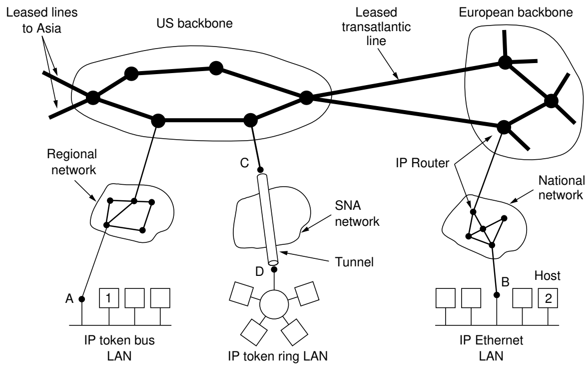
The Internet Protocol (IP)

The Internet: view it as a collection of **autonomous systems** connected together by a bunch of **backbones**:

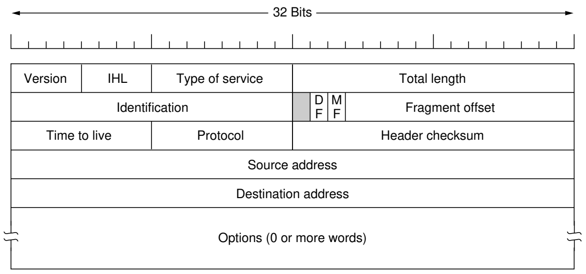


Internet Model

Application
Transport
Network
Host–to–network

- An application offers a data stream to the transport layer, using either connection-oriented or connectionless services.
- The transport layer breaks up the data stream into datagrams, and passes these to the network layer.
- The datagrams are routed through the internet, occasionally fragmented when needed. Routers always pass the datagram to the underlying data link layer (generally LANs and (dial-up or leased) telephone lines).

IP Header



IHL	Length of the header
TOS	3-bit priority plus 3-bit flag (delay, throughput, reliability)
TLen	Length header + payload
ID	Datagram id
DF	Don't fragment this datagram
MF	There are more fragments after this one
FOff	Offset in original datagram where this fragment belongs.
TTL	Maximum number of hops allowed
Prot	Globally defined ids for transport prot.
HCS	Checks the headers (has to be calculated at each hop)

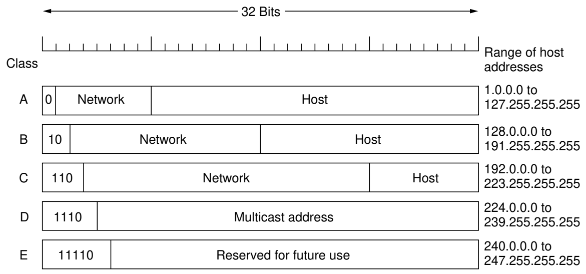
IP Options

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

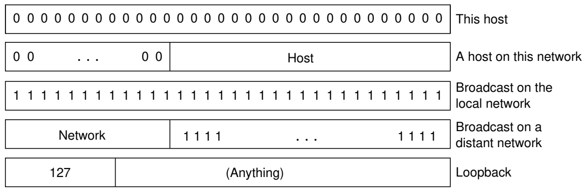
- Security is hardly used: specifying that a datagram is "really top secret" is not such a good idea.
- Source routing can be effectively used to force routes: debugging, politics, and mobile hosts.
- Record route and Timestamping are mainly used for debugging.

Note: Not all routers actually support these options (as noticed when people really wanted to use loose source routing).

IP Addresses



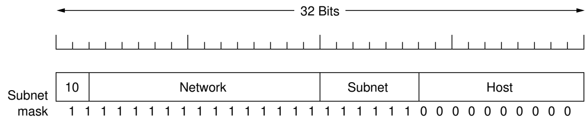
Class	Max. networks	Max. hosts/network
A	126	16,777,214
B	16,382	65,536
C	2,097,150	254



Subnet Masking

Problem: All hosts on the same network must have the same network number. This may cause a single organization to acquire several classes of addresses (e.g. each time it adds a LAN), that would subsequently need to be announced worldwide (**WHY?**).

