**Chapter 5: The Network Layer.**

**Routing and forwarding:**

*Routing* is the process of deciding in which direction to send traffic.

*Forwarding* is the process of sending a packet on its way.

**The end-to-end argument:**

Two opposing trains of thought:

Internet community (“Netheads”): networks are unreliable; network should be “dumb” (this is the end-to-end argument)

* Networks should be connectionless; hosts should do error control
* Intelligence should lie on the outside of the network
* Enables the user to create any application

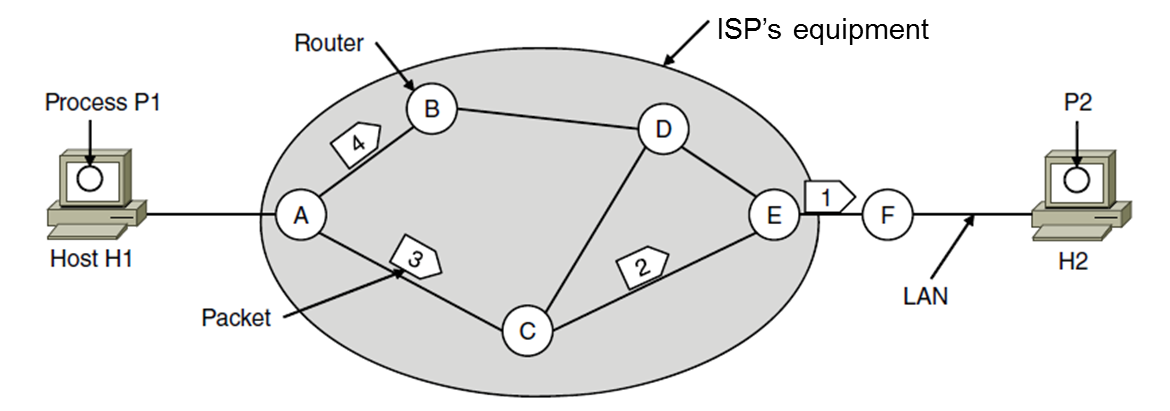
Telco community (“Bellheads”): networks should be reliable, devices should be “dumb”

* We’ll decide what you need (a phone) and QoS (connection-oriented)
* Intelligence should lie on the inside of the network
* We’ll create applications, such as 800 service or call waiting

**Packet switching:** Hosts send packets into the network; packets are forwarded by routers. Routers treat packets as messages, receiving (storing) them and then forwarding them based on how the message is addressed.

**Datagrams:** a packet that contains an absolute destination address; routers need only look up the destination address in a table to find the outgoing line to send the packet on its way.

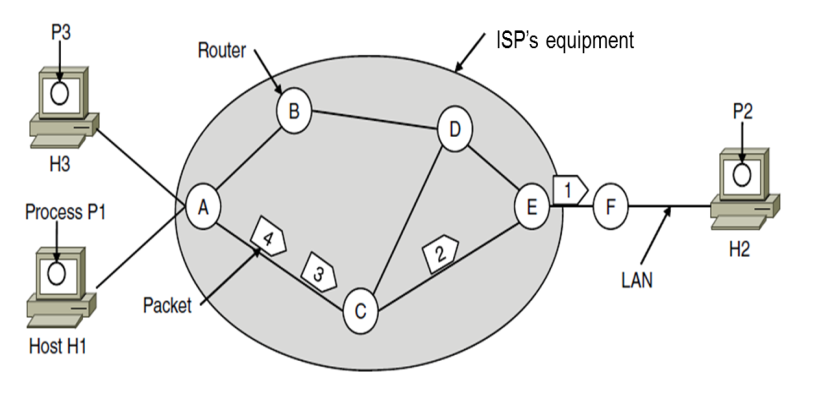
**Comparison of virtual-circuit and datagram networks:**





Packet is forwarded using destination address inside it

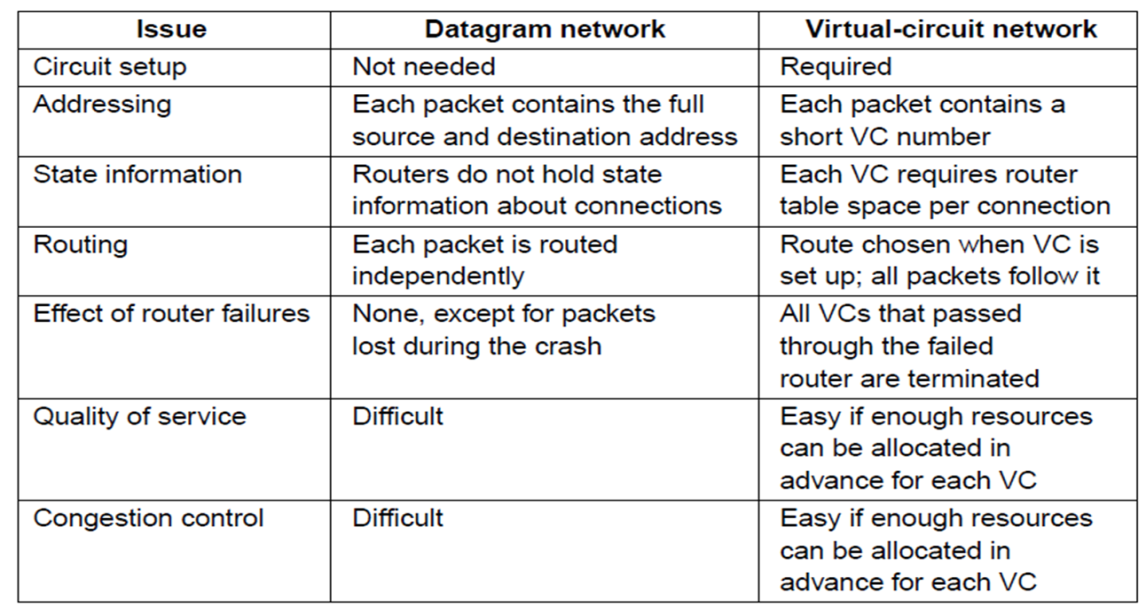
* Different packets may take different paths

