Network Layer

October 21, 2019

Goals

- Getting packets from the source all the way to the destination.
- Getting to the destination may require making many hops at intermediate routers along the way.

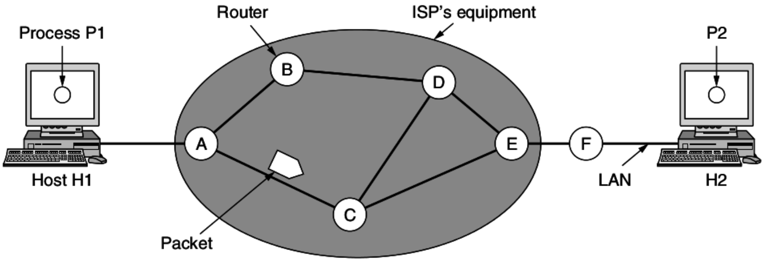
How to achieve this?

- To achieve its goals, N/W layer must know about the topology of the network (i.e., the set of all routers and links) and choose appropriate paths through it.
- Routes chosen should avoid overloading some of the communication lines and routers while leaving others idle.
- Finally, when the source and destination are in different networks, new problems occur. It is up to the network layer to deal with them.

Design Issues

- Basically we are concerned with design of network layer and how/what services are provided to transport layer.

Store and Forward Packet Switching



- A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the ISP.
- The packet is stored there until it has fully arrived and the link has finished its processing by verifying the checksum.
- Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered.

Services provided to Transport Layer

- The network layer provides services to the transport layer at the network layer/transport layer interface.
- Question/Dilemma: Should N/W layer provide connection oriented service or connectionless service.

Connection less service

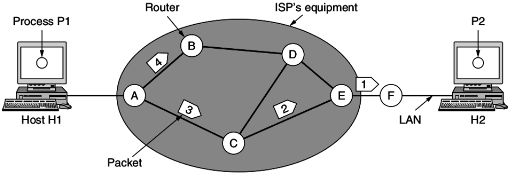
- Argument: Routers' job is moving packets around and nothing else.
- The network is inherently unreliable, no matter how it is designed. Therefore, the hosts should accept this fact and do error control (i.e., error detection and correction) and flow control themselves.
- Should comprise SEND PACKET and RECEIVE PACKET primitives, and each packet must carry the full destination address, because each packet sent is carried independently of its predecessors, if any.

Connection oriented service

- Argument: Success of telephone network. Reliable and provides QoS.
- X.25, Frame Relay (CO): ARPANET, Internet (CL).

Implementation of Connection Less Service

- Packets are injected into the network individually and routed independently of each other. No advance setup is needed.
- Packets are frequently called datagrams (in analogy with telegrams) and the network is called a datagram network.
- The algorithm that manages the tables and makes the routing decisions is called the routing algorithm.



A's table (initially)

Dest.	Line
A	–
B	B
C	C
D	B
E	C
F	C

A's table (later)

A	–
B	B
C	C
D	B
E	B
F	B

C's table

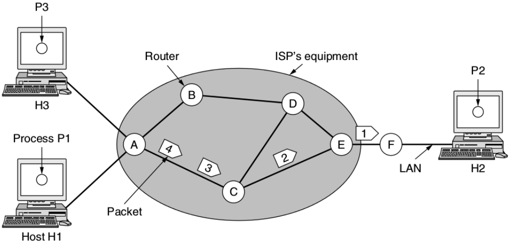
A	A
B	A
C	–
D	E
E	E
F	E

E's table

A	C
B	D
C	C
D	D
E	–
F	F

Implementation of Connection Oriented Service

- Path (VC) from the source router to destination router must be established before any data packets can be sent. Each packet carries an identifier telling which virtual circuit it belongs to.
- *A* assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets.



H1	1	C	1	A	1	E	1	C	1	F	1
H3	1	C	2	A	2	E	2	C	2	F	2

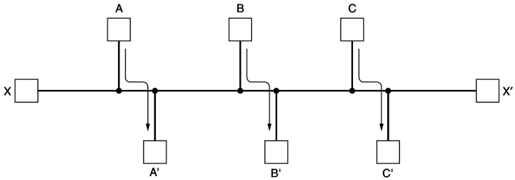
VC & Datagram Networks

- Resources (e.g., buffers, band-width, and CPU cycles) can be reserved in advance, when the connection is established.

Issue	Datagram network	Virtual-circuit network
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Routing Algorithms

- Routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on.
- One can think of a router as having two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing tables. This process is forwarding.
- The other process is responsible for filling in and updating the routing tables.
- Desirable properties of a routing algorithm: correctness, simplicity, robustness, stability, fairness, and efficiency.



Classes of Routing Algorithms

- Non-adaptive: Routes are computed in advance, offline, and downloaded to the routers when the network is booted(static routing).
- Adaptive algorithms: Change their routing decisions to reflect changes in the topology, and sometimes changes in the traffic as well.
- Dynamic routing algorithms differ in where they get their information, when they change the routes, and what metric is used for optimization

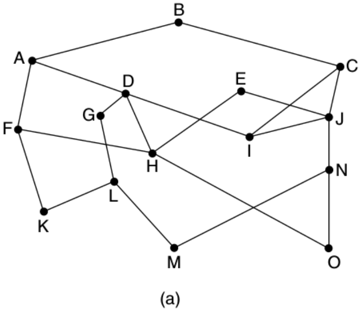
Optimality Principle

- It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.

Optimality Principle

- It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.
- To see this, call the part of the route from I to J r_1 and the rest of the route r_2.
- If a route better than r_2 existed from J to K, it could be concatenated with r_1 to improve the route from I to K, contradicting our statement that r_1 r_2 is optimal.

Optimality Principle



- Based on optimality principle, the set of optimal routes from all sources to a given destination form a tree rooted at the destination **(sink tree)** where the distance metric is the number of hops.
- The goal of all routing algorithms is to discover and use the sink trees for all routers.

