ms....
20  hyperion.CS.Berkeley.EDU (208.1.75.105)  949.824 ms....
```



Visual Trace Route Tool

approximate geophysical trace

trace information

Proxy trace to www.google.com
42 hops / 13.5 seconds

1. comcast.net
2. comcast.net
3. comcast.net
4. comcast.net
5. comcast.net
6. comcast.net
7. comcast.net
8. comcast.net
9. Unknown
10. Level3.net
11. Level3.net
12. Level3.net
13. Level3.net
14. Level3.net
15. Level3.net
16. ALTER.NET
17. ALTER.NET
18. ALTER.NET
19. pnap.net
20. pnap.net
21. dreamhost.com
22. dreamhost.com
23. dreamhost.com
24. pnap.net
25. pnap.net
26. ntt.net

trace the path to a network

Remote Address: www.google.com