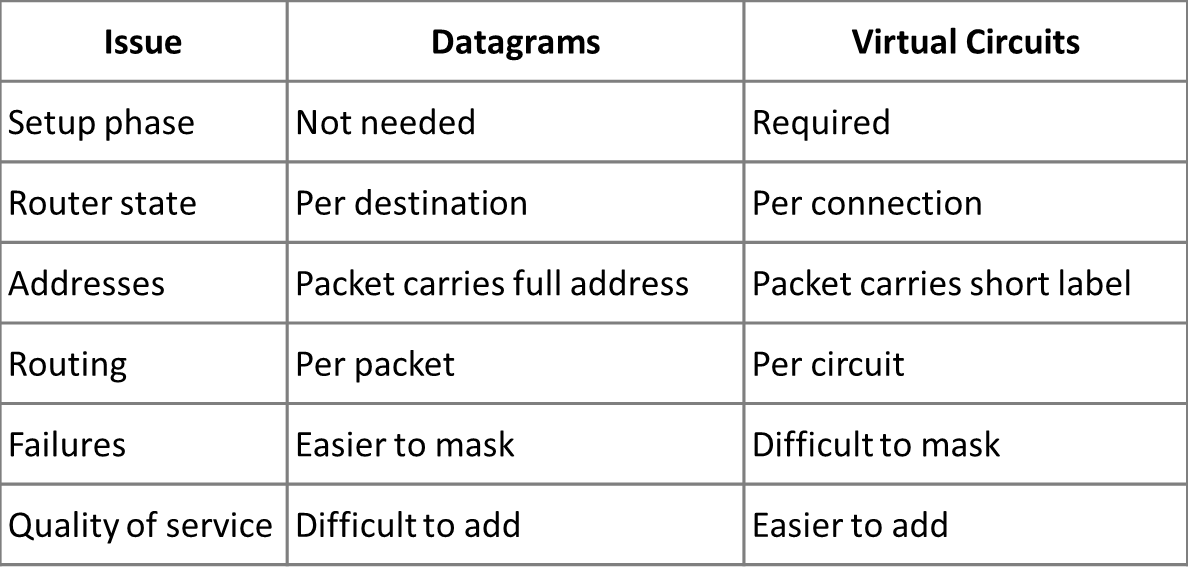




Packet is forwarded along a virtual circuit using tag inside it

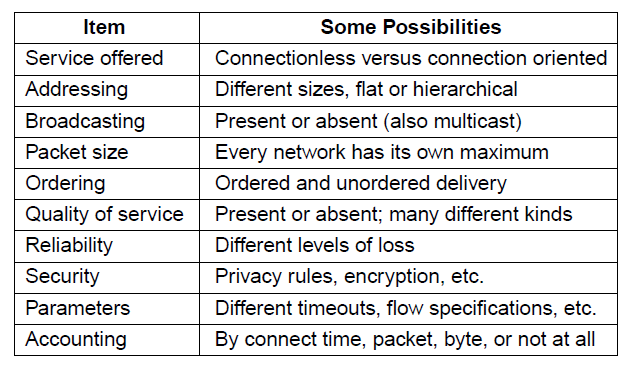
* Virtual circuit (VC) is set up ahead of time





**Internetworking**

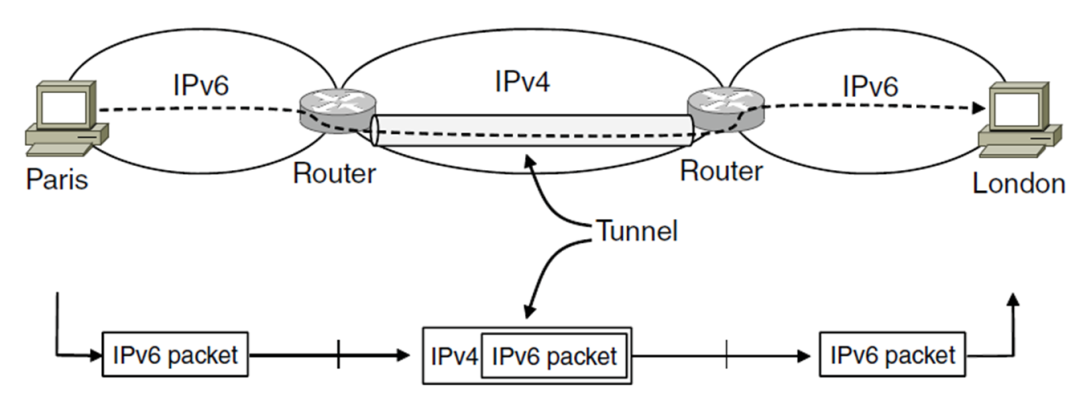
**How networks differ:**



**Tunneling:**

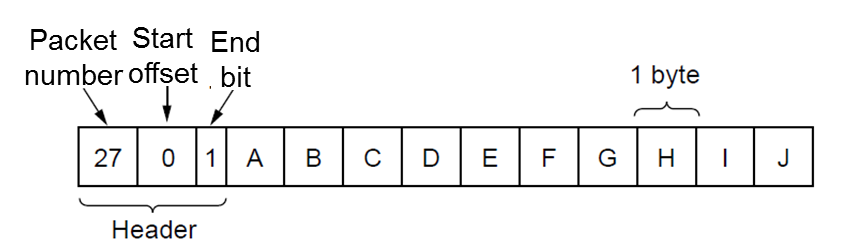
Connects two networks through a middle one

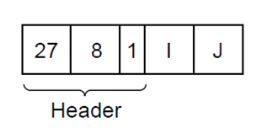
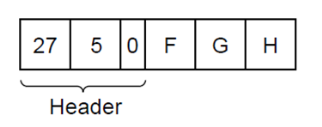
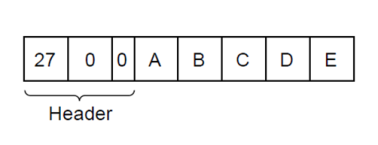
* Packets are encapsulates over the middle
* Tunnel carries IPv6 packets across IPv4 network

.

