RW354
Principles of Computer Networking

A.E. Krzesinski and B.A. Bagula

Department of Computer Science

University of Stellenbosch

Last updated: 19 August 2004

The material presented in these slides is used with permission from

- Larry L. Peterson and Bruce S. Davie. Computer Networks: A Systems Approach (Second Edition). Morgan Kaufmann Publishers. ISBN 1-55860-577-0.
- William Stallings. Data and Computer Communications (Sixth Edition). Prentice-Hall Inc. ISBN 0-13-571274-2.
- Andrew S. Tannenbaum. Computer Networks (Fourth Edition). Prentice Hall Inc. ISBN 0-13-349945-6.

The Internet Protocol

Summary	*IP provides a basic delivery service for transport protocols such as TCP and UDP. IP is responsible for getting data to its destination host & network. IP is not reliable, so the effort may fail.*	
Relevant STD's	*2*	
	3	*includes RFCs 1122 & 1123*
	4	*RFC 1812, re-published*
	5	*includes RFCs 791, 792, 919, 922, 950 & 1112*
Relevant RFCs	*781*	*Timestamp Option*
	791	*Internet Protocol*
	815	*Fragmentation Re-assembly*
	919	*IP Broadcasts*
	922	*Broadcasting on Sub-Nets*
	950	*Sub-Net Recommendations*
	1108	*Security Option*
	1112	*IP Multicasting & IGMP V1*
	1122	*Host Network Requirements*
	1349	*Type-of-Service Flags*
	1455	*Data-Link Security TOS Flags*
	1812	*Router Requirements*
	2113	*Router Alert Option*

The Internet Standardization Process

Internet protocols & services are developed by volunteers who work within a collaborative environment sponsored by the Internet Engineering Task Force. The IETF investigates areas of interest. Each area consists of several Working Groups.

The Internet Engineering Steering Group ratifies specifications as standards.

The Internet Architecture Board & in particular the Internet Research Task Force does research for the IETF. The IAB consists of 13 elected members.

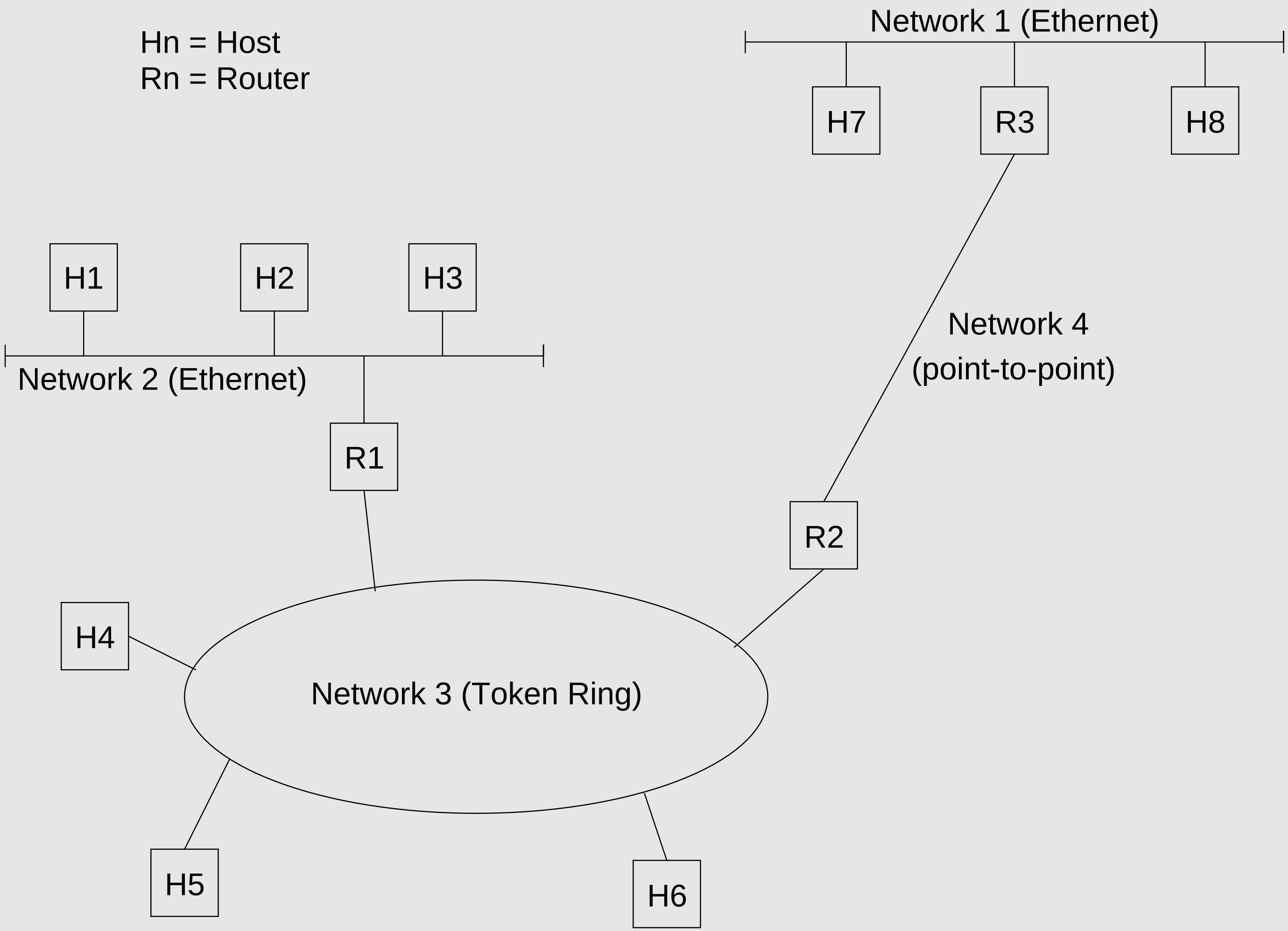
The Internet Assigned Numbers Authority & the RFC Editor support the IAB.

Drafts, RFCs & Standards

RFC 2026 gives an overview of the Internet standardization process: participation is robust

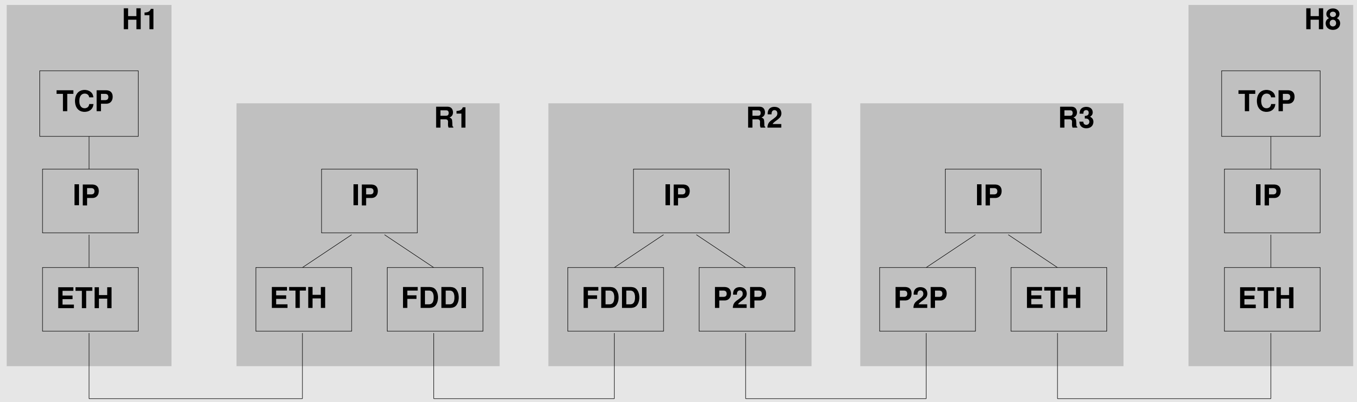
- *the IAB asks the IESG to create an IETF WG to R&D a protocol: a draft proposal is published*
- *the draft proposal is peer-reviewed*
- *the IESG accepts the draft which becomes an RFC*
- *if the RFC is for an IESG standard, the RFC becomes a proposed standard*
- *many RFCs are Informational, Historical, Experimental, Best Common practice, For Your Information*
- *the RFC remains a proposed standard for 6 months until 2 implementations have been demonstrated*
- *the proposed standard becomes a draft standard*
- *the draft standard becomes an Internet standard.*

IP: Overview



Networks are interconnected by routers which operate at layer 3 and route packets between heterogeneous networks.

IP: Overview



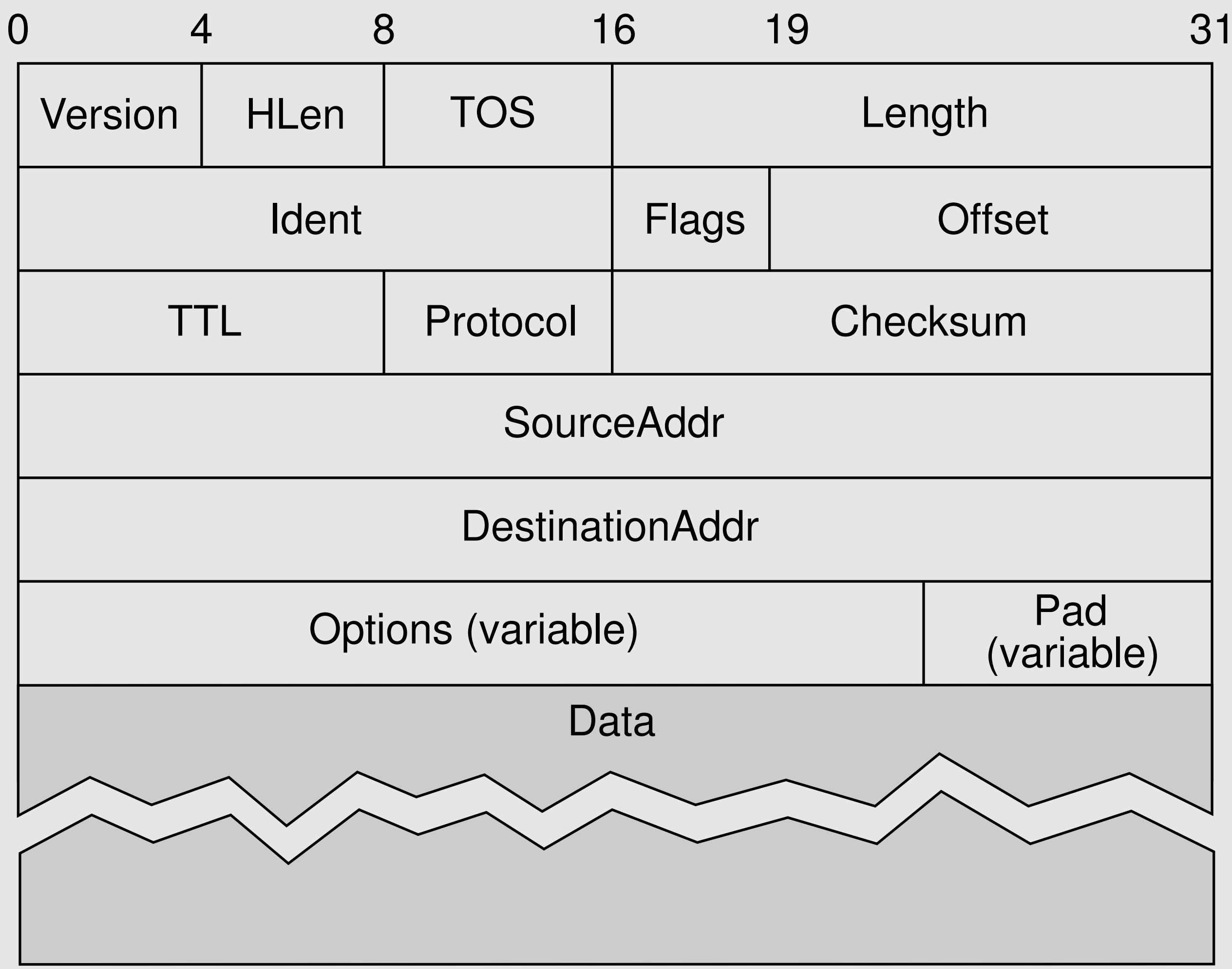
The service model

- *packet delivery model*
- *global addressing scheme.*

IP: Packet Delivery Model

- *Connectionless: datagram-based*
- *Best-effort delivery: unreliable service*
  - *packets are lost*
  - *packets are delivered out of order*
  - *duplicate copies of a packet are delivered*
  - *packets can be delayed for a long time.*
- *Best effort implies*
  - *the routers are simple*
  - *the protocols & applications that run above IP need to be aware of failed packets.*

IP: Datagram Format



IP: Datagram Format

- `Version` *(4): currently IPv4*
- `HLen` *(4): no. of 32-bit words in header (min 20)*
- `TOS` *(8): type of service: not widely used*
- `Length` *(16): no. of bytes in this datagram (max 65,535)*
- `Ident` *(16): used by fragmentation*
- `Flags/Offset` *(16): used by fragmentation*
- `TTL` *(8): no. of hops this datagram has travelled (64)*
- `Protocol` *(8): demux key: TCP=6, UDP=17, . . .*
- `Checksum` *(16): of the header only*
- `DestinationAddr` *&* `SourceAddr` *(32)*
- `Options`*: not widely used . . .*

IP: Datagram Format

The IP options are

- *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 into the packet payload*
- *Timestamp: makes each router append its IP address & timestamp into the packet payload.*

IP: Fragmentation & Re-assembly

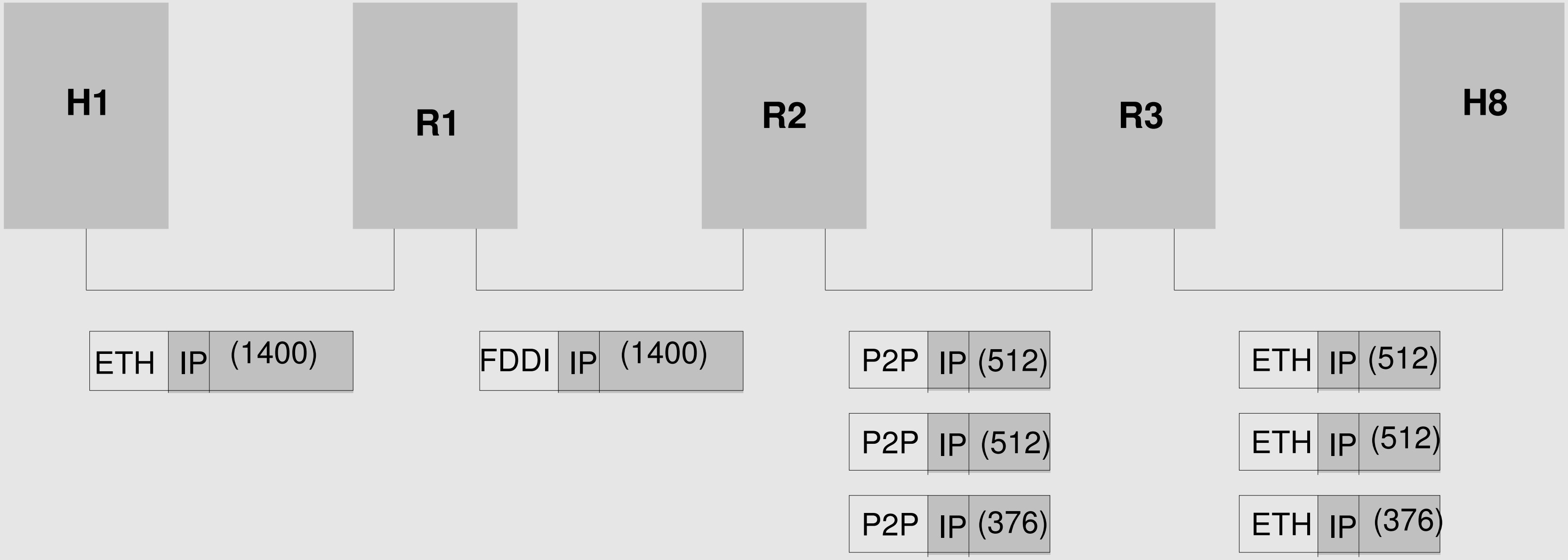
Each network has a Maximum Transmission Unit MTU.