Solution: Use a single network address for the entire organization, and internally divide the host address space into a **subnet address** and a host id:



Note: We've introduced a 3-level routing hierachy.

ICMP

Basic idea: We need to inform hosts and routers when things go wrong, or, likewise, should be able to send queries to get status information. Encapsulate control messages in normal IP datagrams.

- Distinguish several message types (destination unreachable, time exceeded, etc.)
- Each message type is further specified by a code. Example: a destination can be unreachable (message type = 3) because of an unsupported protocol (code = 2).
- An ICMP message includes the header of the IP datagram that caused the message to be sent.

Question: What should we do when an ICMP message is lost?

Question: How can be *recognize* ICMP messages when they are encapsulated in IP datagrams?

Address Resolution Protocol

Situation: Addressing hosts using IP addresses is great, but these addresses are not recognized by the hardware of those hosts. Example: a host on an Ethernet LAN will only read messages encapsulated in frames containing that host's hardware address.

Problem: How do we find out the hardware (i.e. datalink) address of a host, given its Internet address?

1. *Router:* Ask each host on the LAN whether they have the requested IP address. This is done by encapsulating the query as an ARP message in a datalink frame, and broadcasting it.
2. *Host:* Recognizes it is dealing with an ARP message, checks whether it has the requested address, and if so, sends a reply back with its datalink address. **Question:** how can the host recognize an ARP message?
3. *Router:* Recognizes a reply ARP message, and (generally) caches the IP address with the datalink address. It can then forward IP datagrams to the correct host, encapsulating them in datalink frames.

Question: what should the router do when no one replies?

Reverse Address Resolution Protocol

Problem: So how does a host know its own IP address?

Solution: The host uses a limited broadcast (i.e. restricted to its own network) to query a RARP server for its IP address. The RARP server maintains a table of (datalink address,IP address) mappings.

Note: we need a RARP server for every broadcast segment.

IP Routing

Basic idea: Make a distinction between routing *in* an autonoumous system, and *between* autonomous systems:

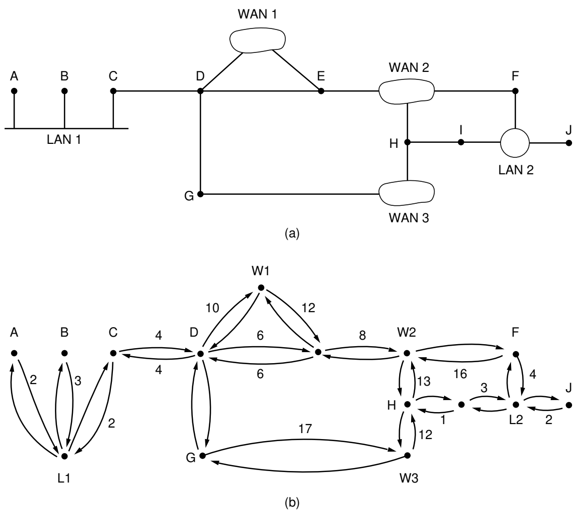
- Intra-AS routing is concerned with getting packets from source to destination. It should do this as best as possible (optimal routing). **Interior Gateway Routing**.
- Inter-AS routing has to deal with a lot of politics. For example, some ASes should not be traversed at all, whereas some do not accept "foreign" packets. **Exterior Gateway Routing**.

Interior Gateway Routing: OSPF

Open Shortest Path First: link state routing protocol, replacement for (still widely used, but inadequate) RIP (routing information protocol). Requirements:

- Openness: the algoritm should be made publicly available so that anyone could implement it.
- Support for different distance metrics (hops, delays, costs, etc.)
- Dynamic and efficient adaptability to changing topologies.
- Support routing based on type of service (already specified in IP, but no one actually used it). Especially important for real-time (multimedia) traffic.
- Support for load balancing: when a route is heavily used, another one should be selected.
- Support hierarchical routing.
- Offer security.
- Support IP tunneling.