**Packet fragmentation:** Classic method, but dated. Routers fragment packets that are too large to forward. Receiving host reassembles to reduce load on routers. However it creates more work for routers, and hosts, it tends to magnify loss rate( If a fragment is lost, the whole packet is lost) and it creates security vulnerabilities also.

* Networks have different packet size limits for many reasons
* Different ways of fragmenting :
  + **Transparent** – packets fragmented / reassembled in each network
  + **Non-transparent** – fragments are reassembled at destination
* Header fields used to handle packet size differences





**IPv4 Fragmentation Procedure**

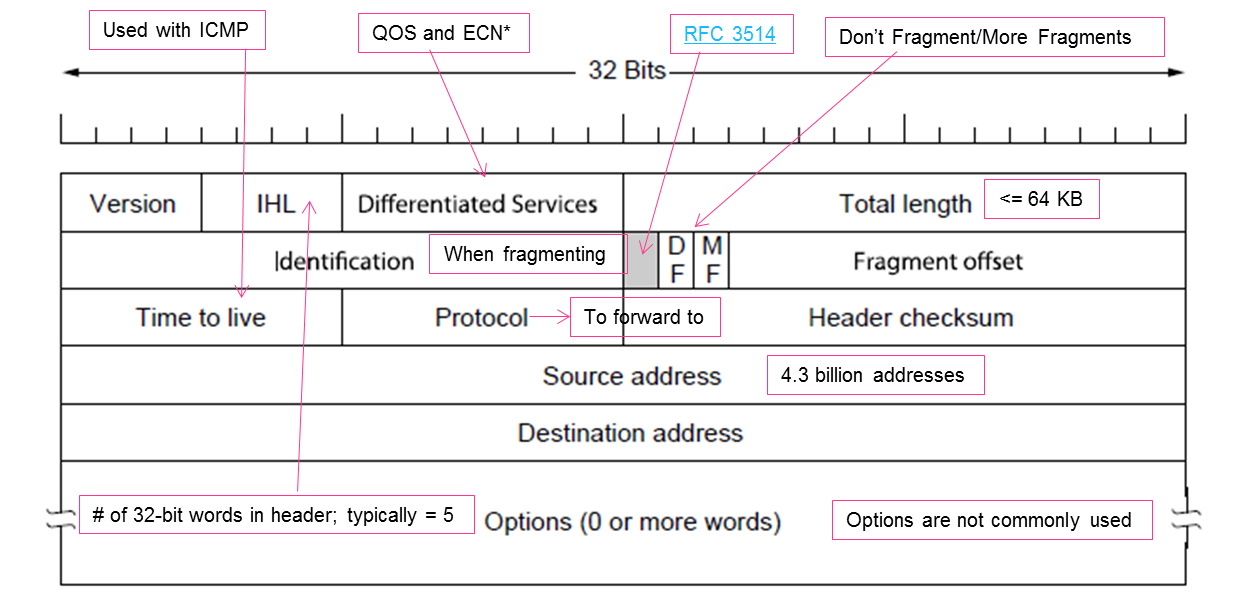
Routers split a packet that is too large:

* Typically break into large pieces
* Copy IP header to pieces
* Adjust length on pieces
* Set offset to indicate position
* Set MF (More Fragments) on all pieces except last

Receiving hosts reassembles pieces:

* Identification field links pieces together, MF tells receiver when it has all pieces

**The IP protocol:**

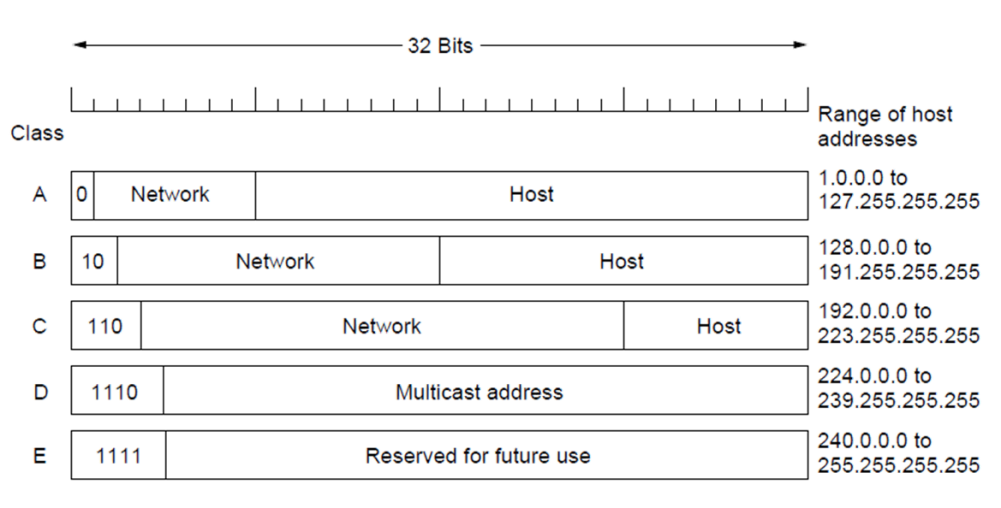
**IP header:** 

**Prefix:** IP addresses are described as consisting of two groups of bits in the address: the [most significant bits](http://en.wikipedia.org/wiki/Most_significant_bit) are the network address or network prefix, which identifies a whole network or subnet, and the [least significant](http://en.wikipedia.org/wiki/Least_significant_bit) set forms the host identifier, which specifies a particular interface of a host on that network. Prefix is determined by the network portion. Has 2L addresses aligned on 2L boundary.