Strategy

- *fragment when necessary (datagram $>$ MTU)*
- *try to avoid fragmentation at the source host*
- *re-fragmentation is possible*
- *fragments are self-contained datagrams*
- *delay re-assembly until the fragments arrive at the destination host*
- *do not recover from lost fragments.*

IP: Fragmentation & Re-assembly

Example



IP: Fragmentation & Re-assembly

Start of header				
ident = x			0	offset = 0
Rest of header				
1400 data bytes				

Unfragmented Packet

Start of header				
ident = x			1	offset = 0
Rest of header				
512 data bytes				

First Fragment

Start of header				
ident = x			1	offset = 512
Rest of header				
512 data bytes				

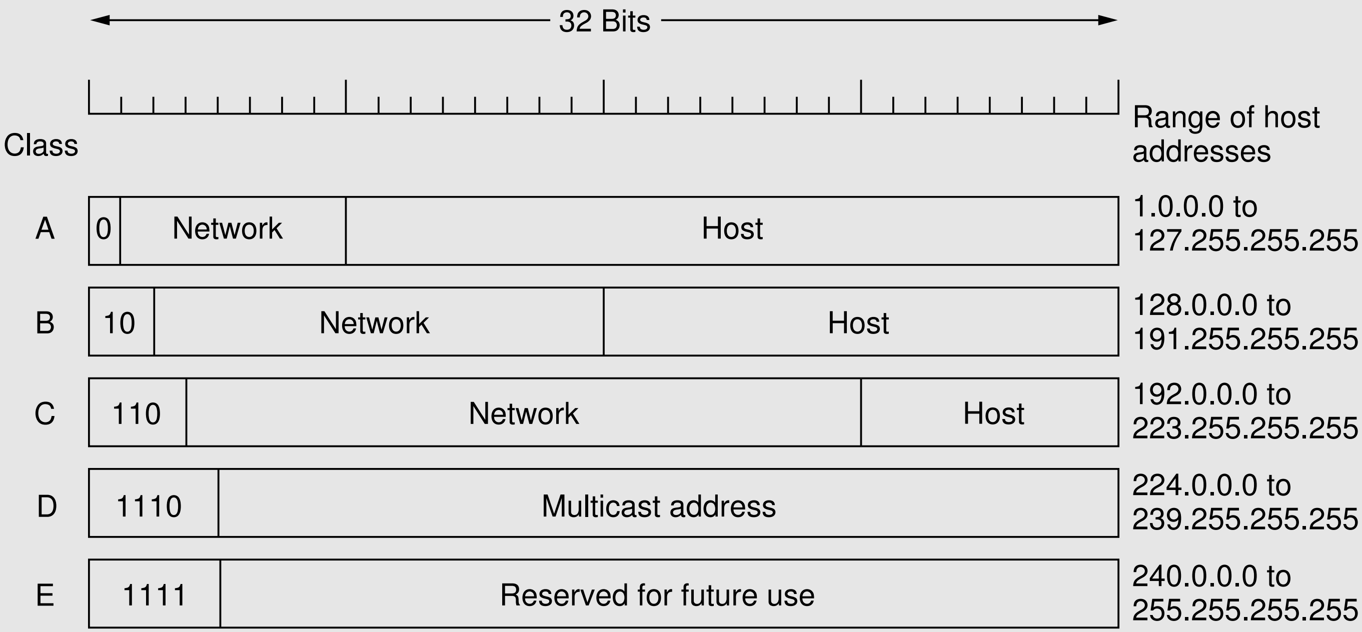
Second Fragment

Start of header				
ident = x			0	offset = 1024
Rest of header				
376 data bytes				

Last Fragment

IP: Global Addresses

- *Properties*
  - *globally unique*
  - *hierarchical: network + host*
- *Format*



IP: Special Addresses

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		This host
0 0 . . . 0 0	Host	A host on this network
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		Broadcast on the local network
Network	1 1 1 1 . . . 1 1 1 1	Broadcast on a distant network
127	(Anything)	Loopback

IP: Datagram Forwarding

Strategy

- *every datagram contains the IP address of the destination host*
- *compare the network parts of the source & destination addresses*
  - *if the source & the destination are on the same network, then forward the datagram to the destination host*
  - *else forward the datagram to a router*
    - *each router maintains a forwarding table*
    - *the forwarding table maps a network number into the next hop on the shortest path to the destination network*
  - *each host has a default router.*

IP: Datagram Forwarding

An example of a routing table

Network Number	*Next Hop*
1	*R3*
2	*R1*
3	*interface 1*
4	*interface 0*

IP: Address Translation

- Map IP addresses into physical addresses
  - encode physical address in host part of IP address
    - cannot encode a 48-bit ethernet address in a 32-bit IP address
  - each host maintains a table of address pairs: IP to physical address bindings.
- Address Resolution Protocol ARP (RFC 826)
  - broadcast an ARP query if the IP/physical binding is not in the table
  - the target machine responds with its physical address
  - table entries are discarded if not refreshed $\sim$10 minutes.

IP: Address Resolution Protocol

Host A broadcasts an ARP request to find the physical address of host B. All hosts attached to A's network receive the ARP request

- *all receivers update their ARP caches with host A's IP/physical pair*
- *host B returns an ARP response to host A containing host B's IP/physical pair.*

Whenever a host receives an ARP request it caches the IP/physical pair.

New machines appearing on net will broadcast their IP/physical addresses.

IP: Address Translation

ARP packet format for mapping IP addresses into ethernet addresses

HardwareType=1		ProtocolType=0x0800
HLEN=48	PLEN=32	Operation
SourceHardwareAddr		
SourceHardwareAddr		SourceProtocolAddr
SourceProtocolAddr		TargetHardwareAddr
TargetHardwareAddr		
TargetProtocolAddr		

- `HardwareType`: *type of physical network (e.g. Ethernet)*
- `ProtocolType`: *type of higher layer protocol (e.g. IP)*
- `HLEN`/`PLEN`: *length of physical/protocol addresses*
- `Operation`: *request or response*
- `Source`/`Target`: *physical/protocol addresses.*

IP: Address Translation

Notes

- *table entries timeout in about 10 minutes*
- *update table with source when you are the target*
- *update table if already have an entry*
- *do not refresh table entries upon reference.*

IP: Dynamic Host Configuration Protocol

Each host must know the IP addresses of itself & its default router.

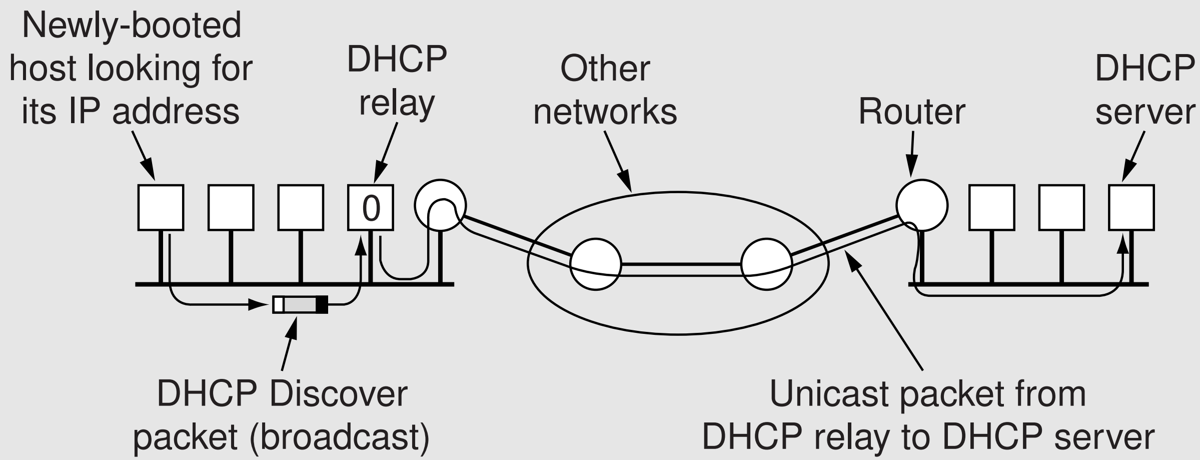
It is not possible to manually configure the IP information needed by the hosts in a network.

The DHCP server – one for each administrative domain – provides the IP configuration information for the hosts.

IP: Dynamic Host Configuration Protocol

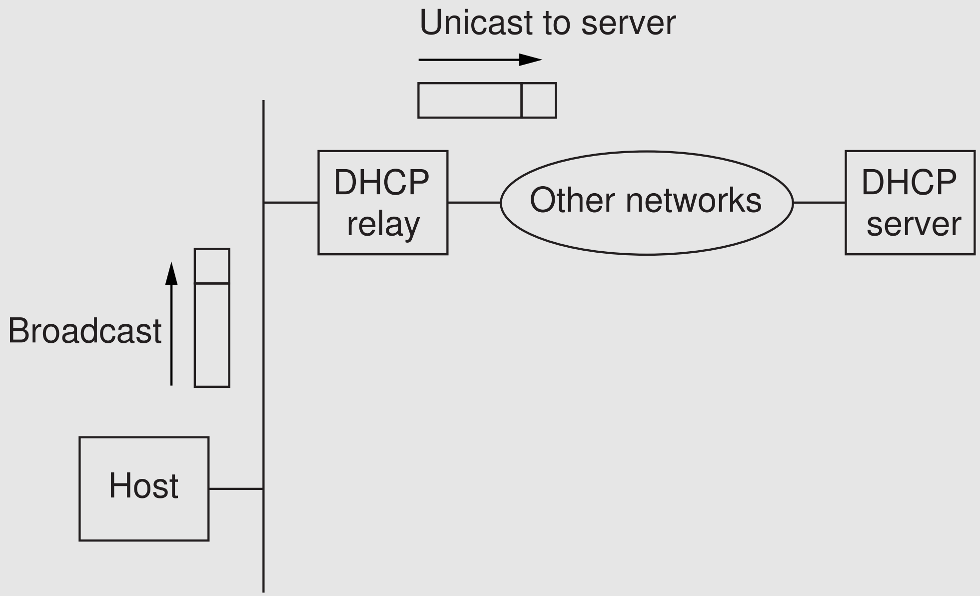
DHCP is defined in RFCs 2131, 2132.

Server discovery



- *the host sends a* `DHCPDISCOVER` *message to broadcast address 255.255.255.255*
- *the DHCP server returns the IP address to the host*
- *the broadcast* `DHCPDISCOVER` *message is not forwarded by the routers . . .*

IP: Dynamic Host Configuration Protocol



A DHCP relay agent receives a broadcast `DHCPDISCOVER` *message from a host & sends a unicast* `DHCPDISCOVER` *message to the DHCP server.*

The DHCP server responds with a `DHCPOFFER` *message containing the requested IP address. The host must respond within 120 seconds with a* `DHCPREQUEST` *message. The server responds with a* `DHCPACK` *message confirming that the host may use the IP address.*

IP: Dynamic Host Configuration Protocol

Operation	HType	HLen	Hops
Xid			
Secs		Flags	
ciaddr			
yiaddr			
siaddr			
giaddr			
chaddr (16 bytes)			
sname (64 bytes)			
file (128 bytes)			
options			

DHCP packet format: encapsulated in UDP

- `chaddr`: *client hardware (ethernet) address*
- `yiaddr`: *your IP address.*

Dynamic IP addresses: the DHCP server leases the IP address for some period of time.

IP: Dynamic Host Configuration Protocol

DHCP allocates IP addresses in 4 ways

- *Permanent fixed addresses are allocated by the network administrator. These addresses are not controlled by DHCP.*
- *Manual allocation. The network administrator sets up these addresses in the DHCP configuration.*
- *Automatic allocation. DHCP assigns permanent addresses from a pool of addresses.*
- *Dynamic allocation. DHCP assigns addresses for a limited time (lease). A user can return the address to the pool at any time. A user must request an extension to lease the address beyond the address' lifetime. Expired addresses are returned to the pool.*

IP: Internet Control Message Protocol RFC 792

ICMP defines a set of error messages that are returned to the source when a router or a host cannot process an IP datagram successfully

- *echo (ping)*
- *redirect (there is a better route to the destination)*
- *destination unknown (network, host)*
- *destination unreachable (network, host, protocol, port)*
- *TTL exceeded (so datagrams don't cycle forever)*
- *checksum failed*
- *re-assembly failed*
- *source quench (seldom used)*
- *cannot fragment.*

ICMP: TraceRoute

TraceRoute determines the names & addresses of routers from a source to a destnation

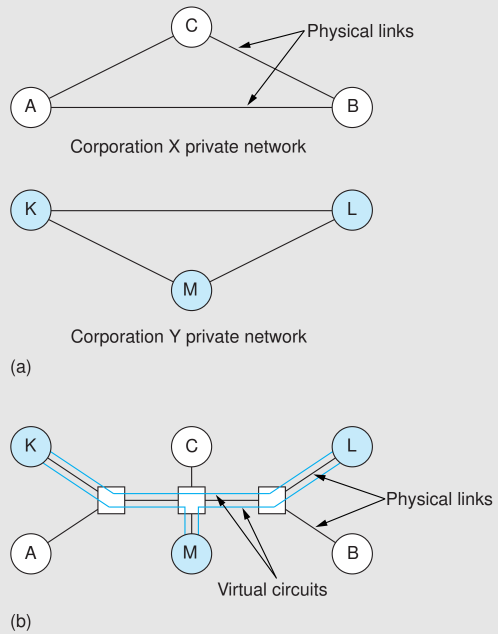
- *TraceRoute in the source sends a series of IP packets to the destination*

  IP$_1$ *has TTL=1, IP*$_2$ *has TTL=2, . . .*
- *when the* n*th IP packet arrives at the* n*th router, its TTL expires*
- *the router sends a timestamped ICMP packet to the source*
- *the source displays the RTT to router* n *and the IP address of the router.*

IP: Virtual Networks & Tunnels

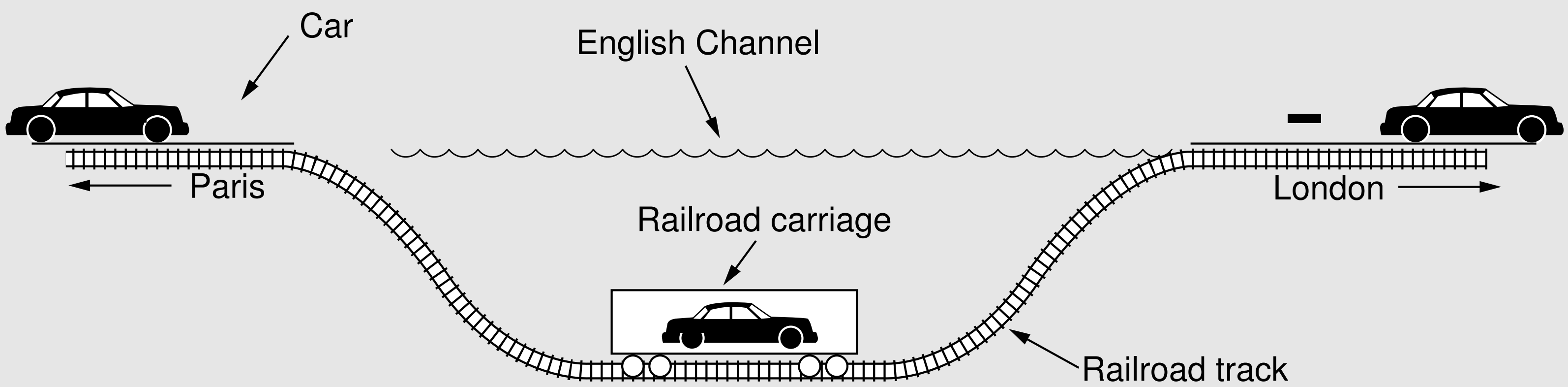
A private network is built by leasing transmission lines & using them exclusively to interconnect certain sites.