Optimality Principle

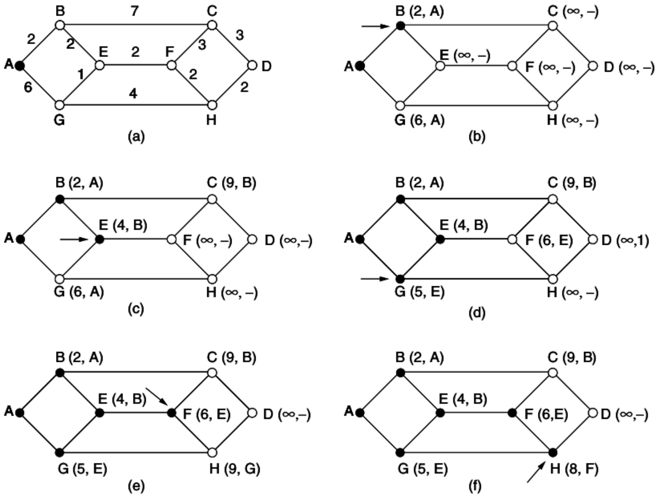
- A sink tree is not necessarily unique; other trees with the same path lengths may exist.
- Since a sink tree is indeed a tree, it does not contain any loops, so each packet will be delivered within a finite and bounded number of hops.

Shortest Path Algorithm

Let us compute optimal paths given a complete picture of the network.

- Dijkstra's algorithm (1959) finds the shortest paths between a source and all destinations in the network. Each node is labelled with its distance from the source node along the best known path.
- Initially, no paths are known, so all nodes are labelled with infinity.
- As the algorithm proceeds and paths are found, the labels may change, reflecting better paths.
- A label may be either tentative or permanent. Initially, all labels are tentative.
- When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

Shortest Path Algorithm



Flooding

When a routing algorithm is implemented, each router must make decisions based on local knowledge, not the complete picture of the network.

- Flooding is the technique in which every incoming packet is sent out on every outgoing line except the one it arrived on.
- Flooding generates vast numbers of duplicate packets, in fact, an infinite number that can be limited by using hop-counter.
- A better technique for damming the flood is to have routers keep track of which packets have been flooded, to avoid sending them out a second time.

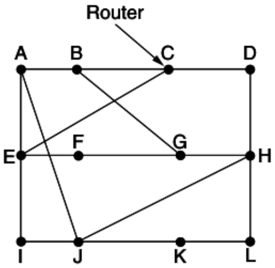
Distance Vector Routing

A distance vector routing algorithm operates by having each router maintain a table (i.e., a vector) giving the best known distance to each destination and which link to use to get there.

- These tables are updated by exchanging information with the neighbors. Eventually, every router knows the best link to reach each destination.
- It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP.

DV Routing

Suppose that J has measured or estimated its delay to its neighbors, A, I, H, and K, as 8, 10, 12, and 6 msec, respectively.



To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

Count to Infinity Problem

- The settling of routes to best paths across the network is called convergence.
- DV reacts rapidly to good news, but leisurely to bad news.

A	B	C	D	E	
	•	•	•	•	Initially
	1	•	•	•	After 1 exchange
	1	2	•	•	After 2 exchanges
	1	2	3	•	After 3 exchanges
	1	2	3	4	After 4 exchanges

A	B	C	D	E	
	1	2	3	4	Initially
	3	2	3	4	After 1 exchange
	3	4	3	4	After 2 exchanges
	5	4	5	4	After 3 exchanges
	5	6	5	6	After 4 exchanges
	7	6	7	6	After 5 exchanges
	7	8	7	8	After 6 exchanges
		⋮			
	•	•	•	•	

Link State Routing

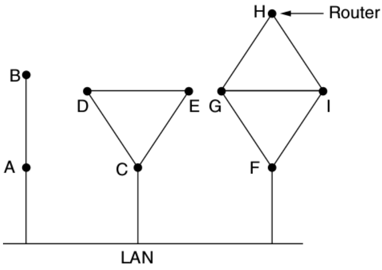
The idea behind link state routing is fairly simple and can be stated as five parts. Each router must do the following things to make it work:

- Discover its neighbors and learn their network addresses.
- Set the distance or cost metric to each of its neighbors.
- Construct a packet telling all it has just learned.
- Send this packet to and receive packets from all other routers.
- Compute the shortest path to every other router.

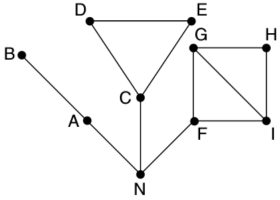
In effect, the complete topology is distributed to every router. Then Dijkstra's algorithm can be run at each router to find the shortest path to every other router.

Link State Routing: Discovering Neighbors

- It accomplishes this goal by sending a special HELLO packet on each point-to-point line. The router on the other end is expected to send back a reply giving its name.
- The router names must be globally unique.

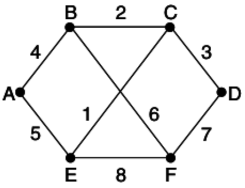


(a)



(b)

Link State Routing: Building Link State Packets



(a)

Link State Packets

A		B		C		D		E		F	
Seq.		Seq.		Seq.		Seq.		Seq.		Seq.	
Age		Age		Age		Age		Age		Age	
B	4	A	4	B	2	C	3	A	5	B	6
E	5	C	2	D	3	F	7	C	1	D	7
		F	6	E	1			F	8	E	8

(b)

Link State Routing: Distributing Link State Packets

The fundamental idea is to use flooding to distribute the link state packets to all routers. To keep the flood in check, each packet contains a sequence number that is incremented for each new packet sent.

The Age field is decremented by each router during the initial flooding process, to make sure no packet can get lost and live for an indefinite period of time.

			Send flags			ACK flags			
Source	Seq.	Age	A	C	F	A	C	F	Data
A	21	60	0	1	1	1	0	0	
F	21	60	1	1	0	0	0	1	
E	21	59	0	1	0	1	0	1	
C	20	60	1	0	1	0	1	0	
D	21	59	1	0	0	0	1	1	

Figure: The packet buffer for router B

Link State Routing: Computing Routes

Once a router has accumulated a full set of link state packets, it can construct the entire network graph because every link is represented.

Dijkstra's algorithm can be run locally to construct the shortest paths to all possible destinations.

OSPF and IS-IS are some examples.

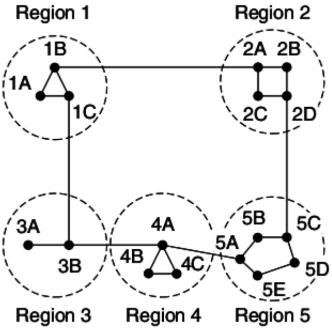
Hierarchical Routing

Can you guess why hierarchical routing?