**Classful and classless IP addresses:**

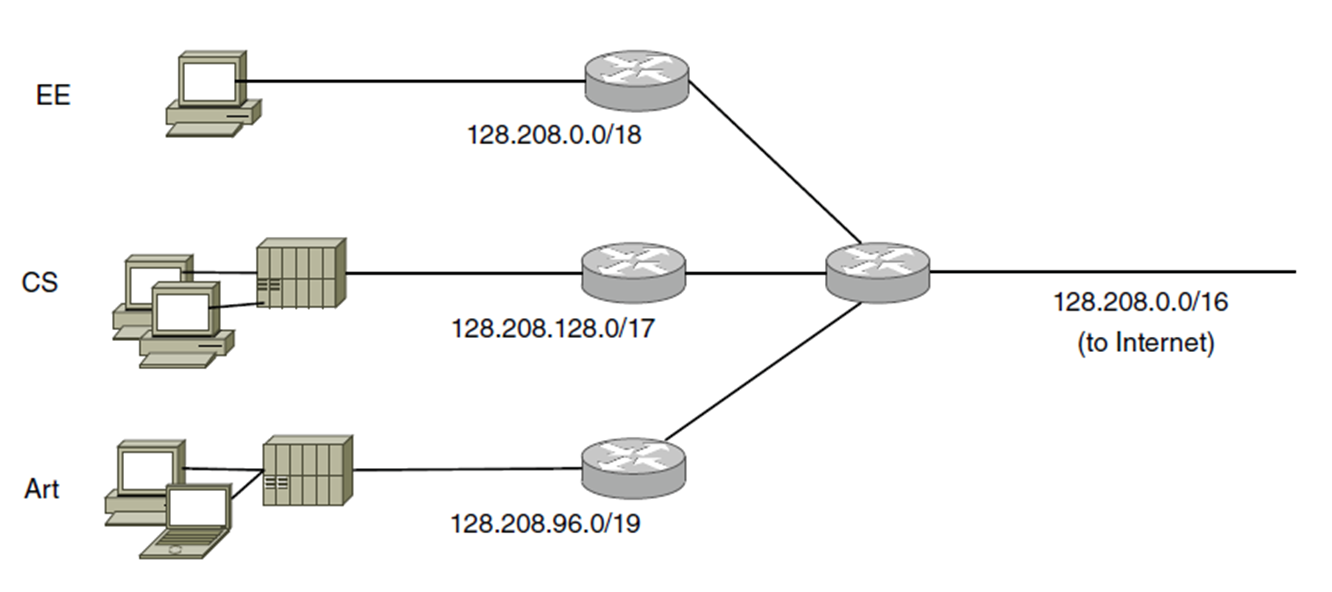
***Classful*** = addresses came in blocks of fixed size. Carries size as part of address, but lacks flexibility.

***Classless*** = allocates address space to [Internet service providers](http://en.wikipedia.org/wiki/Internet_service_provider) and end users on any address [bit](http://en.wikipedia.org/wiki/Bit) boundary.



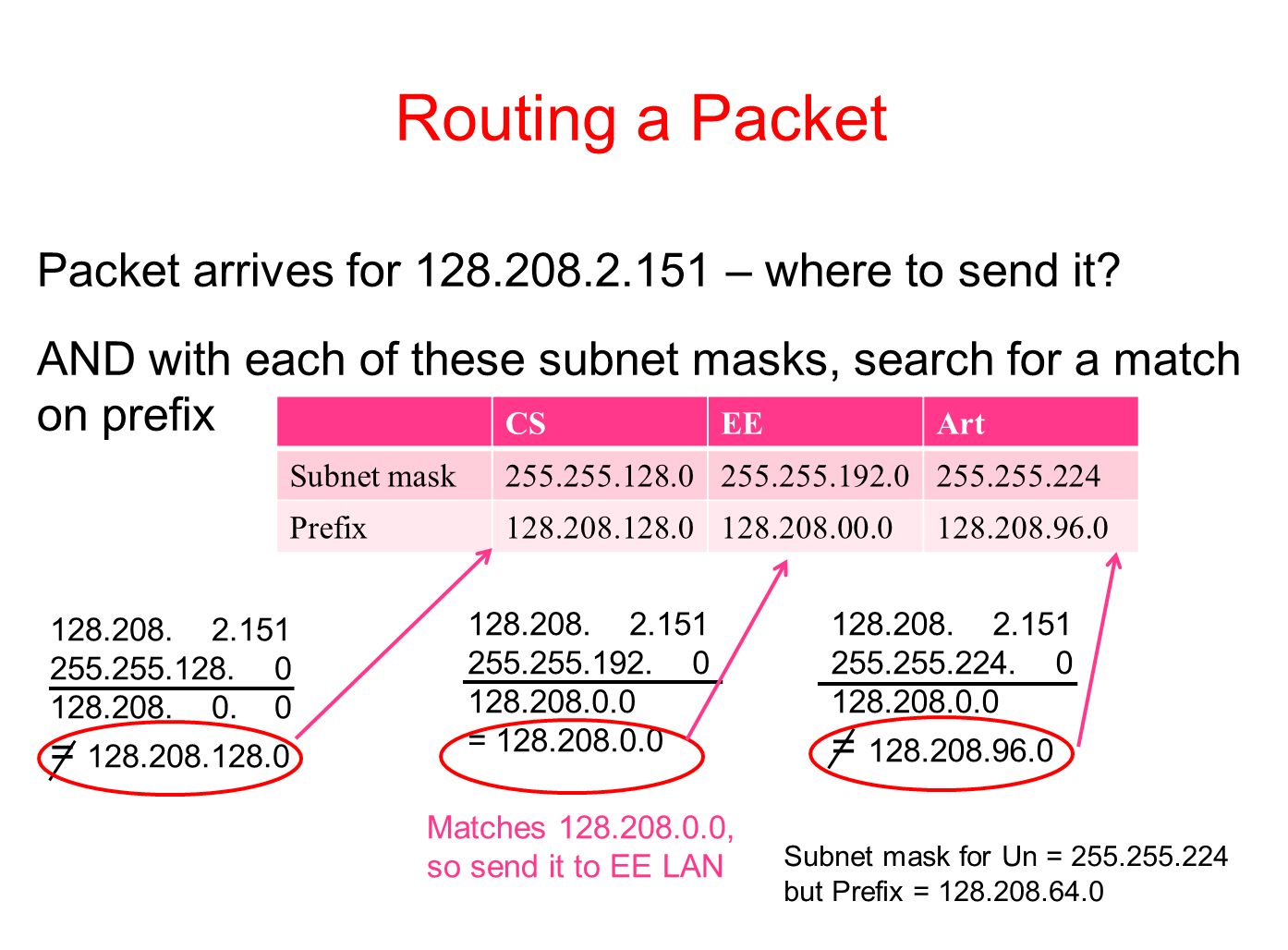
(A,B, and C are part of classful design. D and E are classless)