A Virtual Private Network (VPN) is built from virtual circuits.

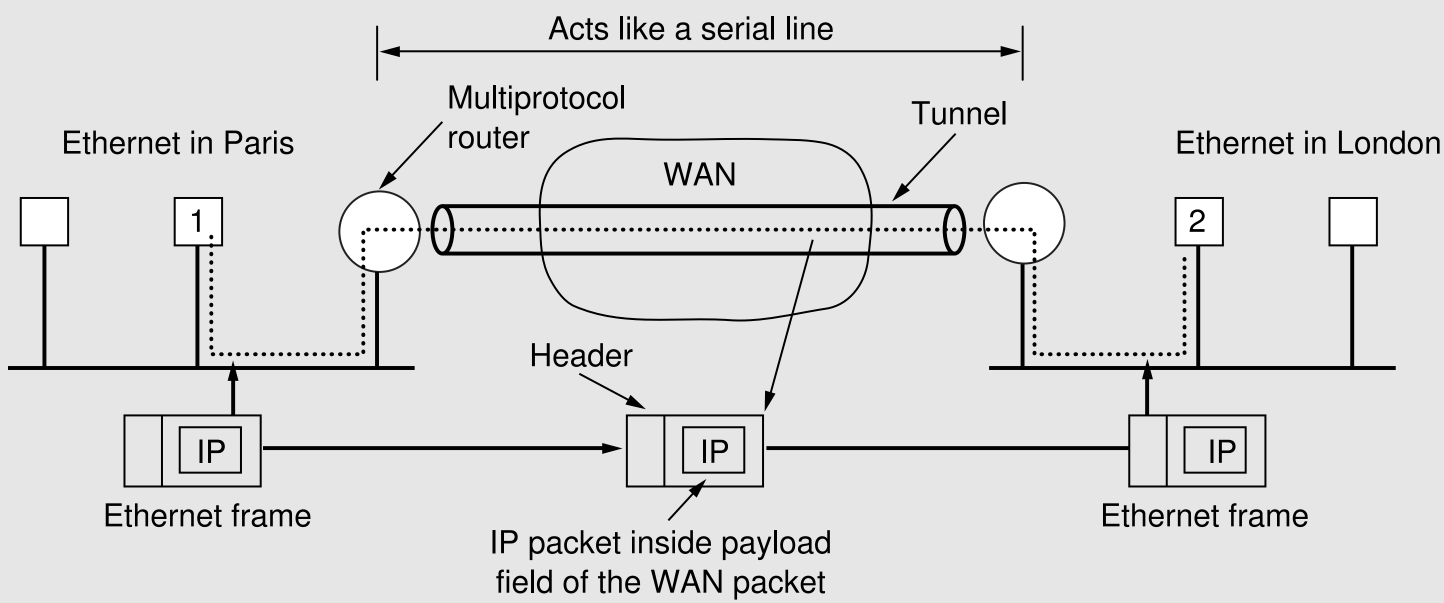


IP VPN's are constructed using IP tunnels: a virtual point-to-point link between an O-D pair that may be separated by an arbitrary number of networks.

IP: Tunnels



Car
English Channel
Paris
London
Railroad carriage
Railroad track

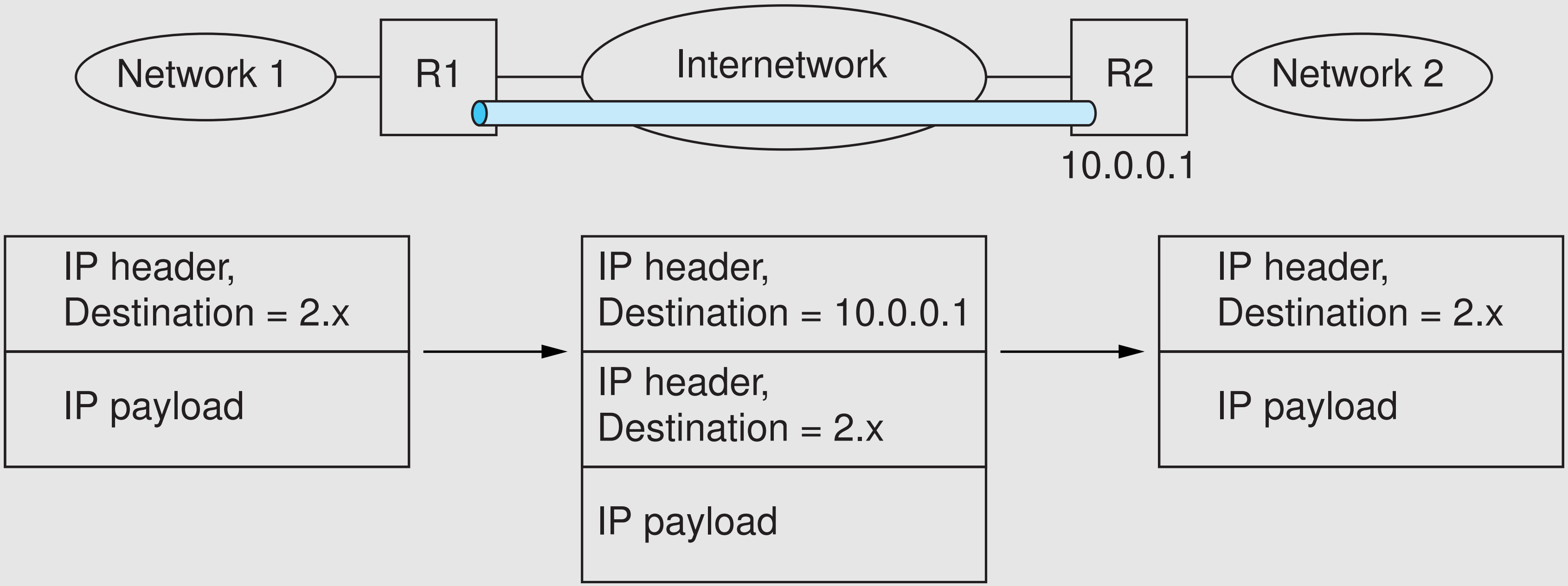


Acts like a serial line
Multiprotocol
router
Tunnel
Ethernet in Paris
WAN
Ethernet in London
1
2
Header
IP
IP
IP
Ethernet frame
IP packet inside payload
field of the WAN packet
Ethernet frame

IP: Tunnels

The router at the entrance to the IP tunnel is provided with the IP address of the router at the end of the tunnel.

A packet that traverses the tunnel: the entrance router encapsulates the packet in an IP datagram with src/dest addresses set to the IP addresses of the start/end of the tunnel.



IP: Tunnels

The tunnel (virtual link) is configured into the forwarding table of the router at the entrance to the tunnel.

A packet sent through the tunnel becomes a datagram destined for the router at the end of the tunnel.

The router at the end of the tunnel notes that the datagram is destined for itself: it removes the IP header & finds an IP packet in the payload which it forwards.

Tunnels are used to provide

- *security*
- *routers with special capabilities can be connected by tunnels to create a virtual network: two IPv6 nodes connected by two IPv4 routers*
- *carry non-IP packets across an IP network.*

Routing: scalability

An Autonomous System is a collection of one or more networks under a single technical administration

- *an AS corresponds to an administrative domain*
- *examples: University, company, backbone network.*

Routing is divided into two parts to improve scalability: intra-domain routing and inter-domain routing.

The intra-domain routing algorithms & protocols do not scale. They are designed for networks with less than 100 nodes.

The intra-domain domain routing protocols (Interior Gateway Protocol: IGP) form a building block for hierarchical inter-domain routing protocols (Border Gateway Protocol: BGP) which do scale.

Routing

- *Forwarding versus routing*
  - *forwarding: to select an output port based on the destination address and the routing table*
  - *routing: the process by which a routing table is built.*
- *Static versus adaptive routing*
  - *static routing algorithms do not base their routing decisions on the network topology & estimates of the current traffic*
  - *adaptive routing algorithms base their routing decisions on the network topology & usually on estimates of the current traffic.*

Routing

If the network uses

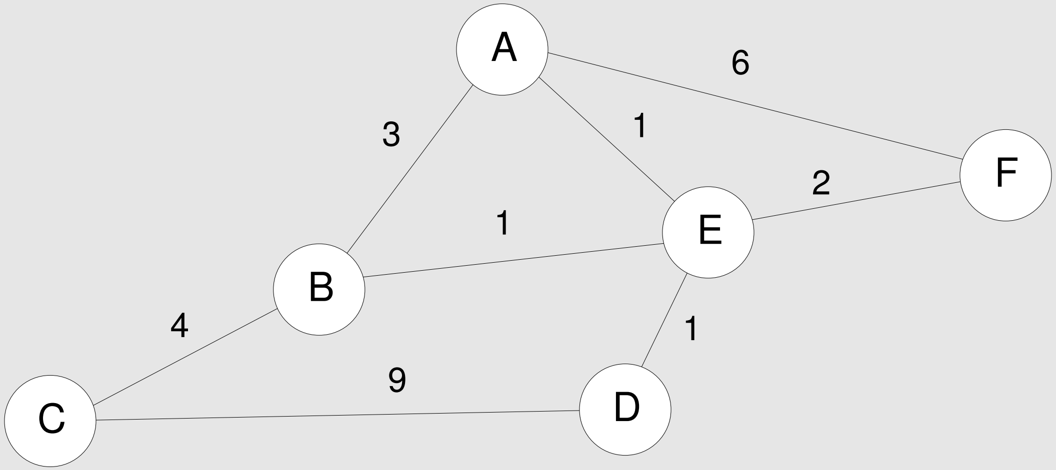
- *datagrams: a routing decision is made for each arriving datagram*
- *virtual circuits: the routing decisions are made when the VCs are set up.*

Certain properties are desirable in a routing algorithm

- *correctness*
- *simplicity*
- *robustness*
- *stability*
- *fairness*
- *optimality.*

Routing

A network can be represented as a graph.



Routing

Find the least cost path between any two nodes. The path cost is the sum of the costs of the links of the path.

The factors involved are

- *The network topology is static except when*
  - *links/nodes fail*
  - *new links/nodes are added.*
- *The traffic carried by the network – and hence the link costs – is dynamic.*

Solution: a distributed, dynamic algorithm to compute shortest paths.

Note that a centralized algorithm would not scale.

Routing

There are two intra-domain routing protocols

- *The distance vector algorithm (Bellman-Ford): Routing Information Protocol (RIP)*
  - *each node communicates only with its directly connected neighbours*
  - *each node communicates all it has learned so far: the distances to all the other nodes (the entire routing table).*
- *The link state algorithm (Dijkstra): Open Shortest Path First (OSPF)*
  - *each node communicates with all other nodes*
  - *each node communicates only what it knows for sure: the state of its directly connected links (a small table).*

Distance Vector (RIPv1 RFC 1058, RIPv2 RFC 2453)

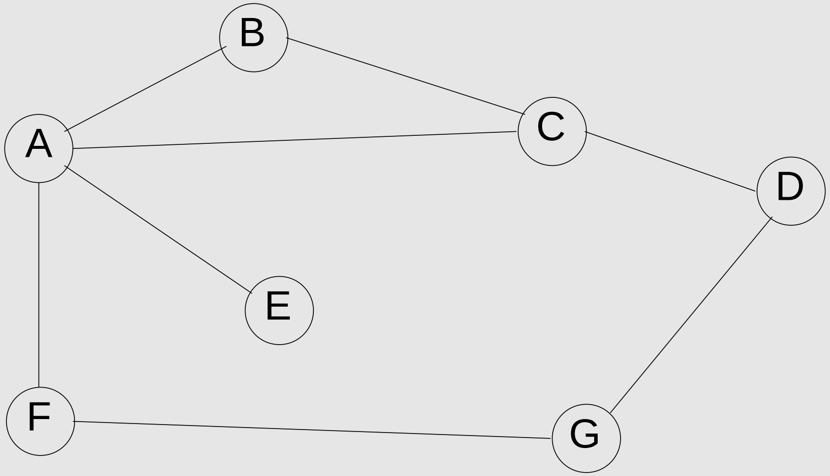
- *Each node maintains a set of triples* $(\mathtt{Destination}, \mathtt{Cost}, \mathtt{NextHop})$
- *Each node sends updates to & receives updates from its directly connected neighbours: the updates are encapsulated in UDP, port 520*
  - *periodic updates every* $\pm$*30 seconds*
  - *triggered updates when its routing table changes.*
- *Each update contains up to 25 pairs* $(\mathtt{Destination}, \mathtt{Cost})$
- *Update a routing table if it receives an update advertising a "better" route with either*
  - *a smaller cost (lower hop count), or*
  - *the update came from the next-hop.*
- *Refresh existing routes, delete them if they time out* $\pm$*180 seconds.*

Distance Vector: Implementation

```
void mergeRoute (Route *new) {
  int i;

  for (i = 0; i < numRoutes; ++i) {
    if (new->Dest == rt[i].Dest) {
      if (new->Cost + 1 < rt[i].Cost)    break;
      if (new->NextHop == rt[i].NextHop) break;
      return;
    }
  }
  rt[i] = *new;
  rt[i].TTL = MAX_TTL;
  ++rt[i].Cost;
  if (i == numRoutes) ++numRoutes;
}
```

Distance Vector: example



The routing table at node B

to	cost	next
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	A

Distance Vector: example

	updates from neighbours			
	from	from	from	from
to	A	I	H	K
A	0	24	20	21
B	12	36	31	28
C	25	18	19	36
D	40	27	8	24
E	14	7	30	22
F	23	20	19	40
G	18	31	6	31
H	17	20	0	19
I	21	0	14	22
J	9	11	7	10
K	24	22	22	0
L	29	33	9	9

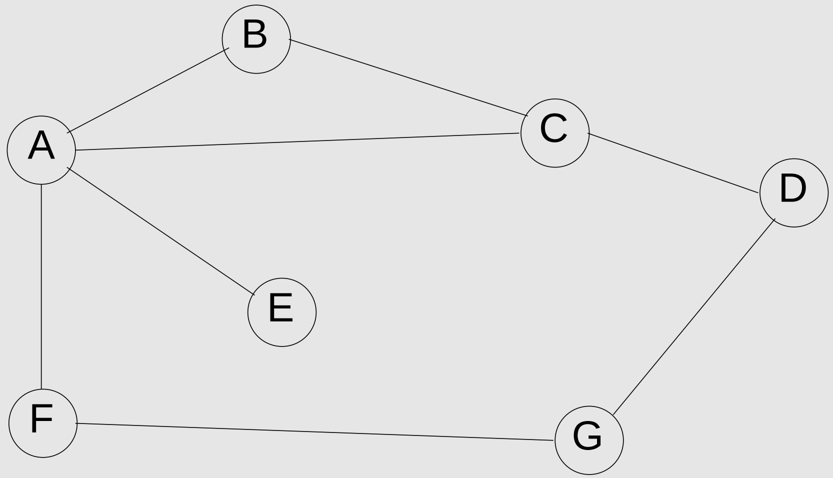
routing table for router J		
to	cost	next
A	8	A
B	20	A
C	28	I
D	20	H
E	17	I
F	30	I
G	18	H
H	12	H
I	10	I
J	0	-
K	6	K
L	15	K

Consider a 12-node network where router J is connected to 4 neighbours A, I, H, K.