Hierarchical Routing

- As networks grow in size, the router routing tables grow proportionally.
- More router memory and more CPU time and more bandwidth is needed to send status reports about them.
- At a certain point, the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network.

Hierarchical Routing



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

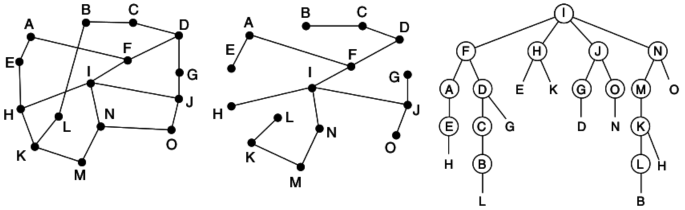
Broadcast Routing

- Sending a packet to all destinations simultaneously is called broadcasting.
- Simple **broadcast** and **multi-destination** broadcasting are naive techniques.
- Flooding is one of the better broadcasting techniques. When implemented with a sequence number per source, flooding uses links efficiently with a decision rule at routers that is relatively simple.
- However, we can do better still once the shortest path routes for regular packets have been computed (**Reverse path forwarding:**)

Reverse path forwarding:

- When a broadcast packet arrives at a router, the router checks to see if the packet arrived on the link that is **normally used for sending packets toward the source of the broadcast.**
- If so, there is an excellent chance that the **broadcast packet itself followed the best route from the router** and is therefore the first copy to arrive at the router.
- This being the case, the router forwards copies of it onto all links except the one it arrived on.
- If, however, the broadcast packet arrived on a link other than the preferred one for reaching the source, the packet is discarded as a likely duplicate.

Reverse Path Forwarding



(a) (b) (c)

Reverse path forwarding:

- The principal advantage of reverse path forwarding is that it is efficient while **being easy to implement**.
- It sends the broadcast packet over each link only once in each direction, **just as in flooding**, yet it requires only that routers know how to reach all destinations, without needing to remember sequence numbers (or use other mechanisms to stop the flood) or list all destinations in the packet.

Spanning Tree:

- A spanning tree is a subset of the network that includes all the routers but contains no loops. Sink trees are spanning trees.
- If each router knows which of **its lines belong to the spanning tree**, it can copy an incoming broadcast packet onto all the spanning tree lines except the one it arrived on.
- This method makes excellent use of bandwidth, generating the absolute minimum number of packets necessary to do the job.

Multicast Routing

- Some applications, such as a multiplayer game or live video of a sports event streamed to many viewing locations, send packets to multiple receivers.
- Unless the group is very small, **sending a distinct packet** to each receiver is **expensive.**
- On the other hand, broadcasting a packet is wasteful if the group consists of, say, **1000 machines on a million-node network**, so that most receivers are not interested in the message (or they are not supposed to see it).

Multicast Routing

- Sending a message to a group is called multicasting, and the routing algorithm used is called multicast routing.
- All multicasting schemes require some way to **create and destroy groups** and to **identify which routers are members** of a group.
- For now, we will assume that **each group is identified by a multicast address** and that **routers know the groups to which they belong.**

Multicast Routing

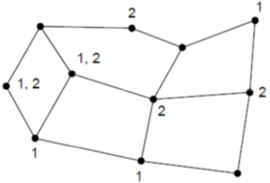
- Multicast routing schemes build on the broadcast routing schemes, sending packets along spanning trees to deliver the packets to the members of the group while making efficient use of bandwidth.
- However, the best spanning tree to use depends on whether the group is dense, with receivers scattered over most of the network, or sparse, with much of the network not belonging to the group.

Multicast Routing

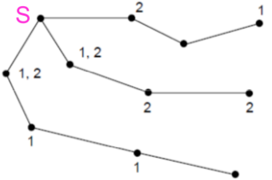
- If the group is dense, broadcast is a good start because it efficiently gets the packet to all parts of the network.
- But broadcast will reach some routers that are not members of the group, which is wasteful.
- The solution explored by **Deering and Cheriton (1990)** is to **prune the broadcast spanning tree** by removing links that do not lead to members.
- The result is an efficient multicast spanning tree.

Multicast sends to a subset of the nodes called a group

- Uses a different tree for each group and source

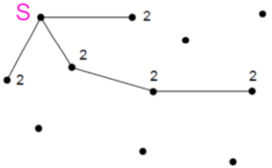


Network with groups 1 & 2



Spanning tree from source S

Multicast tree from S to group 1

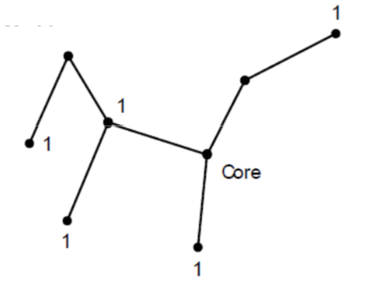


Multicast tree from S to group 2

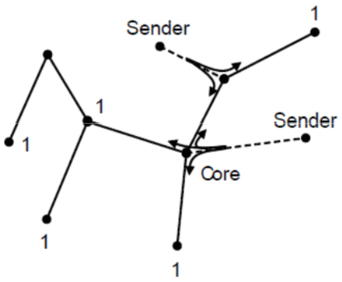
Multicast Routing

CBT (Core-Based Tree) uses a single tree to multicast

- Tree is the sink tree from core node to group members
- Multicast heads to the core until it reaches the CBT



Sink tree from core to group 1

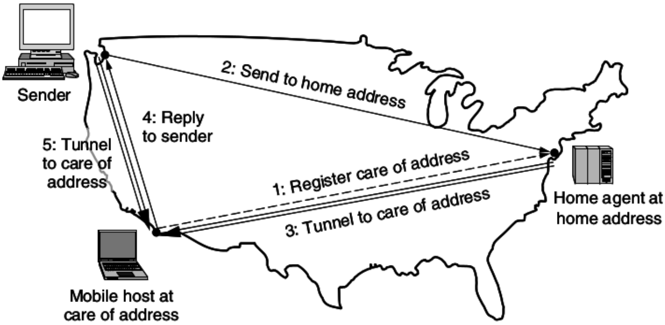


Multicast is send to the core then down when it reaches the sink tree

Routing for Mobile Hosts

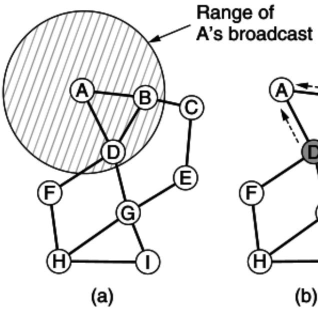
Mobile hosts introduce a new complication: to route a packet to a mobile host, the network first has to find it.