**Subnet and host portions of address:** A subnetwork, or subnet, is a logical, visible subdivision of an IP network. The practice of dividing a network into two or more networks is called subnetting. Computers that belong to a subnet are addressed with a common, identical, most-significant bit-group in their IP address. An IP address has two components, the network address and the [host](http://www.webopedia.com/TERM/H/host.html) address.



**Subnets and the subnet mask:** A mask used to determine what subnet an IP address belongs to. The subnet mask is the network address plus the bits reserved for identifying the subnetwork. It's called a [*mask*](http://www.webopedia.com/TERM/M/mask.html) because it can be used to identify the subnet to which an IP address belongs by performing a [bitwise](http://www.webopedia.com/TERM/B/bitwise_operator.html) [AND operation](http://www.webopedia.com/TERM/A/AND_operator.html) on the mask and the IP address.

**How packets are routed within a subnet:**



**Address aggregation:** Aggregation joins multiple IP prefixes into a single larger prefix to reduce routing table size. Same mechanism as subnets, just a different motivation (of reducing the size of routing tables instead of making it easier to use the block of addresses you have).

**Longest matching prefix:** Packets are forwarded to the entry with the longest matching prefix or smallest address block. This complicates forwarding but adds flexibility.

Can provide default behavior, with less specific prefixes

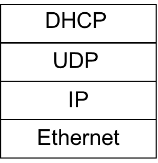
* + To send traffic going outside an organization to a border router

Can special case behavior, with more specific prefixes

* + For performance, economics, security

**IP works with the help of several control protocols:**

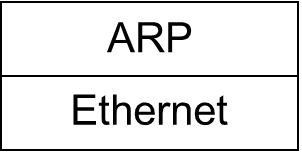
**DHCP:** (Dynamic Host Configuration Protocol) DHCP is a client-server application (uses UDP ports 67 (server), 68 (client)) and a protocol for automatically configuring addresses. It assigns a local IP address to a host.

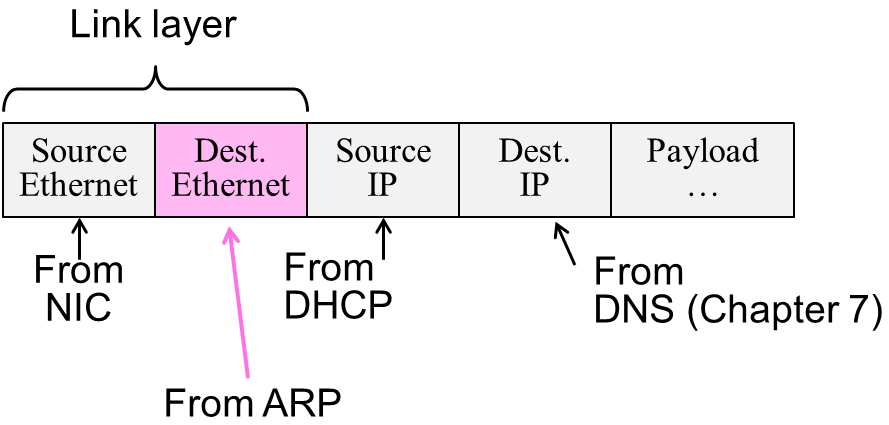


* + Gets host started by automatically configuring it
  + Host sends request to server, which grants a lease
  + Also gets gateway, DNS and NTP server addresses, subnet mask, etc.
  + Provides other parameters too
    - Network prefix
    - Address of local router
  + How does node send a message to DHCP server before it is configured?
    - Node sends broadcast messages delivered to all nodes on the network
    - Broadcast address is all 1s
      * IP (32 bit): 255.255.255.255
      * Ethernet (48 bit): ff:ff:ff:ff:ff:ff

**(ARP):** (The Address Resolution Protocol) ARP finds Ethernet address of a local IP address. It’s the glue that is needed to send any IP packets. Host queries an address and the owner replies.

* How do IP addresses get mapped into data link (Ethernet) addresses?
  + Broadcasts an ARP request saying “who owns IP address such-and-such?”
    - Broadcast also contains requester’s mapping, so all machines on subnet learn it
  + Owner responds with his MAC address
  + The original requester caches response
  + ARP is a layer-2 protocol.





**Network Address Translation (NAT):** maps one external IP address to many internal IP addresses.

It uses TCP/UDP port to tell connections apart. It also violates layering, very common in homes, etc.

A company having a true IP address of 210.20.30.00 uses Network Address Translation (NAT). If a machine sending a packet to be translated has a private IP address of 172.20.60.10, and is using source port 30000 (172.20.60.10:30000), and is communicating with a machine whose IP address is 140.20.30.6, with destination port address 13, what values would be found in the NAT table of the router for this flow (the next available index in the NAT box’s mapping table is 20; write your answer in the row for Index value 20 in the second table below). Assume that 140.20.30.6:13 uses port 13 as its source port when it responds.