Router J computes a new route to G as follows

- *J can get to A in 8 msecs; A can get to G in 18 msecs; J can get to G via A in 26 msecs*
- *likewise J to G via I, H, K in 41, 18, 37 msecs respectively*
- *the best is J to G via H which is entered in the routing table.*

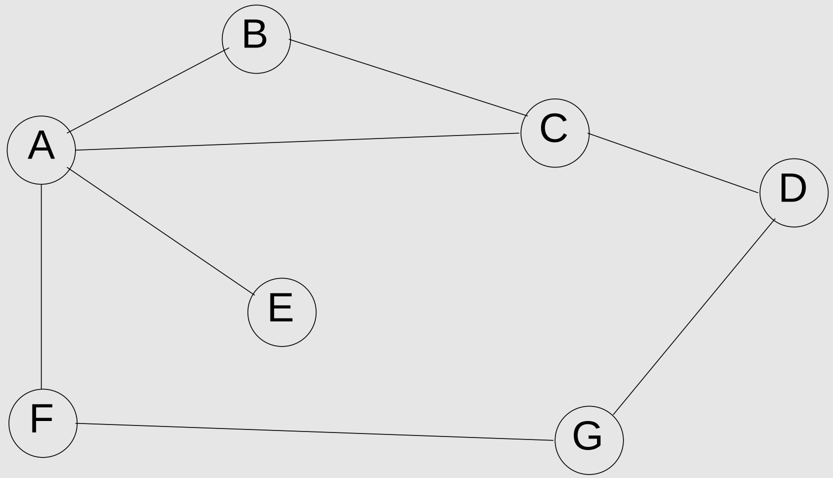
Routing: example



The link from F to G fails

- *F sets the distance to G to ∞, sends an update to A*
  - *A sets the distance to G to ∞ since it uses F to reach G*
- *A receives an update from C with a 2-hop path to G*
  - *A sets the distance to G to 3, sends an update to F*
- *F decides that it can reach G in 4 hops via A.*

Routing loops: count to infinity



The link from A to E fails

- *A sets the distance to E to ∞, sends updates to B, C, F*
- *before C can send an update to B*
  - *B sets the distance to E to 3, sends an update to A*
  - *A sets the distance to E to 4, sends an update to C*
  - *C sets the distance to E to 5, sends an update to A*
  - *A sets the distance to E to 6, sends an update to C*
  - *. . .*

Routing loops: example 2

Heuristics to break routing loops

- *set* ∞ *to 16*
- *split horizon*
- *split horizon with poison reverse*

Link State: OSPF (RFC 2328)

Each node sends information in Link State Advertisement (LSA) packets encapsulated in IP packets, protocol 89

- *each node communicates with all other nodes*
- *each node communicates only what it knows for sure: the state of its directly connected links.*

The LSA contains

- *the id of the node that created the LSA*
- *the cost of the link to each directly connected neighbour (Cisco: cost =* 10^8 */ link bandwidth in bps)*
- *a sequence number* `SEQNO`
- *the time-to-live* `TTL` *for this LSA.*

OSPF scalability

As the size of the network grows, OSPF route computation requires more memory and more CPU resources at each router.

OSPF does not scale.

The solution: the network is partitioned into areas. OSPF computes intra-area routes.

Edge areas are connected to a backbone area via Area Border Routers (ABRs).

Inter-area traffic must pass through the backbone area – the backbone routers know the complete topology of the network.

OSPF: reliable flooding

- *use ARQ to ensure the reliable transmission of LSAs among neighbours*
- *store the most recent LSA (largest* `SEQNO`*) from each node*
- *forward the LSA to all nodes except the one that sent it*
- *generate new LSAs periodically (infrequently) or when the topology changes*
  - *increment the 64-bit* `SEQNO` *(no wrap around)*
- *start the* `SEQNO` *at 0 when rebooting*
- *decrement the* `TTL` *of each stored LSA; discard the LSA when the* $\mathtt{TTL} = \mathtt{0}$.

The most recent copy of each LSA eventually reaches all nodes.

OSPF: Dijkstra's shortest path algorithm

When a node has received an LSA from every other node, it can proceed to find the shortest route from itself to all other nodes. Shortest means least length or least cost.

Notation

- $\mathcal{N}$ = *the set of nodes in the graph*
- $\mathcal{M}$ = *the set of permanently labelled nodes*
- $\ell(i, j)$ = *the non-negative cost of the edge* (i, j)
- $s \in \mathcal{N}$ = *this node*
- $C(n)$ = *the cost of the path from node* s *to node* n.

Shortest routes nest

$$C(n) = \min_{w}(C(n), C(w) + \ell(w, n))$$

OSPF: Dijkstra's shortest path algorithm

/ find the shortest paths from node s to all nodes */*

$\texttt{for}\ (n \in \mathcal{N})\ C(n) := \infty;$

$\mathcal{M} := s; w := s; C(s) := 0;$

$\texttt{while}\ (\mathcal{N} \neq \mathcal{M})\ \{$

/ adjust the cost from n to s via w */*

$\texttt{for}\ (n \in \mathcal{N} - \mathcal{M})$
$C(n) := \min(C(n), C(w) + \ell(w, n));$

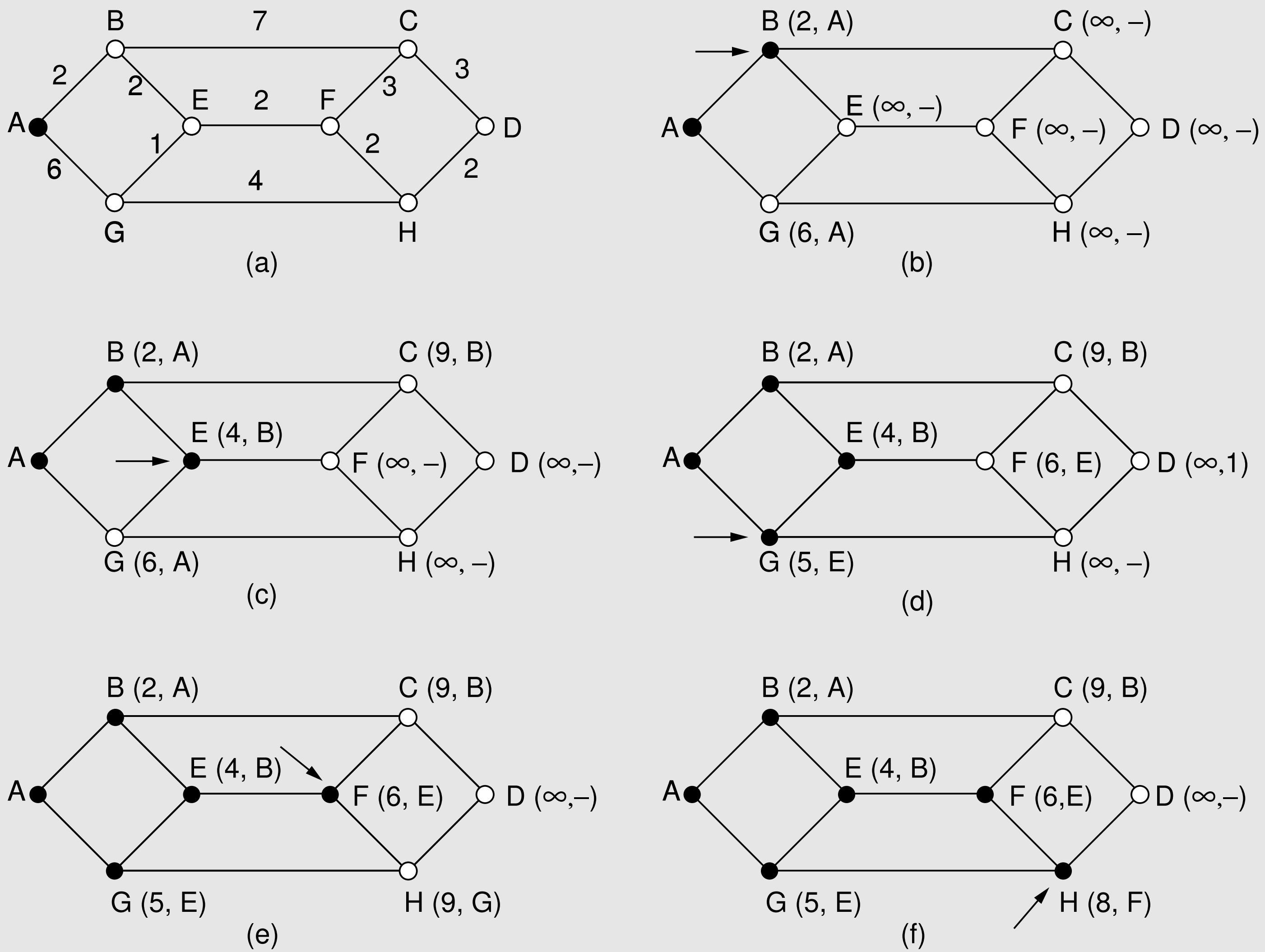
/ find the unlabelled node closest to node s */*

$w := \min_{n \in \mathcal{N} - \mathcal{M}} C(n);$

/ permanently label node w */*

$\mathcal{M} := \mathcal{M} \cup w;$
$\}$

OSPF: Dijkstra’s Algorithm



OSPF: Route Calculation in practice

- *Each router maintains two lists*

  $(\texttt{Confirmed}, \texttt{Tentative})$

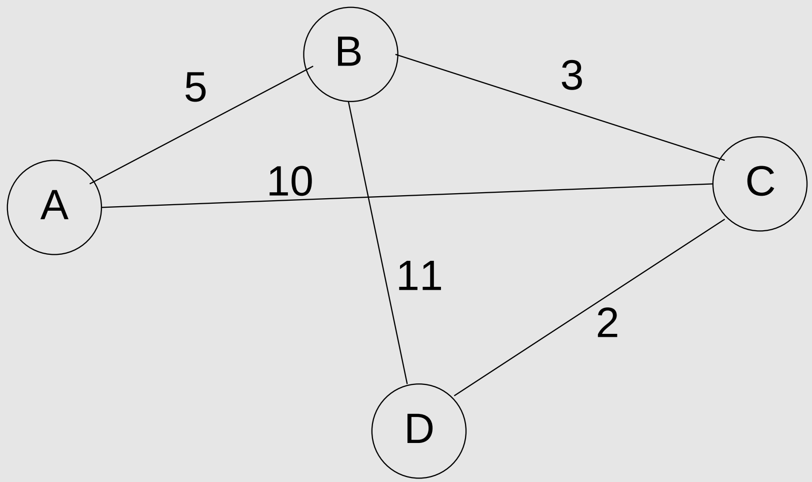
- *Each list consists of a set of triples*

  $(\texttt{Destination}, \texttt{Cost}, \texttt{NextHop})$

OSPF: Route Calculation in practice

1. *Initialize* $\texttt{Confirmed}$ *with the entry* $(\texttt{me}, 0, \texttt{me})$, *set* $\texttt{NextHop} = \texttt{me}$ *and set* $\texttt{Tentative} = \emptyset$.
2. *Select the LSA of the node* $\texttt{NextHop}$ *just added to* $\texttt{Confirmed}$.
3. *Calculate the* $\texttt{Cost}$ *to reach each* $\texttt{Neighbour}$ *of* $\texttt{NextHop}$ *as the sum of the cost from* $\texttt{me}$ *to* $\texttt{NextHop}$ *&* *from* $\texttt{NextHop}$ *to* $\texttt{Neighbour}$
   (a) *If* $\texttt{Neighbour} \notin \texttt{Confirmed} \cup \texttt{Tentative}$ *add* $(\texttt{Neighbour}, \texttt{Cost}, \texttt{NextHop})$ *to* $\texttt{Tentative}$
   (b) *If* $\texttt{Neighbour} \in \texttt{Tentative}$ *&* $\texttt{Cost}$ *is less than the current cost for* $\texttt{Neighbour}$ *then replace the current entry with* $(\texttt{Neighbour}, \texttt{Cost}, \texttt{NextHop})$.
4. *If* $\texttt{Tentative} = \emptyset$ *stop. Else pick the entry in* $\texttt{Tentative}$ *with the lowest cost, move it to* $\texttt{Confirmed}$ *& return to step 2.*

OSPF: Route Calculation in practice



Build the routing table for node D

Step	`Confirmed`	`Tentative`
1	*(D,0,D)*	$\emptyset$
2	*(D,0,D)*	*(B,11,B), (C,2,C)*
3	*(D,0,D), (C,2,C)*	*(B,11,B)*
4	*(D,0,D), (C,2,C)*	*(B,5,C), (A,12,C)*
5	*(D,0,D), (C,2,C), (B,5,C)*	*(A,12,C)*
6	*(D,0,D), (C,2,C), (B,5,C)*	*(A,10,C)*
7	*(D,0,D), (C,2,C), (B,5,C), (A,10,C)*	$\emptyset$

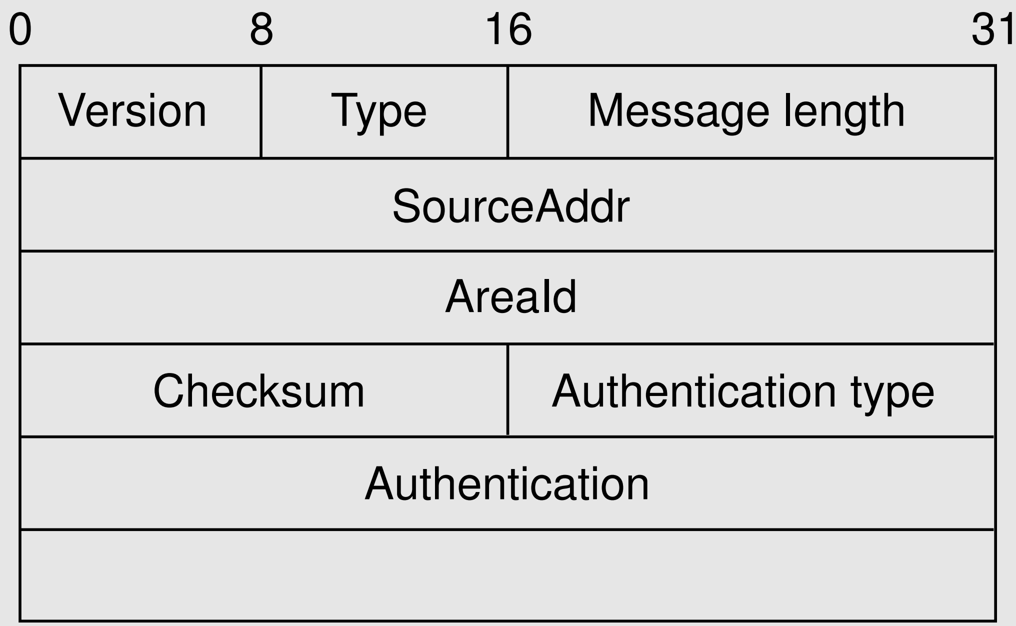
OSPF: details

OSPF provides

- *authentication of routing messages*
- *hierarchy: domains are partitioned into areas*
- *load balancing: several equal cost routes may connect each O-D pair and the traffic will be distributed evenly over these routes.*

OSPF: header

The five OSPF message types all have the same header



LSA packets are encapsulated in IP packets, protocol 89

- *Version (2)*
- *Type (1-5)*

OSPF: Link State Advertisement LSA

LS Age			Options	Type=1
Link state ID				
Advertising router				
LS sequence number				
LS checksum			Length	
0	Flags	0	Number of links	
Link ID				
Link data				
Link type		Num_TOS	Metric	
Optional TOS information				
More links				

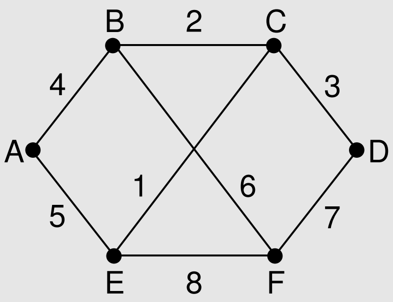
The LSA is the basic building block of OSPF link state messages. One message may contain many LSAs.