Assumption: all hosts are assumed to have a permanent home location that never changes.



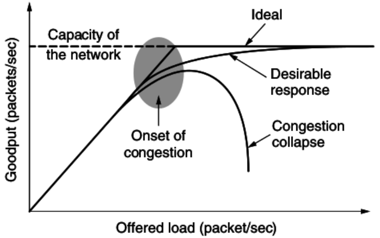
Routing in Ad-hoc Networks

Routers themselves are mobile



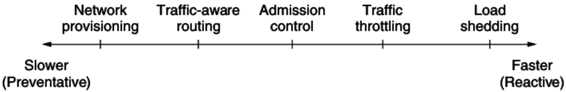
Congestion Control Algorithms

- Too many packets present in (a part of) the network causes packet delay and loss that degrades performance. This situation is called congestion.
- The network and transport layers share the responsibility for handling congestion.
- The most effective way to control congestion is to reduce the load that the transport layer is placing on the network.



Approaches to control congestion

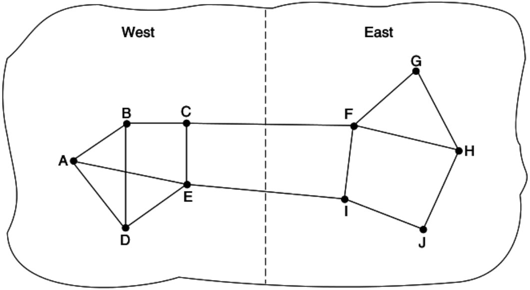
Two solutions come to mind: increase the resources or decrease the load.



Traffic Aware Routing

Routing based on load

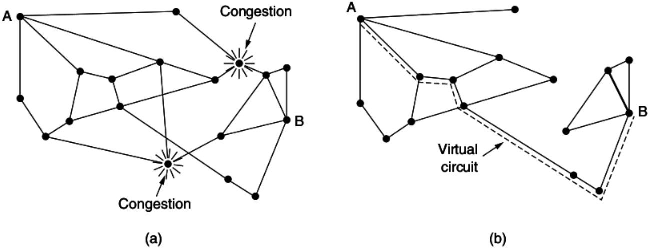
- But frequent oscillation of routes
- Multipath routes
- The routing scheme to shift traffic across routes slowly enough that it is able to converge.



Admission Control

Do not set up a new virtual circuit unless the network can carry the added traffic without becoming congested.

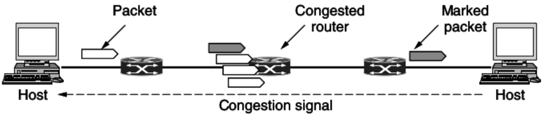
- Admission control can also be combined with traffic-aware routing by considering routes around traffic hotspots as part of the setup procedure.



Traffic Throttling

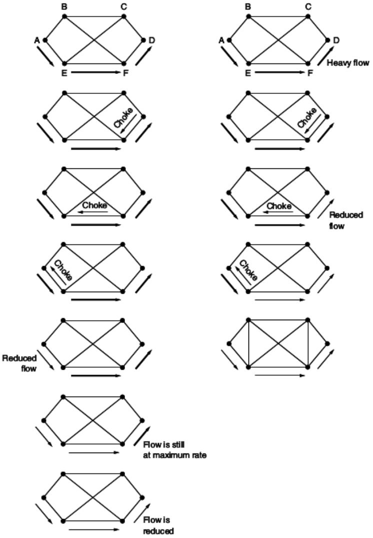
When congestion is imminent, the senders throttle back their transmissions and slow down. How to know?

- Each router can continuously monitor the resources it is using.
- The buffering of queued packets inside the router, and the number of packets that are lost due to insufficient buffering.
- Choke packets, explicit congestion notification



Traffic Throttling

Hop-by-hop back-pressure



Load Shedding

When routers can't take any more, they just throw packets away.

- What packets to drop?

Random Early Detection

The only reliable indication of congestion that hosts get from the network is packet loss.

- To determine when to start discarding, routers maintain a running average of their queue lengths. When the average queue length on some link exceeds a threshold, the link is said to be congested and a small fraction of the packets are dropped at random.
- How many many packets should be dropped?

QoS

Applications demand stronger performance guarantees from the network than "the best that could be done under the circumstances.

- Multimedia applications in particular, often need a minimum throughput and maximum latency to work.
- One way is over provisioning.

QoS

Four issues must be addressed to ensure quality of service:

- What applications need from the network.
- How to regulate the traffic that enters the network.
- How to reserve resources at routers to guarantee performance.
- Whether the network can safely accept more traffic.

QoS: Application Requirements

A stream of packets from a source to a destination is called a flow.

- The needs of each flow can be characterized by four primary parameters: bandwidth, delay, jitter, and loss.

QoS: Application Requirements

Different applications care about different properties

- We want all applications to get what they need

Application	Bandwidth	Delay	Jitter	Loss
Email	Low	Low	Low	Medium
File sharing	High	Low	Low	Medium
Web access	Medium	Medium	Low	Medium
Remote login	Low	Medium	Medium	Medium
Audio on demand	Low	Low	High	Low
Video on demand	High	Low	High	Low
Telephony	Low	High	High	Low
Videoconferencing	High	High	High	Low

"High" means a demanding requirement, e.g., low delay

Traffic Shaping

- Traffic shaping is a technique for regulating the average rate and burstiness of a flow of data that enters the network.
- In the telephone network, this characterization is simple. For example, a voice call needs 64 kbps and consists of one 8-bit sample every 125 μ sec.