Referring to the network diagram below, answer the following questions by filling in the table:

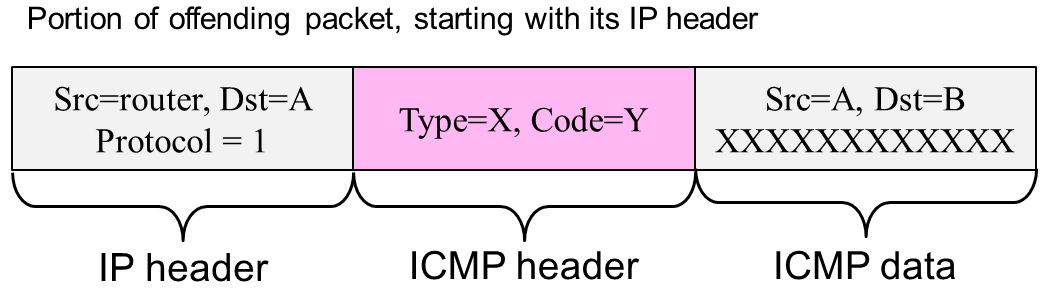
* 1. What would the IP source and destination addresses and ports be when a packet leaves 172.20.60.10 (callout 1, below)?
  2. What would these addresses and ports be when the packet leaves the NAT machine on the way to 140.20.30.6 (callout 2, below)?
  3. What would these addresses and ports be when the packet leaves 140.20.30.6 (callout 3, below) on the way to the NAT box?
  4. What would these addresses and ports be when the packet leaves the NAT box (callout 4, below) on the way back to 172.20.60.10?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Callout # | Source IP Address | Port | Destination IP Address | Port |
|  |  |  |  |  |
| 1 | 172.20.60.10 | 30000 | 140.20.30.6 | 13 |
| 2 | 210.20.30.00 | 20 | 140.20.30.6 | 13 |
| 3 | 140.20.30.6 | 13 | 210.20.30.00 | 20 |
| 4 | 140.20.30.6 | 13 | 172.20.60.10 | 30000 |



**ICMP:** (Internet Control Message Protocol) is a companion to IP that returns error info. They are implemented together and it is required and used in many ways, e.g., for traceroute. It sits on top of IP (IP Protocol=1)

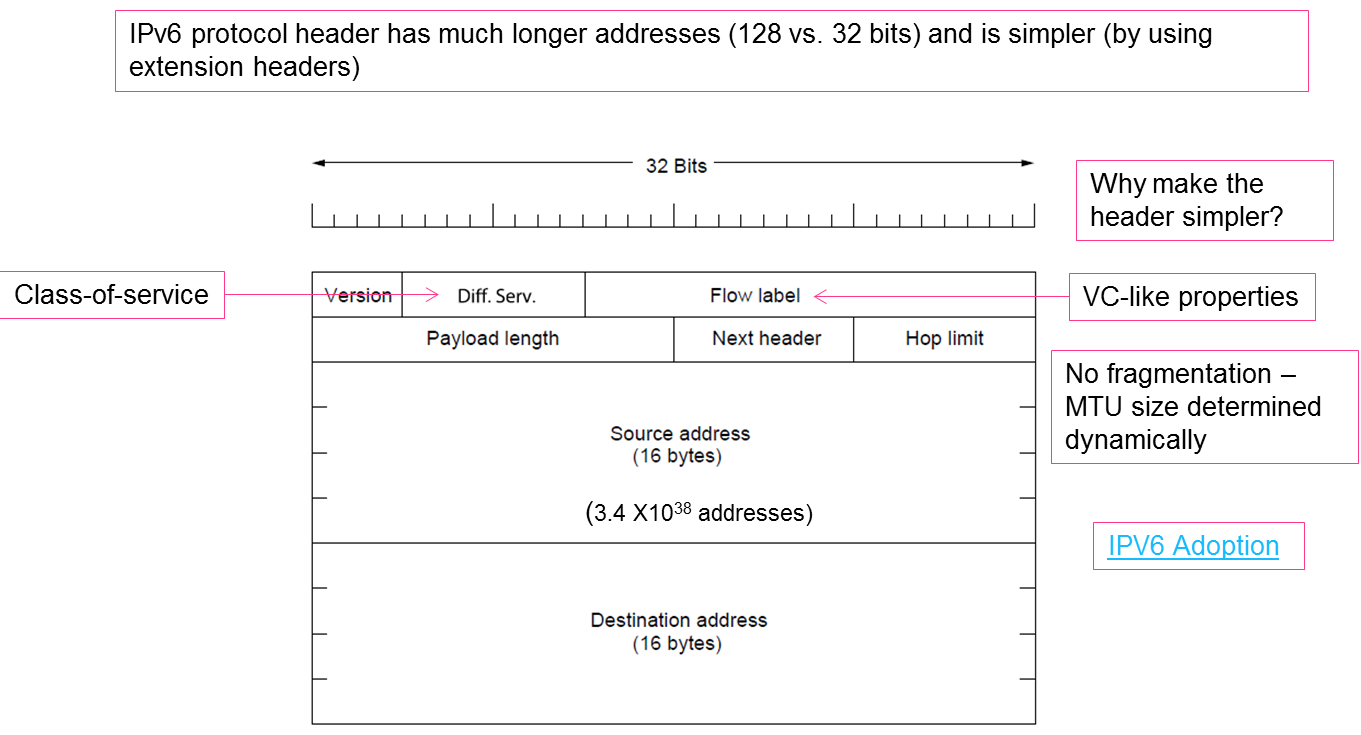
* + Provides error report and testing
    - When router encounters an error while forwarding:
      * It sends an ICMP error report back to the IP source address
      * It discards the problematic packet; host needs to rectify
  + Also testing that hosts can use
  + Each ICMP message has a Type, Code, and Checksum
    - Often carry the start of the offending packet as payload
    - Each message is carried in an IP packet