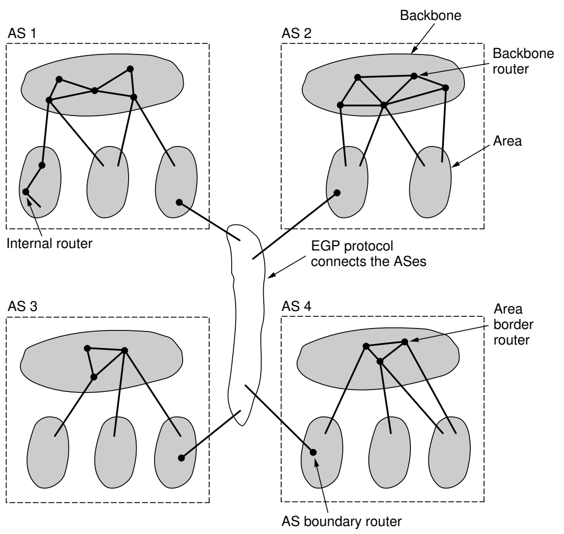
OSPF – Routing Graphs



Note: OSPF build different graphs, depending on the distance metrics it uses (delay, throughput, reliability). Of course, physical links are always taken into account.

Note: Each LAN is represented by a **designated router** that acts as a representative of the other routers on that LAN.

OSPF – Hierarchical Routing

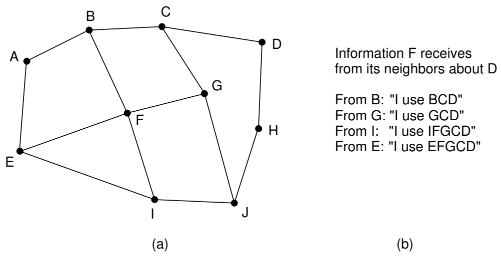


Note: We can use the same algorithm for the areas and the backbone. In fact, the backbone behaves as just another area. This means it should be contiguous.

Exterior Gateway Routing: BGP

Basic idea: To ensure that routing policies are met, they are generally configured manually into each BGP router. In other words: routers do not automatically use the routes they find, but have to check whether it is allowed. Neighboring routers maintain a *connection* to simplify message reliability.

Routing: is based on distance vector routing, but paths rather than distances are announced:



Note: we do not have a count–to–infinity problem here, because a router decides and an entire path.

Classless Interdomain Routing

Problem: Although there are "enough" IP addresses, the use of classes is rapidly making them a scarce resource. The main problem is that too many class B addresses are being used. At the same time, we can't simply assign class C addresses without adequate routing table management (**HUH!?**).

Solution: Assign class C addresses in contiguous blocks of 256 addresses. Also, partition the address space into four zones:

194.0.0.0	–	195.255.255.255	Europe
198.0.0.0	–	199.255.255.255	North America
200.0.0.0	–	201.255.255.255	Central/South America
202.0.0.0	–	203.255.255.255	Asia and the Pacific

A series of contiguous blocks is assigned, together with a **mask**...

CIDR – cnt'd

Site	# Addr.	Range	Mask
S1	2048	194.24.0.0 – .7.255	255.255.248.0
S2	4096	194.24.16.0 – .31.255	255.255.240.0
S3	1024	194.24.8.0 – .11.255	255.255.252.0

Update all routers (in Europe) with three entries, each consisting of a (base address, mask)-pair. When an IP datagram arrives at a router, the latter tries to find the *base address* by using the masks:

$$\begin{array}{ll} (194.24.17.4\&\&255.255.248.0) \neq 194.24.0.0 & \surd \\ (194.24.17.4\&\&255.255.240.0) \neq 194.24.16.0 & \surd \\ (194.24.17.4\&\&255.255.252.0) = 194.24.8.0 & OK \end{array}$$