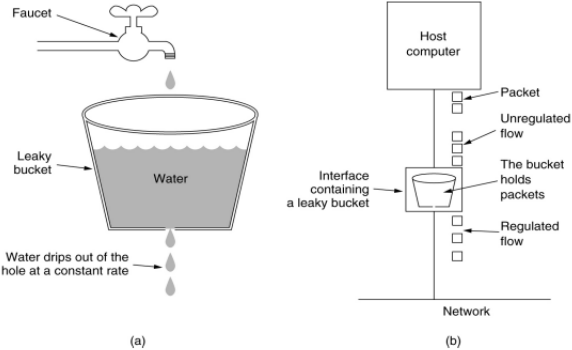
Traffic Shaping

Traffic Shaping

- It is about regulating average rate of data flow.
- It is a method of congestion control by providing shape to data flow before entering the packet into the network.
- At connection set-up time, the sender and carrier negotiate a traffic pattern (shape)
- There are two types of Traffic shaping algorithm :-
  - 1. **Leaky Bucket Algorithm.**
  - 2. **Token Bucket Algorithm**

Leaky Bucket Algorithm

- The **Leaky Bucket Algorithm** used to control rate in a network.
- It is implemented as a single-server queue with constant service time.
- If the bucket (buffer) overflows then packets are discarded.
- In this algorithm the input rate can vary but the output rate remains constant.
- This algorithm saves busty traffic into fixed rate traffic by averaging the data rate.



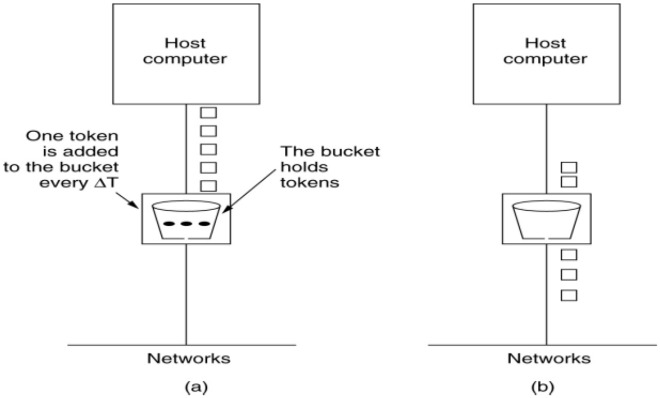
(a) A leaky bucket with water.

(b) a leaky bucket with packets.

Token Bucket Algorithm

- The **Token Bucket Algorithm** compare to Leaky Bucket Algorithm allow the output rate vary depending on the size of burst.
- In this algorithm the buckets holds token to transmit a packet, the host must capture and destroy one token.
- Tokens are generated by a clock at the rate of one token every Δt sec.
- Idle hosts can capture and save up tokens (up to the max. size of the bucket) in order to send larger bursts later.

Token Bucket Algorithm



(a) Before (b) After

Leaky/Token Buckets

- Leaky and token buckets limit the long-term rate of a flow but allow short- term bursts up to a maximum regulated length to pass through unaltered and without suffering any artificial delays.
- Large bursts will be smoothed by a leaky bucket traffic shaper to reduce congestion in the network.

Token Bucket Algorithm

A computer on a 6-Mbps network is regulated by a token bucket. The token bucket is filled at a rate of 1 Mbps. It is initially filled to capacity with 8 megabits. How long can the computer transmit at the full 6 Mbps?

Leaky/Token Buckets

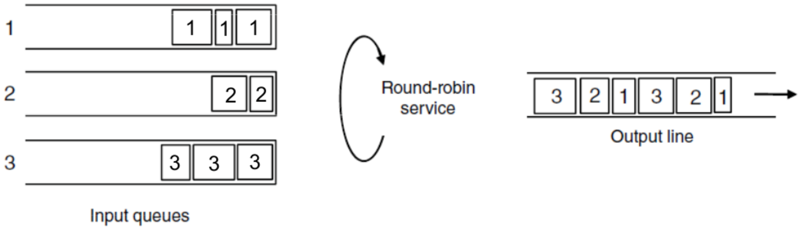
If we call the burst length $\mathbf{S}$ sec., the maximum output rate $\mathbf{M}$ bytes/sec, the token bucket capacity $\mathbf{B}$ bytes, and the token arrival rate $\mathbf{R}$ bytes/sec, we can see that an output burst contains a maximum of $\mathbf{B} + \mathbf{RS}$ bytes. We also know that the number of bytes in a maximum speed burst of length S seconds is MS. Hence, we have $\mathbf{B} + \mathbf{RS} = \mathbf{MS}$.

Packet Scheduling

- Using traffic shaping, we are able to regulate the shape of the offered traffic.
- However, to provide a performance guarantee, we must reserve sufficient resources along the route that the packets take through the network.
- Algorithms that allocate router resources among the packets of a flow and between competing flows are called packet scheduling algorithms.
- 3 different kinds of resources can potentially be reserved for different flows: 1. Bandwidth. 2. Buffer space. 3. CPU cycles.

Packet Scheduling

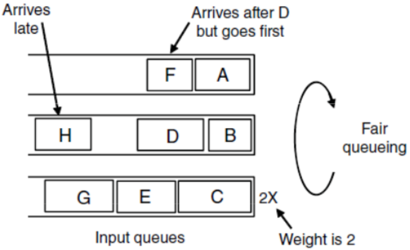
Packet scheduling divides router/link resources among traffic flows with alternatives to FIFO (First In First Out)



Example of round-robin queuing

Fair Queueing approximates bit-level fairness with different packet sizes; weights change target levels

- Result is WFQ (Weighted Fair Queueing)