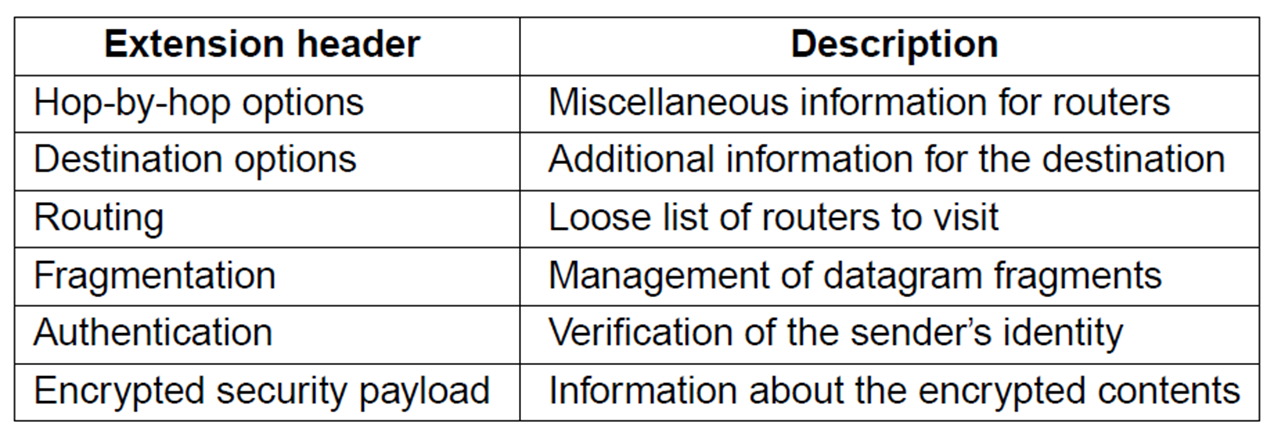


**Traceroute:** a computer network diagnostic tool for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network. The history of the route is recorded as the round-trip times of the packets received from each successive host (remote node) in the route (path); the sum of the mean times in each hop indicates the total time spent to establish the connection. Traceroute proceeds unless all (three) sent packets are lost more than twice, then the connection is lost and the route cannot be evaluated.

**IP V6 goals:**

* Much larger addresses (128 bits)
* Support billions of hosts
* Reduce routing table size
* Simplify protocol
* Better security
* Attention to type of service
* Aid multicasting
* Roaming host without changing address
* Allow future protocol evolution
* Permit coexistence of old, new protocols, …
* Lots of other, smaller changes
  + Streamlined header processing
  + Flow label to group of packets
  + Better fit with “advanced” features (mobility, multicasting, security)





**Distance Vector Routing (RIP):** is a distributed routing algorithm, the shortest path computation is split across nodes.

Algorithm: Each node

* knows distance of links to its neighbors
* advertises vector of lowest known distances to all destinations and shares with neighbors
* uses received vectors to update its own
* repeats periodically and on topology change; use Bellman-Ford algorithm to select route
* Uses UDP
* Time to converge and scalability poor
  + Reacts rapidly to good news and slowly to bad news
* Easily configured
* Tells its neighbors about the world (“routing by rumor”)

**The Count-to-Infinity Problem:** Failures can cause DV to “count to infinity” while seeking a path to an unreachable node. if A tells B that it has a path somewhere, there is no way for B to know if the path has B as a part of it. To see the problem clearly, imagine a subnet connected like A–B–C–D–E–F, and let the metric between the routers be "number of jumps". Now suppose that A is taken offline. In the vector-update-process B notices that the route to A, which was distance 1, is down – B does not receive the vector update from A. The problem is, B also gets an update from C, and C is still not aware of the fact that A is down – so it tells B that A is only two jumps from C (C to B to A), which is false. Since B doesn't know that the path from C to A is through itself (B), it updates its table with the new value "B to A = 2 + 1". Later on, B forwards the update to C and due to the fact that A is reachable through B (From C point of view), C decides to update its table to "C to A = 3 + 1". This slowly propagates through the network until it reaches infinity (in which case the algorithm corrects itself, due to the relaxation property of Bellman–Ford).

Partially addressed by:

(1) split horizon - if C routes through B to get to A then C does not advertise its route to A to B and

(2) poisoned reverse – if C routes through B to get to A then C advertises to B that its distance to A is infinity

**Link State Routing:** is an alternative to distance vector.

* Discover neighbors, learn network addresses through reachability protocol
  + - Hello packet and response
    - sent every 10 seconds as “keep alive”
* Set distance/cost metric to each neighbor
  + - May be bandwidth or delay
* Construct link state packet (advertisement) telling all it just learned about cost
  + LSP (Link State Packet) for a node lists neighbors and weights of links to reach them
* Periodically and when a change occurs, send packet to, receive packets from other routers.

**Comparison of Distance Vector Routing and Link State Routing:**

DV: router tells its neighbors about the world

LS: router tells the world about its neighbors

* More computation but simpler dynamics

**Hierarchical:** Hierarchical routing is what you think it is, e.g., to reach a given telephone first head towards the right country, then the right city in the country, then the phone in the city. Each node keeps only one entry per region for other regions, plus an entry for all nodes in the local region.

The advantages are smaller routing tables, smaller routing computations to run at nodes, and fewer/smaller messages to send to describe the network but may result in slightly longer paths than flat routing.