OSPF: LSA

- `LSAge`: *TTL*
- `LinkstateID`: *the id of the router that created this message = lowest IP address assigned to router*
- `LSsequencenumber`: *SEQNO*
- `LSchecksum` *covers all the fields except LS Age*
- `Length` *of the message*
- `LinkID`: *id of the router at the far end of the link*
- `Linkdata`: *identify among several parallel links*
- `Metric`: *the cost of the link*
- `LinkType`: *the type of the link (point-to-point, . . .)*
- `Num_TOS`: *multiple metrics, one for each type of service. OSPF can chose different routes depending on the TOS field in the IP header (not widely used).*

OSPF: LSAs

The LSAs for six routers



(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)

ARPANET Metrics

The original ARPANET metric

- *measured the number of packets on each link queue*
- *took neither the latency or the bandwidth of the link into consideration.*

The new ARPANET metric

- *stamp each incoming packet with its arrival time :* $\mathtt{AT}$
- *record the departure time:* $\mathtt{DT}$
- *when the link-level ACK arrives, compute*

$$\mathtt{Delay} = (\mathtt{DT} - \mathtt{AT}) + \mathtt{Transmit} + \mathtt{Latency}$$

- *if timeout, reset* $\mathtt{DT}$ *to departure time for retransmission*
- *link cost = average delay over some time period*

ARPANET Metrics

The problems with the new metric

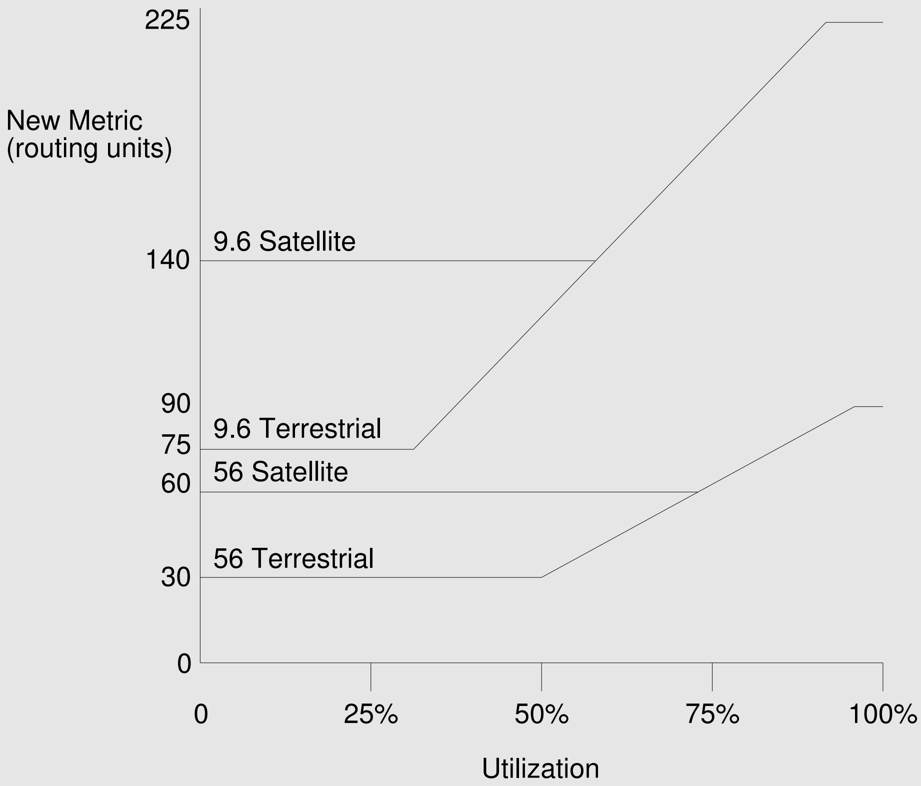
- *Under low load, the static factors dominate the cost & the metric works OK.*
- *Under high load, the congested links had very high costs; packets oscillated between the congested & the idle links.*
- *The range of link costs is too large*
  - *a heavily loaded 9.6 Kbps link could be 127 times more costly than a lightly loaded 56 Kbps link*
  - *the routing algorithm would choose a 126-hop path of 56 Kbps links instead of a path consisting of one 9.6 Kbps link.*

ARPANET Metrics

The revised ARPANET metric

- *replaced delay measurement with link utilization*
- *compressed dynamic range*
  - *highly loaded link never has a cost more than 3 times its idle cost*
  - *most expensive link only 7 times the cost of the least expensive*
  - *high-speed satellite link more attractive than low-speed terrestrial link*
  - *cost is a function of link utilization only at moderate to high loads.*

ARPANET Metrics



Global Internet: Scalability Issues

IP "hides" the hosts in the address hierarchy, but. . .

- *Inefficient use of address space*
  - *class C network with 2 hosts (2/254 = 0.78% efficient)*
  - *class B network with 256 hosts (256/65534 = 0.39% efficient).*
- *There are too many networks*
  - *today's Internet has tens of thousands of networks*
  - *forwarding tables do not scale*
  - *route propagation protocols do not scale.*

Global Internet: Subnetting

- *add another level to address/routing hierarchy: subnet*
- *subnet masks define a variable partition of the host part of class A, B and C addresses*
- *subnets are visible only within the site.*

Network Number	Host Number

Class B address

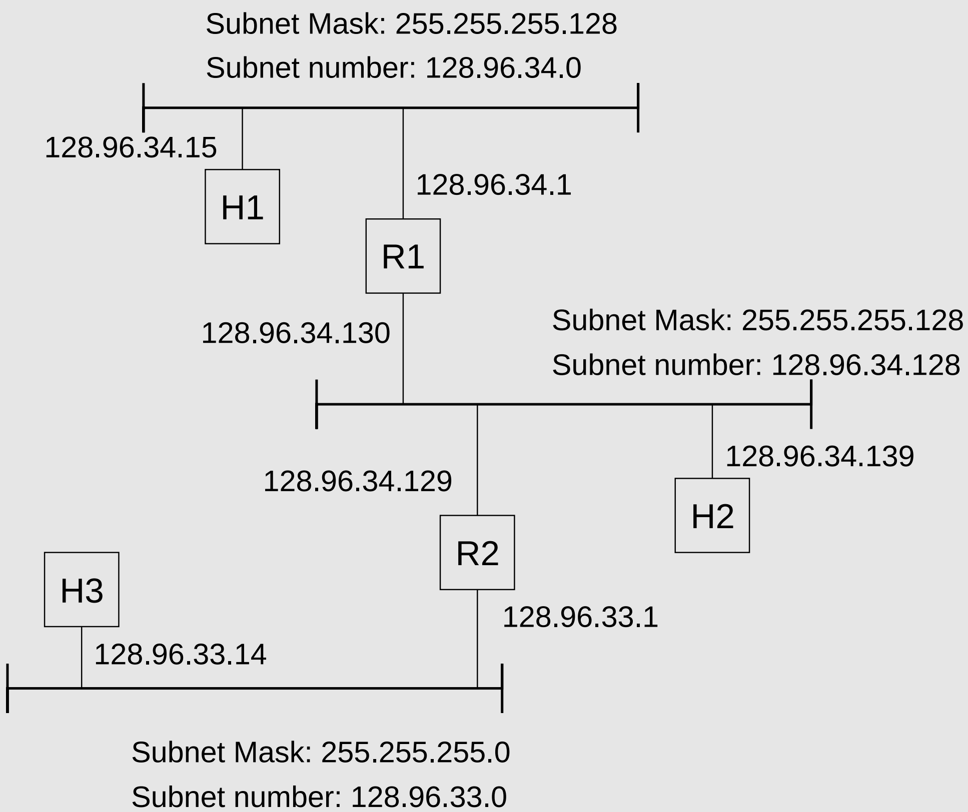
111111111111111111111111	00000000

Subnet Mask (255.255.255.0)

Network Number	Subnet ID	Host ID

Subnetted Address

Global Internet: Subnet Example



The forwarding table at router R1

Subnet Number	*Subnet Mask*	*Next Hop*
128.96.34.0	*255.255.255.128*	*interface 0*
128.96.34.128	*255.255.255.128*	*interface 1*
128.96.33.0	*255.255.255.0*	*R2*

Global Internet: Forwarding Algorithm

```
D = destination IP address
for each entry (SubnetNum, SubnetMask, NextHop)
    D1 = SubnetMask & D
    if (D1 == SubnetNum)
        if NextHop is an interface
            deliver datagram directly to destination
        else
            deliver datagram to NextHop (a router)
```

- *use a default router if nothing matches*
- *not necessary for all the 1s in the subnet mask to be contiguous*
- *multiple subnets can be put on one physical network*
- *subnets are not visible from the rest of the Internet.*

Global Internet: CIDR (RFCs 1517, 1518, 1519, 1520)

Classless Inter-Domain Routing CIDR does away with fixed class A, B, C network addresses

- *assign a block of contiguous network numbers to near-by networks*
- *each block is represented by a pair* `first_network_address/count`
  - `count` *is the number of bits in the network address: for example 150.100.252.0/22*
- *use a bit mask to identify the* `first_network_address`
  - *the remaining bits are the host field which may be subnetted*
- *all routers must understand CIDR addressing.*

CIDR: route aggregation

CIDR reduces the size of the forwarding tables.

Clients get IP addresses from an ISP. An ISP gets address blocks from an Internet Registry.

ISP-X owns block 180.180.0.0/16.

ISP-X assigns 180.180.1.0/24 to Client-A and 180.180.2.0/24 to client-B.

ISP-X's forwarding table contains entries for all subnets in 180.180.0.0/16, but ISP-X advertises only 1 prefix 180.180.0.0/16 to other AS's: route aggregation.

If Client-A changes to a new ISP then the block 180.180.1.0/24 is returned to ISP-X and a new block of addresses is acquired from ISP-Y . . .

IP Forwarding: Longest Prefix Match

Consider a forwarding table with entries

171.69
171.69.10

A packet set to 171.69.10.5 matches both entries. Use the longest prefix match to select 171.69.10.

A packet set to 171.69.20.5. Use the longest prefix match to select 171.69.

Global Internet: Route Propagation

A second hierarchical level is imposed on the network to improve the scalability of packet forwarding.

- *An Autonomous System is a collection of one or more networks under a single technical administration*
  - *an AS corresponds to an administrative domain*
  - *examples: University, company, backbone network*
  - *technical administration refers to aspects of the networking like routing policies . . .*
  - *each AS is assigned a unique identifier: a portion of its IP address.*

Global Internet: Route Propagation

Routing is divided into two parts using a two-level route propagation hierarchy

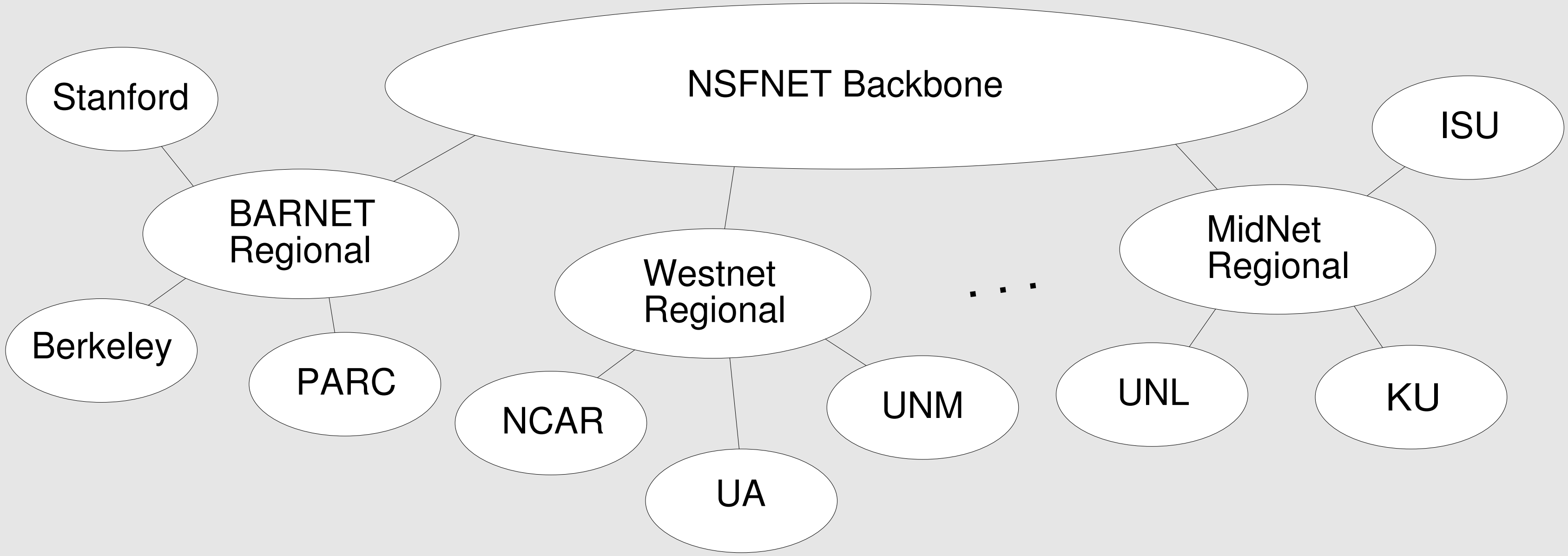
- *Routing within an AS: intra-domain routing*
  - *Interior Router Protocol IRP: each AS selects its own IRP*
  - *IRP is mostly driven by performance considerations.*
- *Routing between AS's: inter-domain routing*
  - *Exterior Router Protocol ERP: an Internet-wide standard*
  - *ERP depends on policy issues, economics, . . . as well as on performance.*

Global Internet: Interior Router Protocol

- *RIP: Route Information Protocol*
  - *distributed with Unix*
  - *distance-vector algorithm*
  - *based on hop-count.*
- *OSPF: Open Shortest Path First*
  - *link-state algorithm*
  - *based on link costs*
  - *supports multiple routing areas, load balancing & authentication.*

Global Internet: Exterior Gateway Protocol EGP

The Internet had an hierarchical structure.