Packets may be sent out of arrival order

Packet	Arrival time	Length	Finish time	Output order
A	0	8	8	1
B	5	6	11	3
C	5	10	10	2
D	8	9	20	7
E	8	8	14	4
F	10	6	16	5
G	11	10	19	6
H	20	8	28	8

$$F_i = \max(A_i, F_{i-1}) + L_i/W$$

Finish virtual times determine transmission order

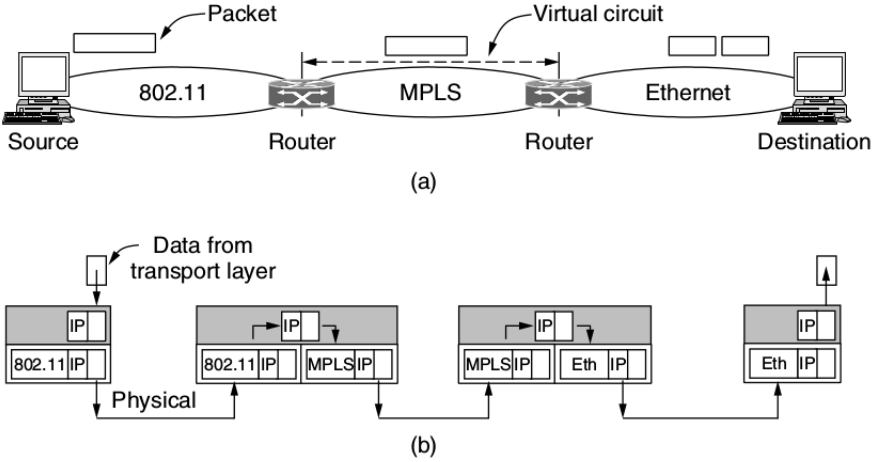
Internetworking

- What issues arise when two or more networks are connected to form an internetwork, or more simply an internet?

How networks differ?

Item	Some Possibilities
Service offered	Connectionless versus connection oriented
Addressing	Different sizes, flat or hierarchical
Broadcasting	Present or absent (also multicast)
Packet size	Every network has its own maximum
Ordering	Ordered and unordered delivery
Quality of service	Present or absent; many different kinds
Reliability	Different levels of loss
Security	Privacy rules, encryption, etc.
Parameters	Different timeouts, flow specifications, etc.
Accounting	By connect time, packet, byte, or not at all

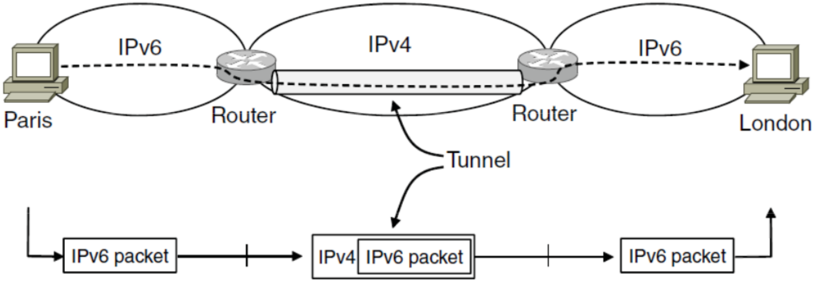
How networks can be connected?



Tunneling

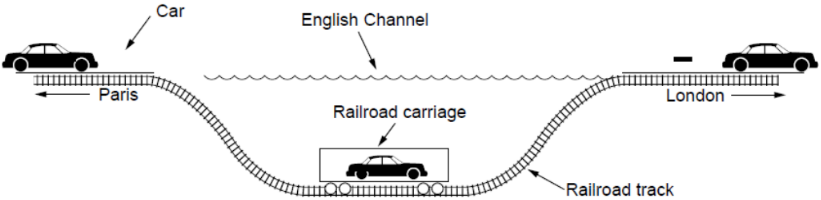
Connects two networks through a middle one

- Packets are encapsulates over the middle



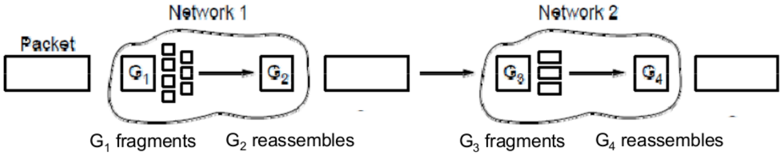
Tunneling analogy:

- tunnel is a link; packet can only enter/exit at ends

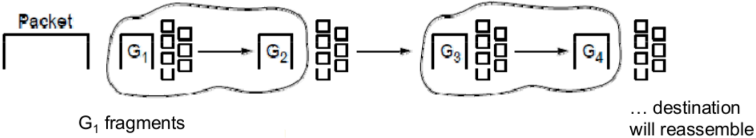


Networks have different packet size limits for many reasons

- Large packets sent with fragmentation & reassembly



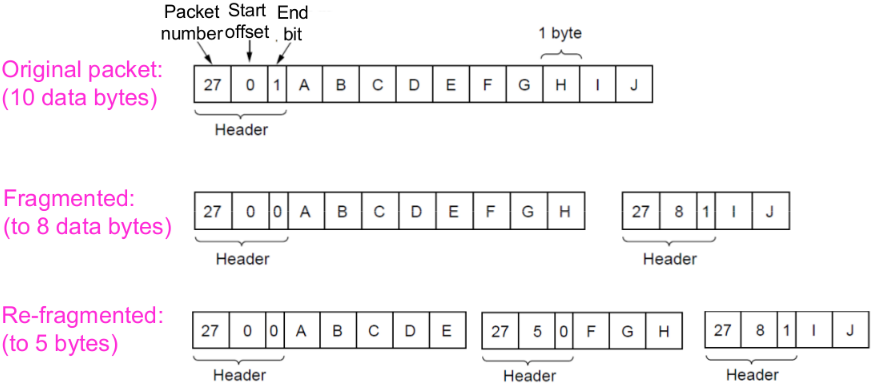
Transparent – packets fragmented / reassembled in each network



Non-transparent – fragments are reassembled at destination

Packet Fragmentation

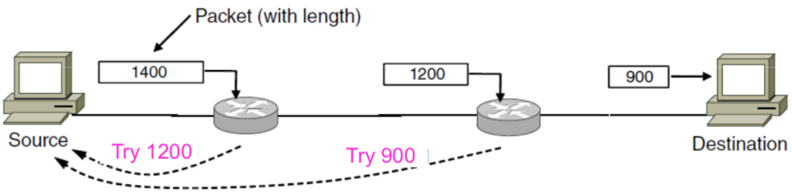
Example of IP-style fragmentation:



Path MTU Discovery

Path MTU Discovery avoids network fragmentation

- Routers return MTU (Max. Transmission Unit) to source and discard large packets