**Comparison of routing and bridging**

|  |  |
| --- | --- |
| **Router** | **Bridge** |
| * Router uses layer-3 hierarchical addresses | * Bridge uses layer-2 flat addresses |
| * Table size proportional to number of subnets | * Table size proportional to number of end-systems |
| * End-system forwards data packet to router using router MAC address (must know address of gateway) | * Bridge sees and processes each frame; end-station doesn’t see bridge |
| * Broadcasts not forwarded (to avoid stressing the WAN) | * Broadcasts flooded to maintain transparency |
| * Independent of layer 2 | * Dependent on layer 2 |
| * Router is “gateway” to WAN | * LANs not coupled to WANs because of high traffic (b’cast domain) |
| * Can use redundant paths | * Uses spanning tree to avoid loops |
| * Router separates the network into multiple broadcast domains | * Bridge separates a LAN into multiple collision domains but still 1 b’dcast domain |
| * Broadcast frames stopped at router | * Broadcast frames flooded |
| * Slower (processing in software) but trend is to layer-3 switching | * Faster (processing in hardware) |
| * Address resolution (ARP) needed | * No address resolution |
| * Location change requires adjustment of layer 3 address (but can use DHCP) | * Location change does not require adjustment of address |

**Broadcast:** To all.

**Multicast and anycast routing:**

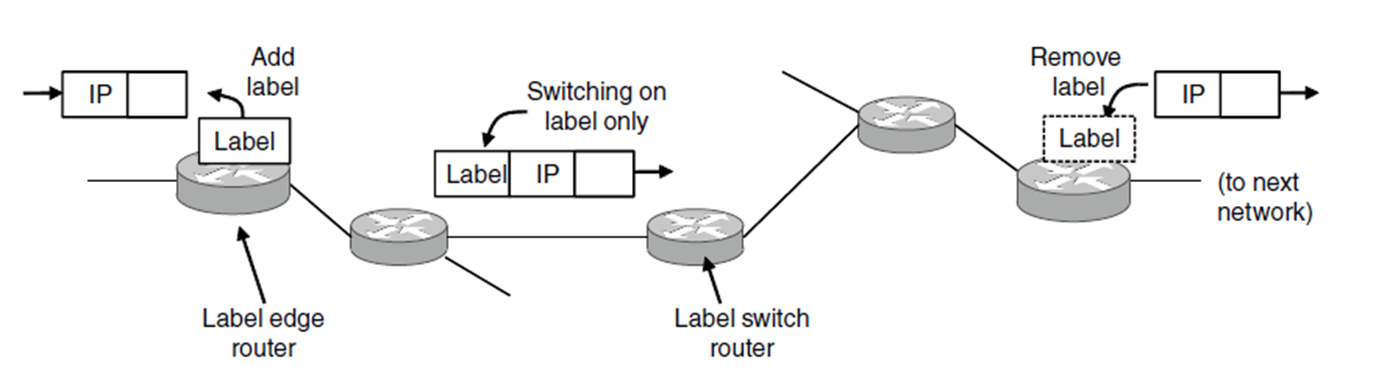
*Multicast* – sends to a subset of the nodes called a group. Uses a different tree for each group and source

*Anycast* – a packet is delivered to the nearest member of a group

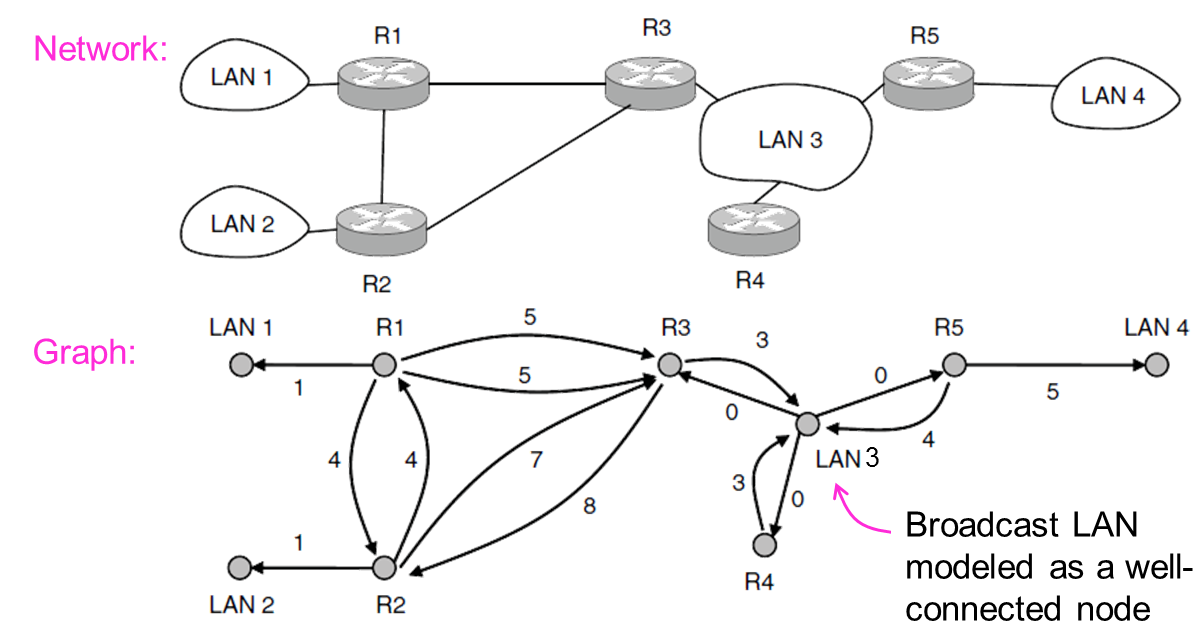
**Label switching and MPLS:**

MPLS (Multi-Protocol Label Switching) sends packets along established paths; ISPs can use for QoS

* Path indicated with label below the IP layer
* Uses label as index into a table -- fast lookup



**OSPF:** computes routes for a single network (e.g., ISP). Models network as a graph of weighted edges. OSPF divides one large network (Autonomous System) into areas connected to a backbone area. This helps to scale; summaries go over area borders.



**Border Gateway Protocol (BGP):** computes routes across interconnected, autonomous networks. Its key role is to respect networks’ policy constraints. It is a distance vector (path vector) protocol.

Example policy constraints:

* No commercial traffic for educational network
* Never put Iraq on route starting at Pentagon
* Choose cheaper network
* Choose better performing network
* Don’t go from Apple to Google to Apple

**Session routing:**

**Non-adaptive (static) and adaptive algorithms:**

**Interior and exterior protocols:**

**Autonomous systems:**