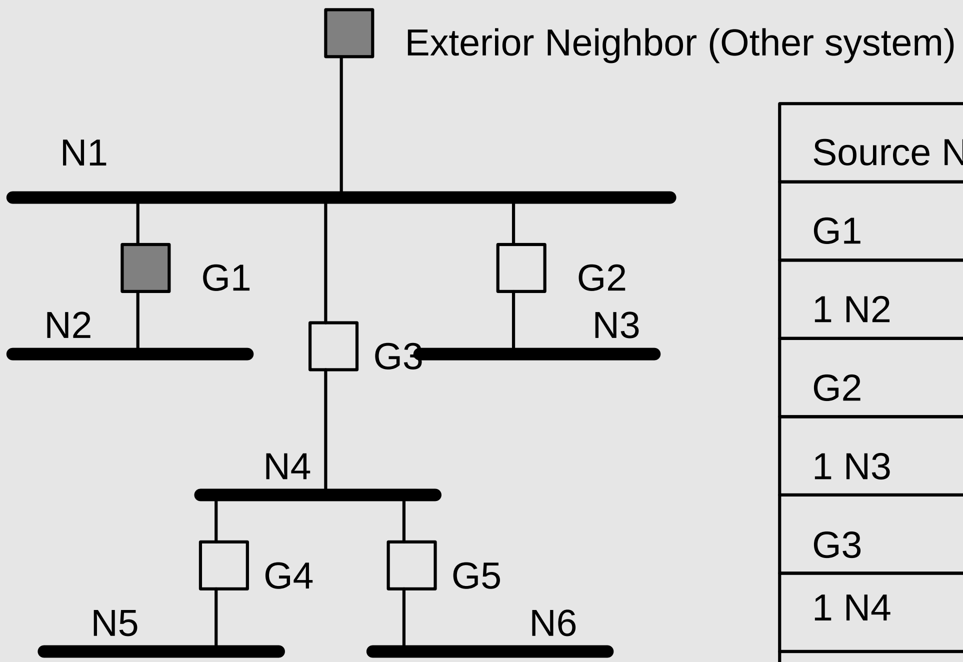
- *EGP was designed for a tree-structured Internet*
- *concerned with reachability, not with optimal routes.*

Global Internet: EGP

EGP protocol messages

- *neighbour acquisition: one router requests that another router be its peer; peers exchange reachability information*
- *neighbour reachability: one router periodically tests to see if the other router is still reachable; exchange HELLO/ACK messages; uses a k-out-of-n rule*
- *routing updates: peers periodically exchange their routing tables (distance-vector).*

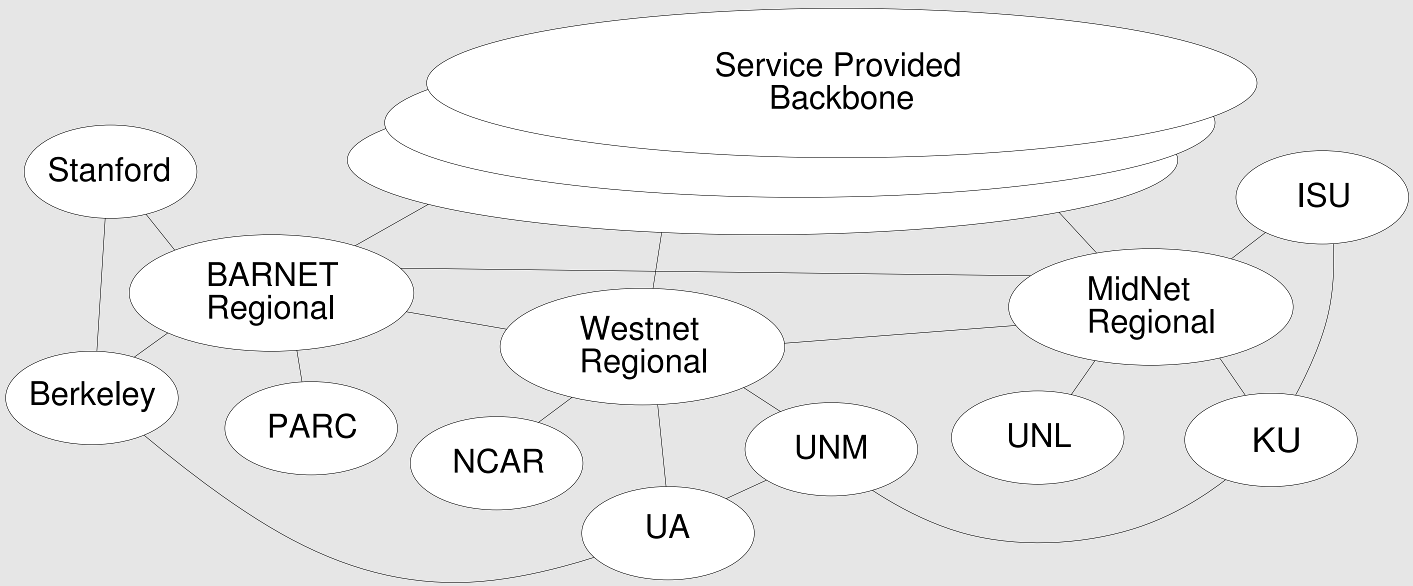
Global Internet: EGP Example



Source Net: N1
G1
1 N2
G2
1 N3
G3
1 N4
2 N5
2 N6

Global Internet: Border Gateway Protocol BGP-4

The Internet no longer has an hierarchical structure.



BGP-4 caters for arbitrarily inter-connected sets of AS's.

BGP-4 is concerned with reachability, not with optimal routes.

Global Internet: BGP-4

There are two types of traffic

- *local traffic originates/terminates at nodes within an AS*
- *transit traffic passes through an AS.*

There are three types of AS's

- *a stub AS has a single connection to one other AS. A stub AS carries local traffic only*
- *a multihomed AS has connections to more than one other AS, but refuses to carry transit traffic*
- *a transit AS has connections to more than one other AS, and can carry transit and local traffic.*

Global Internet: BGP-4 (RFC 1771)

Intra-domain routing finds optimal paths by minimizing some link metric.

Inter-domain routing is difficult: the goal is to find any loop-free path to the destination.

Inter-domain routing is difficult because

- *scale: a backbone router must be able to forward packets to any destination – some 70,000 address prefixes are needed in the routing table*
- *each AS runs its own IGP with its own link metrics – it may not be possible to compute the cost of a route that crosses multiple AS's*
- *trust: can AS A trust the routes advertised by AS B?*

BGP Functions

BGP implements inter-domain routing through

- *neighbour acquisition*
- *neighbour reachability*
- *network reachability.*

BGP features

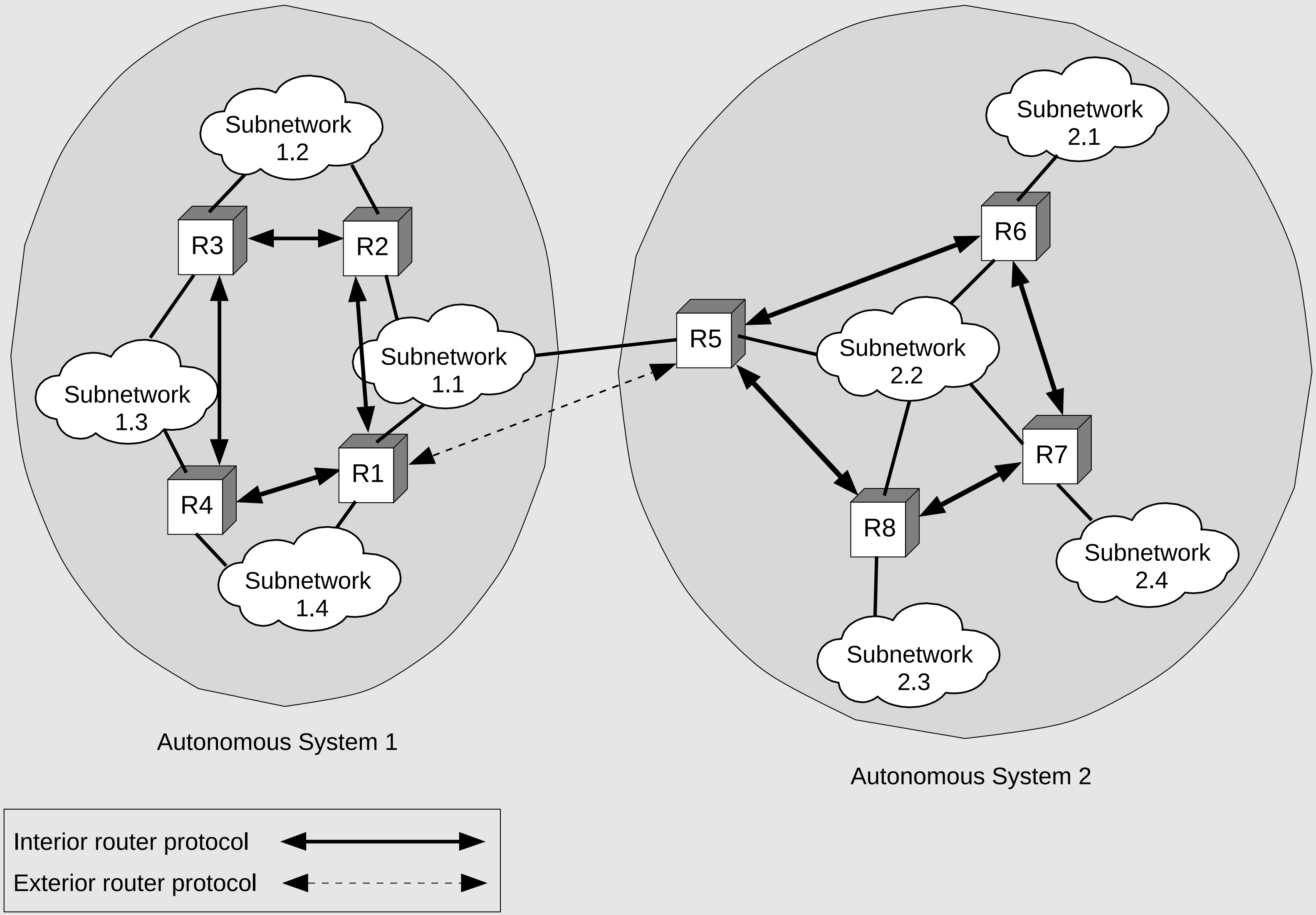
- *exchanges route information between AS's*
- *conveys information about AS path topology*
- *is a path vector protocol running over TCP*
- *implements incremental updates.*

Global Internet: BGP-4

Each AS has one or more border routers

- *The administration of the AS picks one border router to be a BGP speaker for the AS that advertises*
  - *local networks within its own AS*
  - *other reachable networks (transit AS's only).*
- *The BGP path information is sent over TCP connections in BGP-4 messages*
  - *open: open a neighbour relationship*
  - *update: define one new route and/or withdraw routes*
  - *keep alive: confirm the neighbour relationship*
  - *notification: an error condition is detected.*

BGP-4: Neighbour Acquisition



Neighbour routers are attached to the same network.

BGP-4: Neighbour Acquisition

- *Neighbour routers are attached to the same subnetwork.*
- *Neighbour acquisition allows neighbour routers in different AS's to regularly exchange routing information*
  - *why: a router may not want to take part in a neighbour relation: it may be too heavily loaded to carry the traffic of a neighbour AS*
  - *how: one of the border routers opens a TCP connection to send an open message to the other border router which may either accept (return a keep alive message) or refuse the offer.*
- *BGP4 does not consider how the border routers know each other's address – this is done by the AS administration at configuration time.*

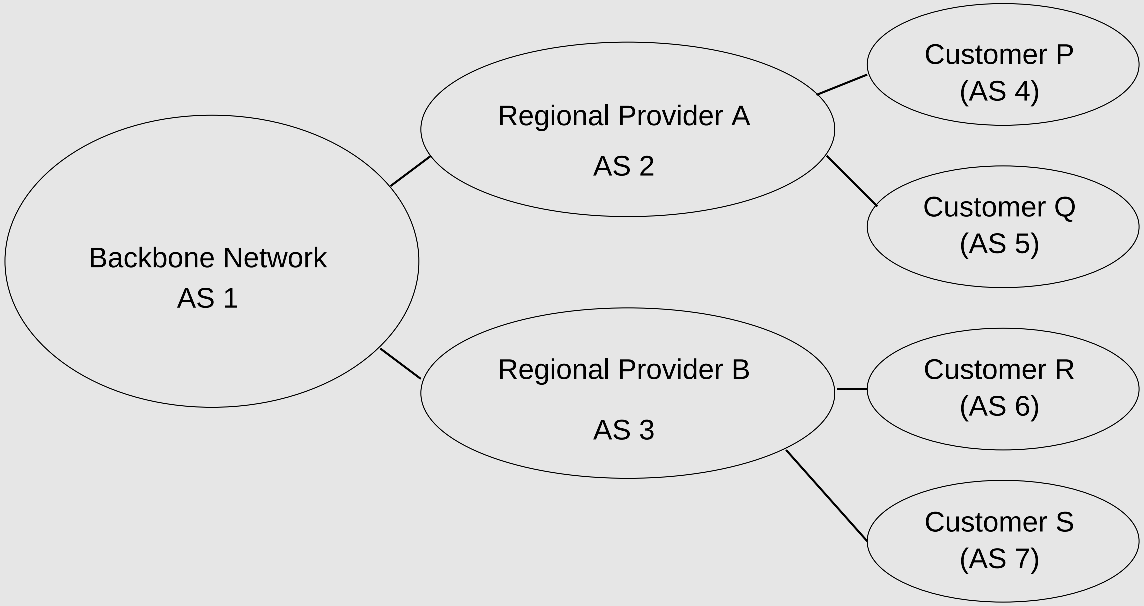
BGP-4: Neighbour Reachability

- *Neighbour reachability is used to maintain the relationship between neighbour routers.*
  - *why: each router needs to be assured that the other partner still exists and is still engaged in the neighbour relationship*
  - *how: send periodic keep alive messages to each other.*

BGP-4: Network Reachability

- *Each border router maintains a database of subnetworks that it can reach and the preferred route for each subnetwork.*
  - *BGP4 is a path-vector algorithm, not a distance-vector algorithm.*
- *Initially exchange routing tables (once only), further modifications are incremental.*
- *Broadcast update messages to all other routers implementing BGP when the database is changed (route additions and withdrawals)*
- *Loop-free: if a router R receives an update message which includes router R in the advertised path, the update is not forwarded to the other routers.*
- *Errors are reported by notification messages.*

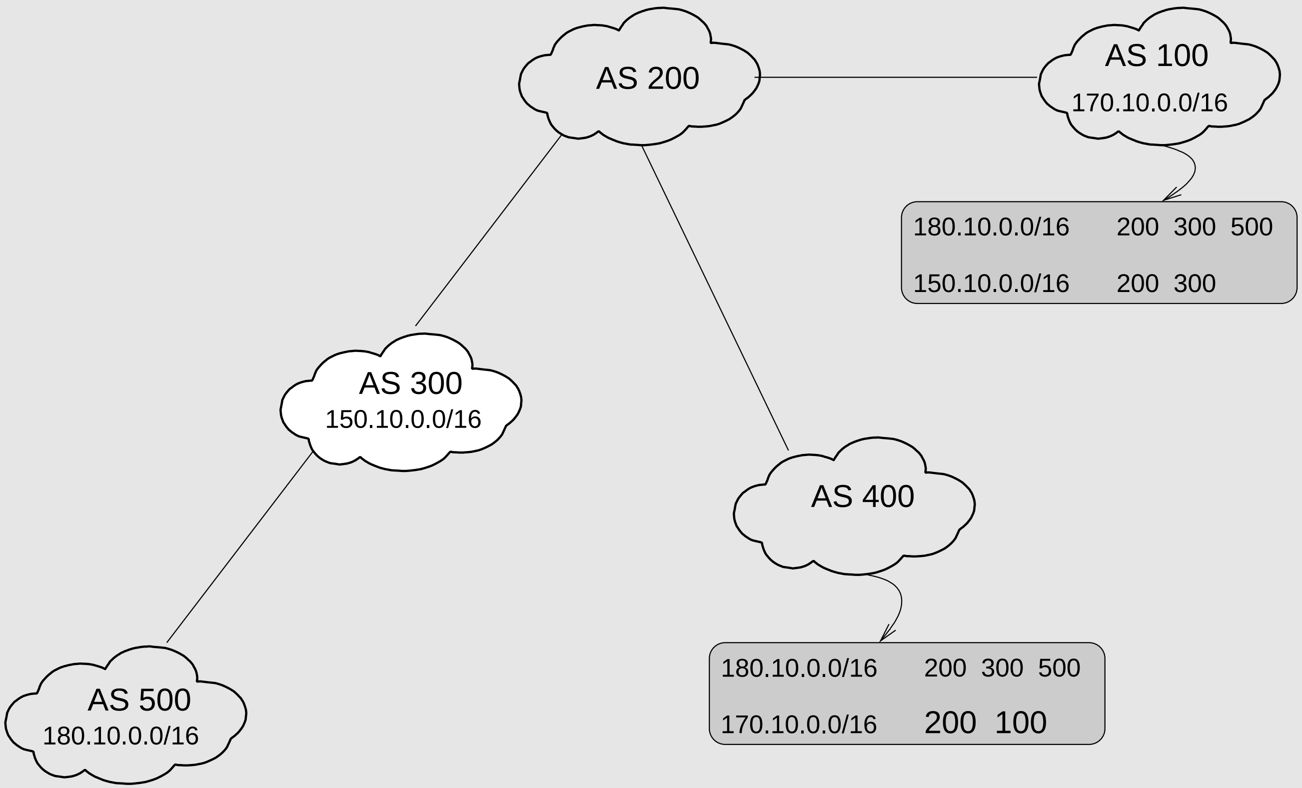
BGP-4: Example



- *The speaker for AS2 advertises that all the networks in AS4 & AS5 can be reached directly from AS2.*
- *The speaker for AS1 then advertises that all the networks in AS4 & AS5 can be reached along the path (AS1, AS2).*
- *Speakers can cancel previously advertised paths.*