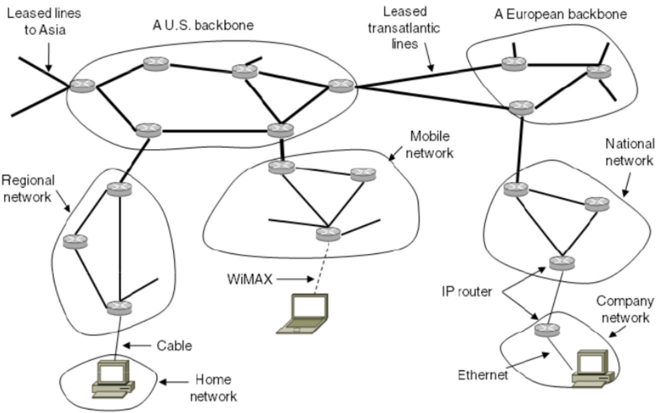
Network Layer in the Internet

IP has been shaped by guiding principles:

- Make sure it works
- Keep it simple
- Make clear choices
- Exploit modularity
- Expect heterogeneity
- Avoid static options and parameters
- Look for good design (not perfect)
- Strict sending, tolerant receiving
- Think about scalability
- Consider performance and cost

Network Layer in the Internet

Internet is an interconnected collection of many networks that is held together by the IP protocol



IPv4 Protocol

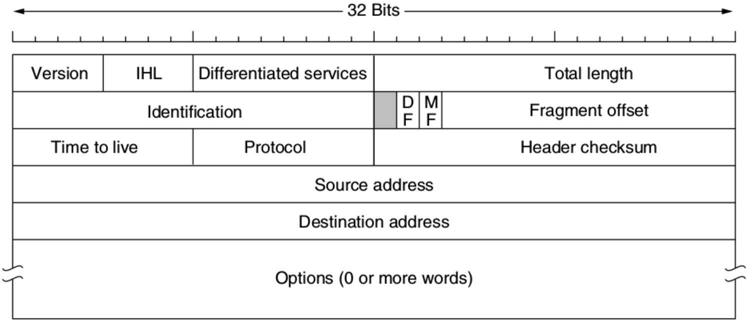


Figure 5-46. The IPv4 (Internet Protocol) header.

IPv4 Header Options

Option	Description
Security	Specifies how secret the datagram is
Strict source routing	Gives the complete path to be followed
Loose source routing	Gives a list of routers not to be missed
Record route	Makes each router append its IP address
Timestamp	Makes each router append its address and timestamp

Figure 5-47. Some of the IP options.

IP Addresses

- **Address** - The unique number ID assigned to one host or interface in a network.
- **Subnet** - A portion of a network that shares a particular subnet address.
- **Subnet mask** - A 32-bit combination used to describe which portion of an address refers to the subnet and which part refers to the host.
- **Interface** - A network connection.

Understanding IP Addresses

- An IP address is an address used in order to uniquely identify a device on an IP network. The address is made up of 32 binary bits, which can be divisible into a network portion and host portion with the help of a **subnet mask**.
- The 32 binary bits are broken into four octets (1 octet = 8 bits). Each octet is converted to decimal and separated by a period (dot). (for example, 172.16.81.100).
- The value in each octet ranges from 0 to 255 decimal, or 00000000 - 11111111 binary.

Understanding IP Addresses

Example IP address: 10.1.23.19
The octets are broken down to provide an addressing scheme that can accommodate large and small networks. There are five different classes of networks, A to E.

Classful IP Addresses

Given an IP address, its class can be determined from the three high-order bits (the three left-most bits in the first octet).

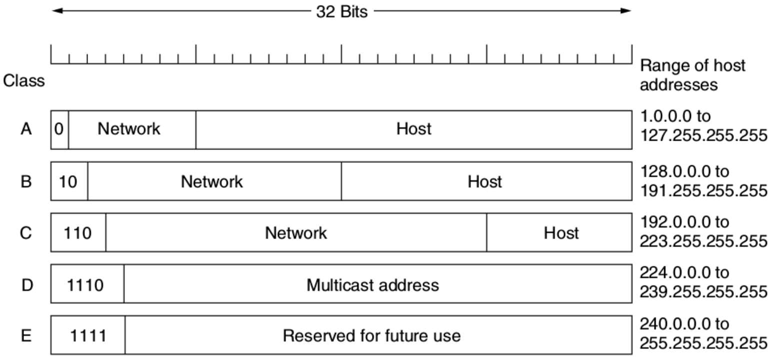


Figure 5-53. IP address formats.

Classful IP Addresses

- In a Class A address, the first octet is the network portion, so the Class A has a major network address of 1.0.0.0 - 127.255.255.255. Class A addresses are used for networks that have more than 65,536 hosts (actually, up to 16777214 hosts!).
- In a Class B address, the first two octets are the network portion, so the Class B has a major network address of 128.0.0.0 - 191.255.255.255. Class B addresses are used for networks that have between 256 and 65534 hosts.
- The Class C address has a major network address of 192.0.0.0 - 223.255.255.255. Octet 4 (8 bits) is for local subnets and hosts - perfect for networks with less than 254 hosts.

Network Masks

A network mask helps you know which portion of the address identifies the network and which portion of the address identifies the node. Class A, B, and C networks have default masks, also known as natural masks, as shown here:

Class A: 255.0.0.0 Class B: 255.255.0.0 Class C: 255.255.255.0

An IP address on a Class A network that has not been subnetted would have an address/mask pair similar to: 8.20.15.1 255.0.0.0. In order to see how the mask helps you identify the network and node parts of the address, convert the address and mask to binary numbers.

8.20.15.1 = 00001000.00010100.00001111.00000001
255.0.0.0 = 11111111.00000000.00000000.00000000

Once you have the address and the mask represented in binary, then identification of the network and host ID is easier. Any address bits which have corresponding mask bits set to 1 represent the network ID. Any address bits that have corresponding mask bits set to 0 represent the node ID.

```
8.20.15.1 = 00001000.00010100.00001111.00000001
255.0.0.0 = 11111111.00000000.00000000.00000000
          -----------------------------------
            net id |      host id
```

netid = 00001000 = 8 **hostid = 00010100.00001111.00000001 = 20.15.1**

Understanding Subnetting

- Subnetting allows you to create multiple logical networks that exist within a single Class A, B, or C network. If you do not subnet, you are only able to use one network from your Class A, B, or C network, which is unrealistic.
- Each data link on a network must have a unique network ID, with every node on that link being a member of the same network. If you break a major network (Class A, B, or C) into smaller subnetworks, it allows you to create a network of interconnecting subnetworks.
- Each data link on this network would then have a unique network/subnetwork ID. Any device, or gateway, that connects 'n' networks/subnetworks has 'n' distinct IP addresses, one for each network / subnetwork that it interconnects.