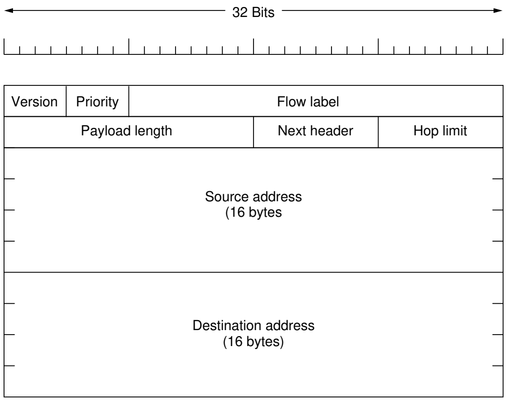
Note: The assignment is not unique in the sense that an address may match two or more entries. This is caused by so-called **multi-homed** hosts, i.e. hosts with more than one IP address. This is not further explained in Tanenbaum.

IPv6

Problem: The current version of IP can not support enough addresses, but also lacks the flexibility to act as the basis for a large variety of users. The goals:

- Support billions of hosts.
- Reduce the size of routing tables.
- Simplify the protocol to enable faster routers.
- Provide (better) security.
- Better support for type of service.
- Support scopes with multicasting.
- Support roaming hosts without address changes.
- Coexistence of the old and new protocol, and evolution of the new one.

IPv6 – Header



Note: The flow label is used to set up a pseudoconnection between source and destination. It identifies a flow for which, for example, bandwidth has been reserved.

Note: A simpler header is almost impossible – further info is provided by *next headers*.

Note: No checksum, and no fragmentation fields.

IPv6 – Address Space

Big difference: IPv6 uses 16-byte addresses. This is really a lot: 7×10^{23} addresses per square meter. It does allow us to be less efficient with address allocation: 72% is unassigned.

Prefix	**Usage**	**Fraction**
0000 0000	Reserved (incl. IPv4)	1/256
0000 0001	Unassigned	1/256
0000 001	OSI NSAP addresses	1/128
0000 010	Novell Netware IPX addresses	1/128
0000 011	Unassigned	1/128
0000 1	Unassigned	1/32
0001	Unassigned	1/16
001	Unassigned	1/8
010	Provider-based addresses	1/8
011	Unassigned	1/8
100	Geographic-based addresses	1/8
101	Unassigned	1/8
110	Unassigned	1/8
1110	Unassigned	1/16
1111 0	Unassigned	1/32
1111 10	Unassigned	1/64
1111 110	Unassigned	1/128
1111 1110 0	Unassigned	1/512
1111 1110 10	Link local use addresses	1/1024
1111 1110 11	Site local use addresses	1/1024
1111 1111	Multicast	1/256

IPv6 – Extension Headers

Basic idea: Keep the main header as simple as possible, and provide any further information in an (optional) extension header:

Ext. header	**Description**
Hop–by–hop options	Information for routers
Routing	Full or partial route to follow
Fragmentation	Management of datagram fragments
Authentication	Verification of the sender's identity
Encrypted payload	Info on the encrypted contents
Destination options	Additional info for destination

Most of them are quite obvious – see Tanenbaum for further details.

Important: Note that fragementation is still supported, but that only the source host can do it. Routers never fragment datagrams anymore.

IPv6 – Security

Illustrative example: There was a lot of discussion on where and how to incorporate security in IPv6:

- If you are really concerned about security, would you trust anything else but end–to–end encryption? **Question:** what does this mean!?
- Having security in the network layer offers a generally useful service to many applications. Those that don't want to use it, just ignore it.
- Network-layer protocols have to run in every country. Some countries disallow cryptosystems that the government can't decrypt easily.
- Are the default crypto-algorithms good enough? For example, MD5 has recently been cracked.

The main issue here, as with almost every protocol, is to decide in which layer we should put functionality. There are many people who argue that only end–to–end solutions should be applied. The rest (i.e. general solutions) will never be good enough.