BGP-4: AS_PATH

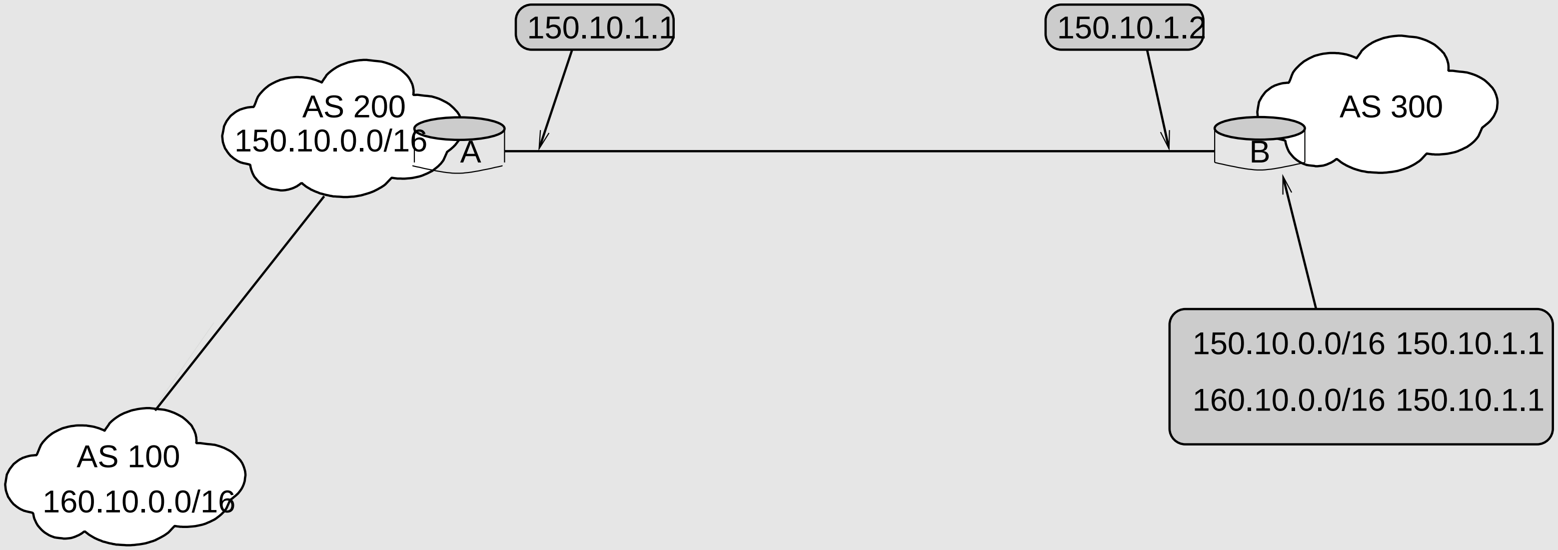
The AS_PATH is a list of AS's traversed by a path to a network.



The AS_PATH is used to select the best path from among several paths to a network. The AS_PATH is also used to implement policy-based routing.

BGP-4: eBGP and Next Hop

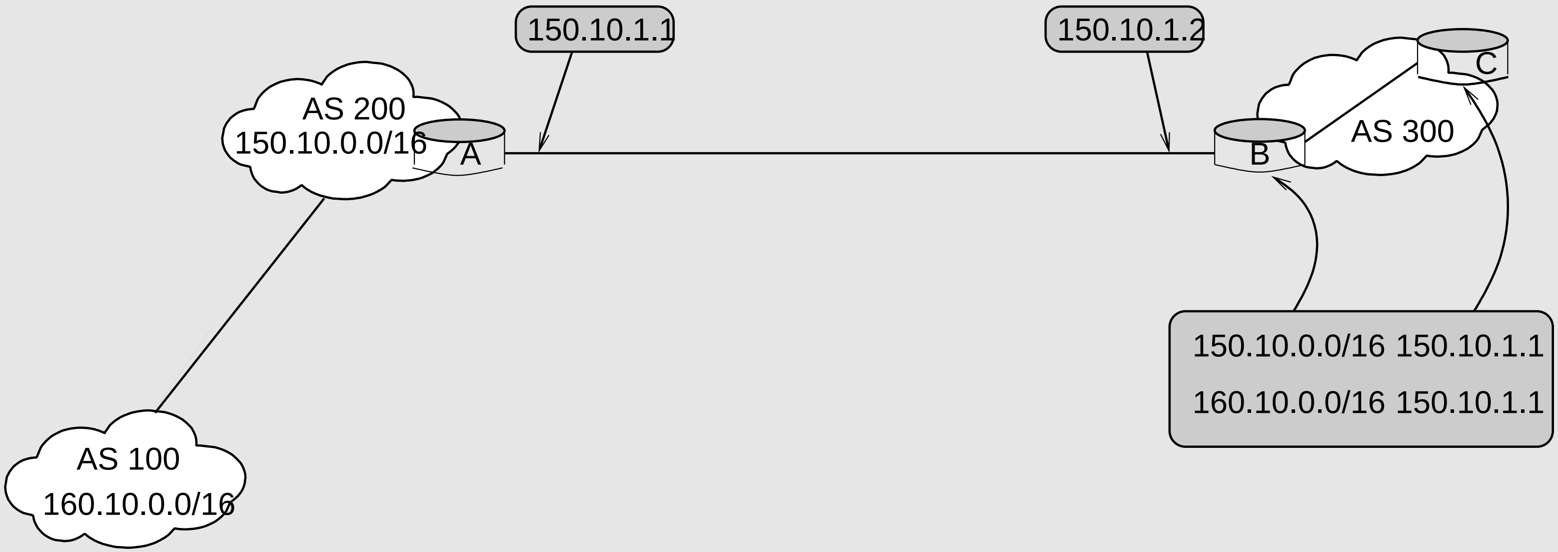
Next hop: the IP address of the border router that should be used for the next hop of this path.



BGP-4: iBGP and Next Hop

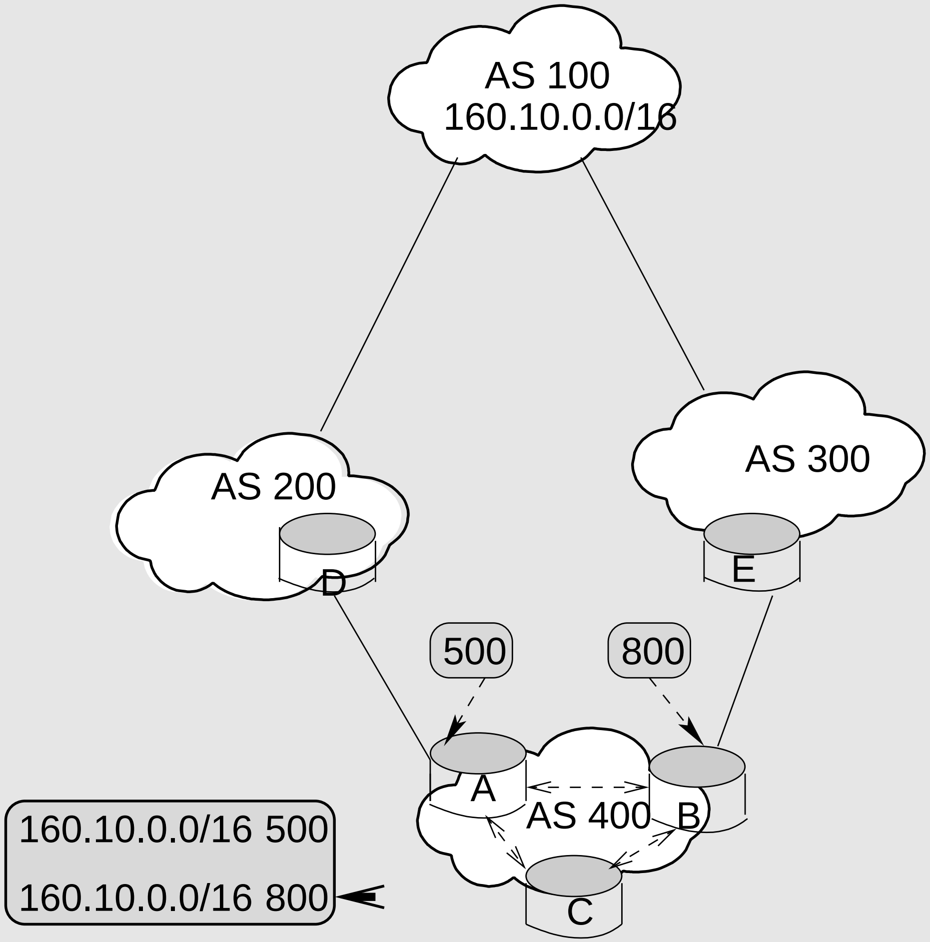
iBGP: an application of the BGP protocol within an AS to carry exterior-routing information.

iBGPis used to propagate BGP tables throughout an AS.



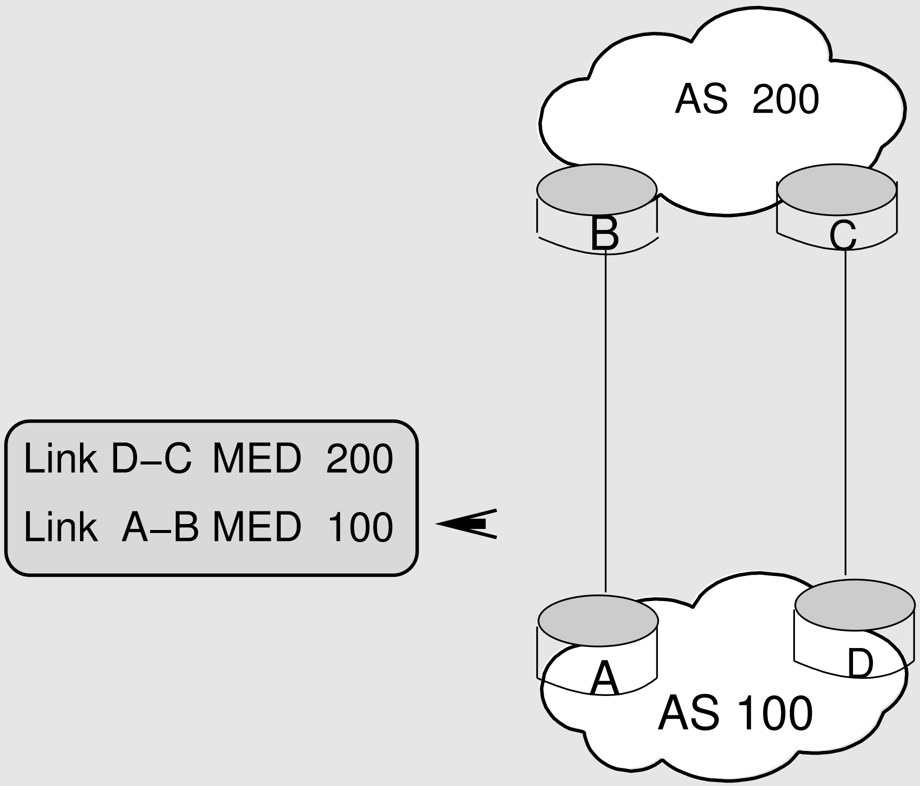
BGP-4: Local Preference

Local preference: used to inform other BGP speakers in the same AS about preferences for a particular route.



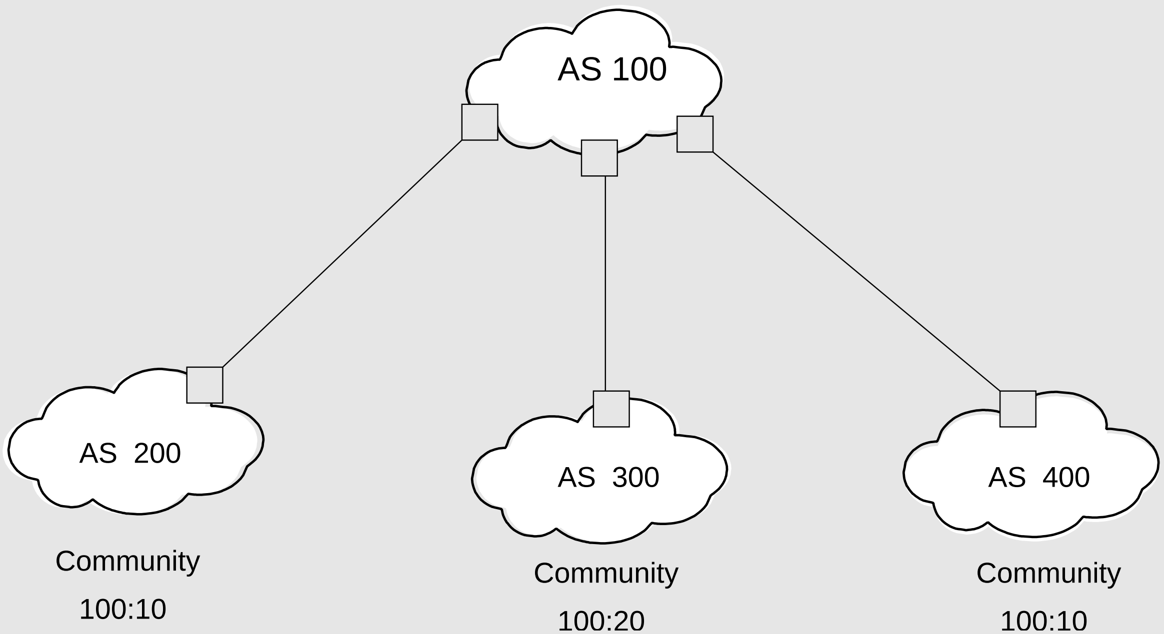
BGP-4: Multi-Exit Discriminator

*Multi-exit discriminator**: used to choose among multiple exit points in neighbouring AS's.*



BGP-4: BGP Communities

BGP communities are used to make policy decisions by grouping destinations into communities and applying policy decisions based on the BGP community attribute instead of directly on the prefixes.



IPv6: Major Features

Motivation: to deal with the scaling problems caused by the Internet's growth

- *128-bit addresses: support billions of hosts*
- *multicast*
- *simplify the protocol: routers process packets faster*
- *authentication and security*
- *autoconfiguration*
- *smaller routing tables*
- *protocol extensions: support for resource allocation*
- *mobile hosts can roam with changing their addresses*
- *migration: IPv4 & IPv6 can co-exist.*

Most of these features are now included in IPv4.