Understanding Subnetting

- **In order to subnet a network, extend the natural mask with some of the bits from the host ID portion of the address in order to create a subnetwork ID. For example, given a Class C network of 204.17.5.0 which has a natural mask of 255.255.255.0, you can create subnets in this manner:**

204.17.5.0 - 11001100.00010001.00000101.00000000

255.255.255.224 - 11111111.11111111.11111111.11100000

--------------------------------|sub|----

- **By extending the mask to be 255.255.255.224, you have taken three bits (indicated by "sub") from the original host portion of the address and used them to make subnets.**
- **With these three bits, it is possible to create eight subnets. With the remaining five host ID bits, each subnet can have up to 32 host addresses, 30 of which can actually be assigned to a device since host ids of all zeros or all ones are not allowed(it is very important to remember this).**

Understanding Subnetting

255.255.255.224	204.17.5.000 00000 - 204.17.5.000 11111
255.255.255.224	204.17.5.001 00000 - 204.17.5.001 11111
255.255.255.224	204.17.5.010 00000 - 204.17.5.010 11111
255.255.255.224	204.17.5.011 00000 - 204.17.5.011 11111
255.255.255.224	204.17.5.100 00000 - 204.17.5.100 11111
255.255.255.224	204.17.5.101 00000 - 204.17.5.101 11111
255.255.255.224	204.17.5.110 00000 - 204.17.5.110 11111
255.255.255.224	204.17.5.111 00000 - 204.17.5.111 11111

The mask of 255.255.255.224 can also be denoted as /27 as there are 27 bits that are set in the mask.

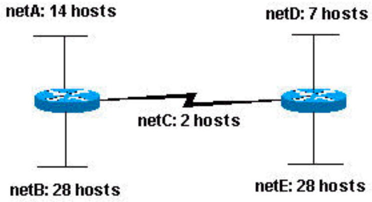
Understanding Subnetting

If you have network 172.16.0.0, then you know that its natural mask is 255.255.0.0 or 172.16.0.0/16. Extending the mask to anything beyond 255.255.0.0 means you are subnetting.

If you use a mask of 255.255.248.0 (/21), how many subnets and hosts per subnet does this allow for?

Understanding Subnetting

Given the Class C network of 204.15.5.0/24, subnet the network in order to create the network below with the host requirements shown.



Understanding Subnetting

You can start by looking at the subnet requirement. In order to create the five needed subnets you would need to use three bits from the Class C host bits. Two bits would only allow you four subnets (2^2).

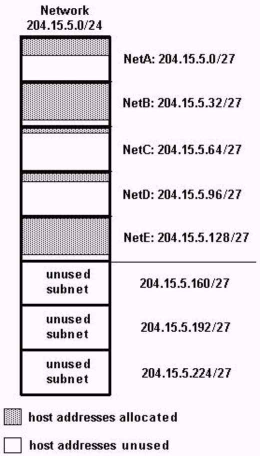
Since you need three subnet bits, that leaves you with five bits for the host portion of the address. How many hosts does this support? 2^5 = 32 (30 usable). This meets the requirement.

Therefore you have determined that it is possible to create this network with a Class C network. An example of how you might assign the subnetworks is:

```
netA: 204.15.5.0/27      host address range 1 to 30
netB: 204.15.5.32/27     host address range 33 to 62
netC: 204.15.5.64/27     host address range 65 to 94
netD: 204.15.5.96/27     host address range 97 to 126
netE: 204.15.5.128/27    host address range 129 to 158
```

Variable Length Subnet Mask

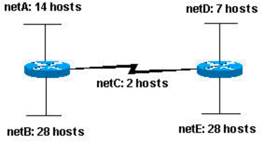
In the previous example,



VLSM allows you to use different masks for each subnet, thereby using address space efficiently.

Variable Length Subnet Mask

Given the same network and requirements develop a subnetting scheme with the use of VLSM, given:



Variable Length Subnet Mask

Determine what mask allows the required number of hosts.

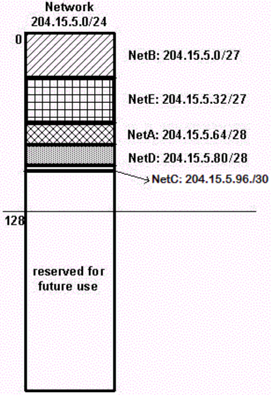
```
netA: requires a /28 (255.255.255.240) mask to support 14 hosts
netB: requires a /27 (255.255.255.224) mask to support 28 hosts
netC: requires a /30 (255.255.255.252) mask to support 2 hosts
netD*: requires a /28 (255.255.255.240) mask to support 7 hosts
netE: requires a /27 (255.255.255.224) mask to support 28 hosts

* a /29 (255.255.255.248) would only allow 6 usable host addresses
  therefore netD requires a /28 mask.
```

The easiest way to assign the subnets is to assign the largest first. For example, you can assign in this manner:

```
netB: 204.15.5.0/27  host address range 1 to 30
netE: 204.15.5.32/27 host address range 33 to 62
netA: 204.15.5.64/28 host address range 65 to 78
netD: 204.15.5.80/28 host address range 81 to 94
netC: 204.15.5.96/30 host address range 97 to 98
```

Variable Length Subnet Mask



CIDR

- Classless Inter domain Routing (CIDR) was introduced in order to improve both address space utilization and routing scalability in the Internet.
- CIDR moves way from the traditional IP classes (Class A, Class B, Class C, and so on). In CIDR , an IP network is represented by a prefix, which is an IP address and some indication of the length of the mask.
- CIDR also depicts a more hierarchical Internet architecture, where each domain takes its IP addresses from a higher level. This allows for the summarization of the domains to be done at the higher level.
- Ex: if an ISP owns network 172.16.0.0/16, then the ISP can offer 172.16.1.0/24, 172.16.2.0/24, and so on to customers. Yet, when advertising to other providers, the ISP only needs to advertise 172.16.0.0/16.