IPv6: Addresses

- *128-bit address:* 3.4×10^{38} *nodes,* 6×10^{23} *addresses per square meter of the earth's surface*
- *Classless addressing/routing (similar to CIDR)*
- *Notation: x:x:x:x:x:x:x:x (x = 16-bit hex number)*
  - *contiguous 0's are compressed 47CD::A456:0124*
  - *IPv4 compatible IPv6 address ::128.42.1.87*
  - *IPv4 mapped IPv6 address ::FFFF:128.42.1.87*
- *Address allocation*
  - *there are various categories of addresses*
    - *provider-based unicast address.*

IPv6: Provider-Based Unicast Address

3	m	n	o	p	125–m–n–o–p
010	Registry ID	Provider ID	Subscriber ID	Subnet ID	Interface ID

- `RegisteryID` *identifies the registration authority which assigned the provider portion of the address.*
- `ProviderID` *identifies the internet service provider which assigned the subscriber portion of the address*
- `SubscriberID` *distinguishes among multiple subscribers attached to the provider portion of the address.*
- `SubnetID` *identifies the subnet.*
- `InterfaceID` *identifies a single node interface in the subnet.*

IPv6: Address Allocation

Prefix	*Use*
0000 0000	*Reserved*
0000 0001	*Unassigned*
0000 001	*Reserved for NSAP Allocation*
0000 010	*Reserved for IPX Allocation*
0000 011	*Unassigned*
0000 1	*Unassigned*
0001	*Unassigned*
001	*Unassigned*
010	*Provider-Based Unicast Address*
011	*Unassigned*
100	*Reserved for Geographic-Based Unicast Addresses*
. . .	*. . .*

IPv6: Address Allocation

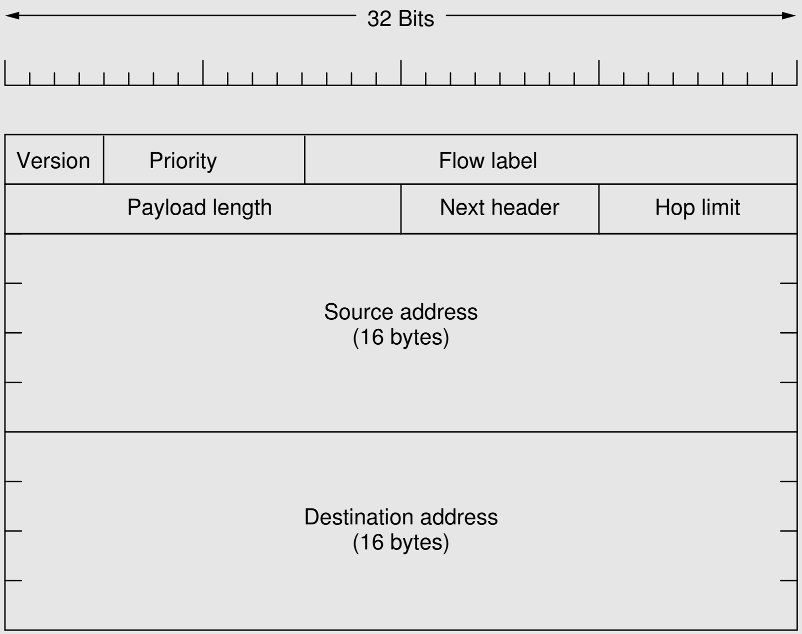
Prefix	Use
. . .	. . .
100	Reserved for Geographic-Based Unicast Addresses
101	Unassigned
110	Unassigned
1110	Unassigned
1111 0	Unassigned
1111 10	Unassigned
1111 110	Unassigned
1111 1110 0	Unassigned
1111 1110 10	Link Local Use Addresses
1111 1110 11	Site Local Use Addresses
1111 1111	Multicast Addresses

IPv6: Address Allocation

Prefix	Use
. . .	. . .
1111 1110 10	Link Local Use Addresses
1111 1110 11	Site Local Use Addresses
1111 1111	Multicast Addresses

- A link local address can be used on a single link or subnet (not a global address). Used during autoconfiguration.
- A site local address can be used on a network of subnets that are not connected to the Internet now but may be connected later.

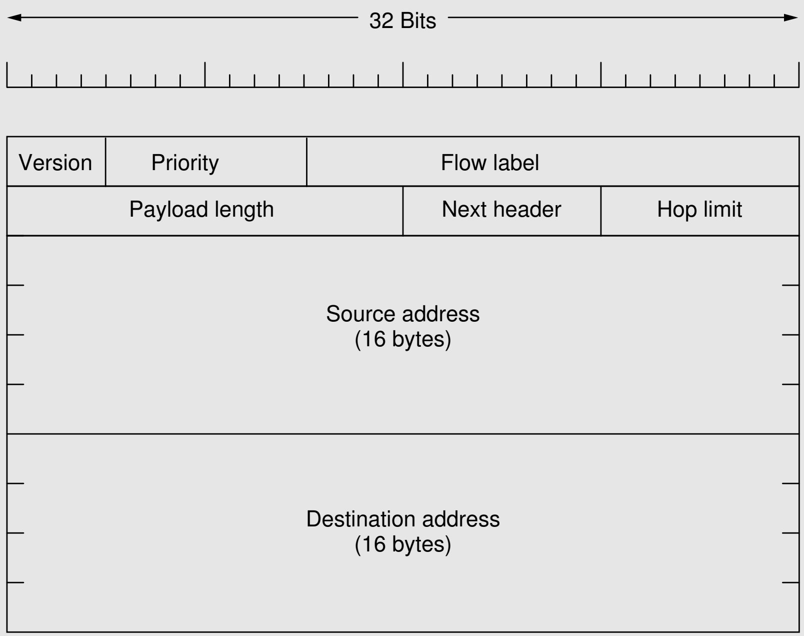
IPv6 Header



- `Priority`: *0–7 for congestion-aware services, 8–15 for delay–aware services*
- `Flow label`: *implements VCs*

No checksum: must be provided by the transport layer.

IPv6 Header



The `Next header` *field identifies the type of the next extension header.*

If no extension header follows then `Next header` *identifies the transport layer header (e.g.* `Next header=6`*: a TCP header follows) encapsulated in the IPv6 payload.*

IPv6 Header

- *40-byte "base" header*
- *Extension headers follow in a fixed order, mostly of fixed length*
  - *hop-by-hop options*
  - *final & intermediate destination options*
  - *source routing*
  - *fragmentation*
  - *authentication*
  - *security*
  - *final destination options.*

IPv6 Header: Flow Labels

Flow labels can be used to support Quality of Service.

A flow is a sequence of packets from a source to a destination for which the source requests special handling by the intervening routers.

A flow is typically a sequence of packets from a single application having the same transfer service requirements. An application may generate multiple flows.

Flow requirements are set up prior to flow commencement. A unique flow label is assigned to each flow.

A router may treat packets from different flows differently: resource allocation, discard policy, security, . . .

IPv6 Header: Autoconfiguration

Each host on the internet needs an IP address.

IPv4: DHCP automates the allocation of IP addresses.

IPv6: autoconfiguration is even easier

- *the link-level address (e.g. the ethernet address) is used as the interface-id part of the 128-bit IP address*
- *the correct address prefix for the subnet is obtained using a DHCP-like protocol*
    - *unlike IPv4 DHCP, IPv6 DHCP is stateless: the DHCP server retains no knowledge of the permananet IP addresses that it issues*
    - *stateful IP address creation is also permitted to issue temporary IP addresses.*

IPv6 Header: Network Address Translation (RFC 1918)

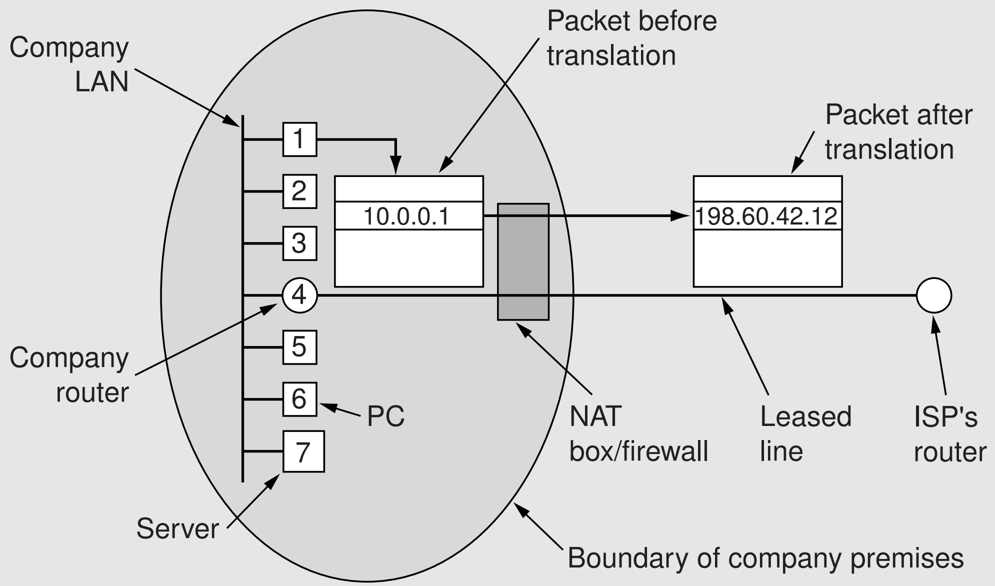
Many hosts on an intra-net may need to communicate with each other but not with the global Internet.

Such hosts can be assigned a private IP address that is not globally unique, but that is unique within the intra-net.

- A $10.x.x.x$ previously used by ARPANET
- B $172.16.x.x$ to $172.31.x.x$
- C $192.168.0.x$ to $192.168.255.x$

Note: NAT may seriously delay the widespread use of IPv6.

IPv6 Header: Network Address Translation



A host can communicate outside the intra-net using a NAT-box to translate its NAT address to a globally unique IP address. Likewise incoming IP addresses are translated to NAT addresses. The NAT-box usually has a small pool of unique IP address – not all intra-net hosts need to communicate globally.

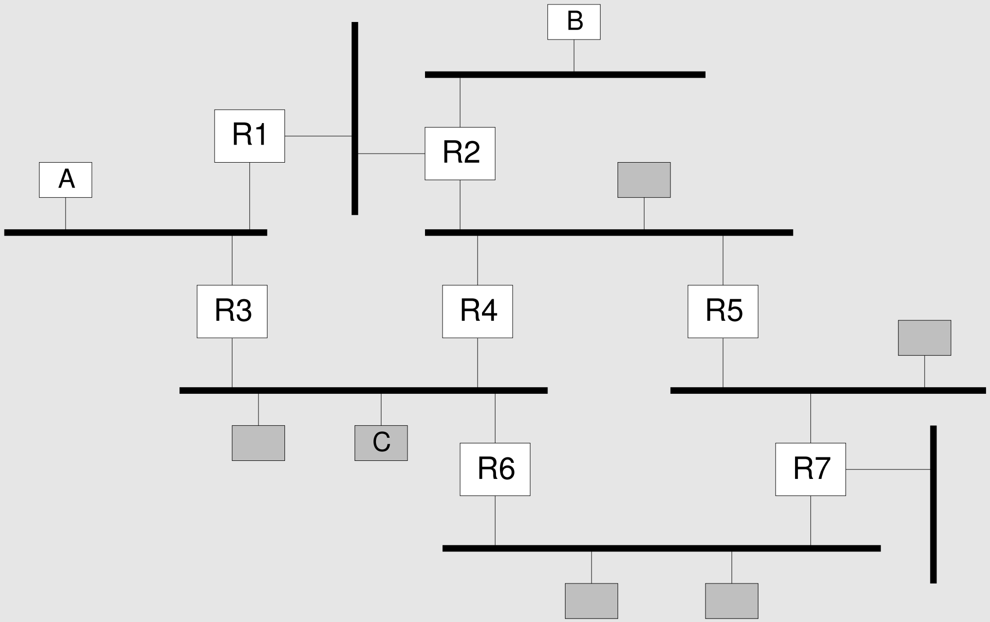
Internet Multicast: Overview

- *IPv4*
  - *class D addresses*
  - *demonstrated with MBone*
  - *uses tunneling*
- *Integral part of IPv6*
  - *problem is making it scale*

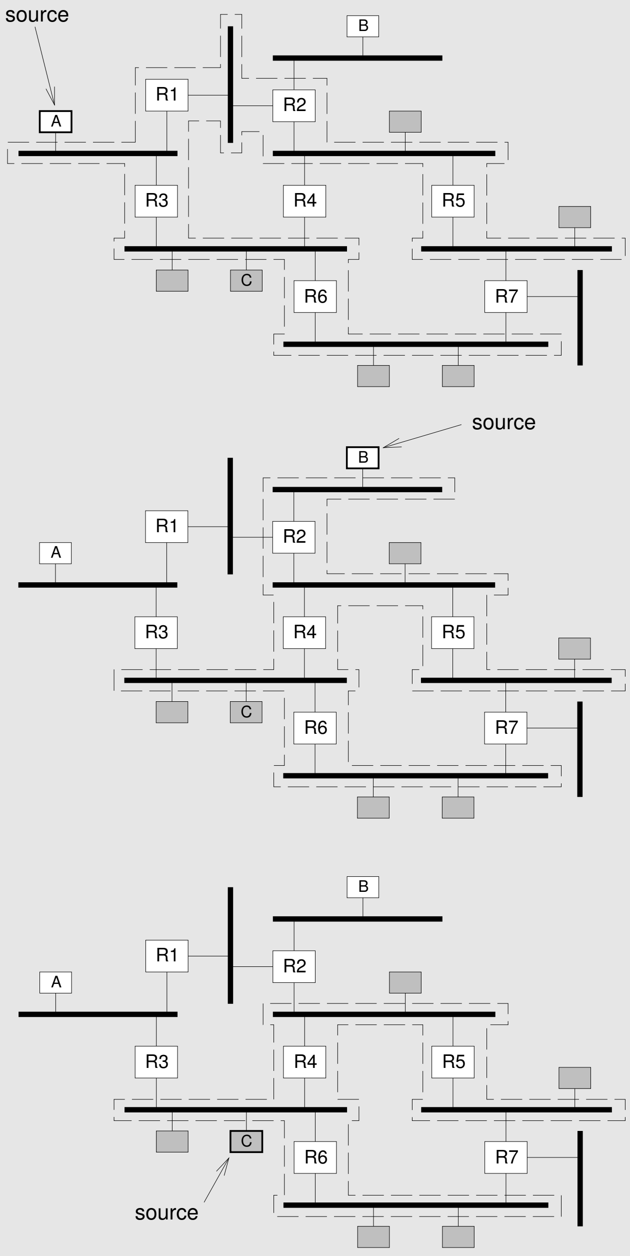
Internet Multicast: Link-State Multicast

- *Each host on a LAN periodically announces the groups it belongs to (IGMP).*
- *Augment update message (LSP) to include set of groups that have members on a particular LAN.*
- *Each router uses Dijkstra's algorithm to compute shortest-path spanning tree for each source/group pair.*
- *Each router caches tree for currently active source/group pairs.*

Internet Multicast: Example



Internet Multicast: Example



Internet Multicast: Distance-Vector Multicast

Reverse Path Broadcast (RPB)

- *Each router already knows that shortest path to destination S goes through< router N.*
- *When receive multicast packet from S, forward on all outgoing links (except the one on which the packet arrived), iff packet arrived from N.*
- *Eliminate duplicate broadcast packets by only letting "parent" for LAN (relative to S) forward*
  - *shortest path to S (learn via distance vector)*
  - *smallest address to break ties*

Internet Multicast: Reverse Path Multicast (RPM)

- *Goal: Prune networks that have no hosts in group G*
- *Step 1: Determine of LAN is a leaf with no members in G*
  - *leaf if parent is only router on the LAN*
  - *determine if any hosts are members of G using IGMP*
- *Step 2: Propagate “no members of G here” information*
  - *augment* $\langle$`Destination, Cost`$\rangle$ *update sent to neighbors with set of groups for which this network is interested in receiving multicast packets.*
  - *only happens with multicast address becomes active.*