

Computer Network Management

- textbook -

1. Routing Basics

The Internet system consists of a number of interconnected packet networks supporting communication among host computers using the Internet protocols. These protocols include the Internet Protocol (IP), the Internet Control Message Protocol (ICMP), the Internet Group Management Protocol (IGMP), and a variety transport and application protocols that depend upon them.

All Internet protocols use IP as the basic data transport mechanism. IP is a datagram, or connectionless, internetwork service and includes provision for addressing, type-of-service specification, fragmentation and reassembly, and security. ICMP and IGMP are considered integral parts of IP, although they are architecturally layered upon IP. ICMP provides error reporting, flow control, first-hop router redirection, and other maintenance and control functions. IGMP provides the mechanisms by which hosts and routers can join and leave IP multicast groups.

Reliable data delivery is provided in the Internet protocol suite by Transport Layer protocols such as the Transmission Control Protocol (TCP), which provides end-end retransmission, resequencing and connection control. Transport Layer connectionless service is provided by the User Datagram Protocol (UDP).

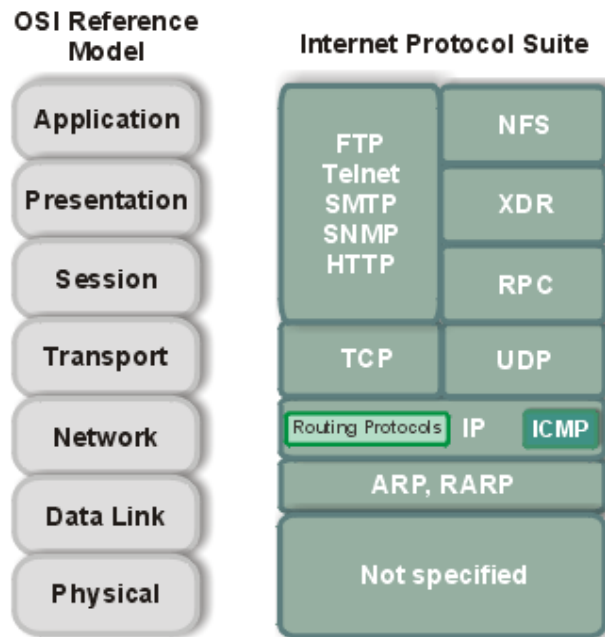


Figure 1.1: Internet protocols span the complete range of OSI model layers.

Internet protocols were first developed in the mid-1970s, when the Defense Advanced Research Projects Agency (DARPA) became interested in establishing a packet-switched network that would facilitate communication between dissimilar computer systems at research institutions. With the goal of heterogeneous connectivity in mind, DARPA funded

research by Stanford University and Bolt, Beranek, and Newman (BBN). The result of this development effort was the Internet protocol suite, completed in the late 1970s.

TCP/IP later was included with Berkeley Software Distribution (BSD) UNIX and has since become the foundation on which the Internet and the World Wide Web (WWW) are based.

Documentation of the Internet protocols (including new or revised protocols) and policies are specified in technical reports called Request For Comments (RFCs), which are published and then reviewed and analyzed by the Internet community. Protocol refinements are published in the new RFCs. To illustrate the scope of the Internet protocols, Figure 1.1 maps many of the protocols of the Internet protocol suite and their corresponding OSI layers. This chapter addresses the basic elements and operations of these and other key Internet protocols.

1.1 Internet Protocol

The Internet Protocol is the building block of the Internet. Its functions include:

- Defining the datagram, which is the basic unit of transmission in the Internet
- Defining the Internet addressing scheme
- Moving data between the Network Access Layer and the Host-to-Host Transport Layer
- Routing datagrams to remote hosts
- Performing fragmentation and re-assembly of datagrams

Before describing these functions in more detail, let's look at some of IP's characteristics. First, IP is a connectionless protocol. This means that IP does not exchange control information (called a "handshake") to establish an end-to-end connection before transmitting data. In contrast, a connection-oriented protocol exchanges control information with the remote system to verify that it is ready to receive data before any data is sent. When the handshaking is successful, the systems are said to have established a connection. Internet Protocol relies on protocols in other layers to establish the connection if they require connection-oriented service.

IP also relies on protocols in the other layers to provide error detection and error recovery. The Internet Protocol is sometimes called an unreliable protocol because it contains no error detection and recovery code. This is not to say that the protocol cannot be relied on - quite the contrary. IP can be relied upon to accurately deliver your data to the connected network, but it doesn't check whether that data was correctly received. Protocols in other layers of the TCP/IP architecture provide this checking when it is required.

1.1.1 The datagram

The TCP/IP protocols were built to transmit data over the ARPANET, which was a packet switching network. A packet is a block of data that carries with it the information necessary to deliver it - in a manner similar to a postal letter, which has an address written on its envelope. A packet switching network uses the addressing information in the packets to

switch packets from one physical network to another, moving them toward their final destination. Each packet travels the network independently of any other packet.

The datagram is the packet format defined by Internet Protocol. Figure 1.2 is a pictorial representation of an IP datagram. The first five or six 32-bit words of the datagram are control information called the header. By default, the header is five words long; the sixth word is optional. Because the header's length is variable, it includes a field called Internet Header Length (IHL) that indicates the header's length in words. The header contains all the information necessary to deliver the packet.

The Internet Protocol delivers the datagram by checking the Destination Address in word 5 of the header. The Destination Address is a standard 32-bit IP address that identifies the destination network and the specific host on that network.

If the Destination Address is the address of a host on the local network, the packet is delivered directly to the destination. If the Destination Address is not on the local network, the packet is passed to a gateway for delivery. Gateways are devices that switch packets between the different physical networks. Deciding which gateway to use is called routing. IP makes the routing decision for each individual packet.

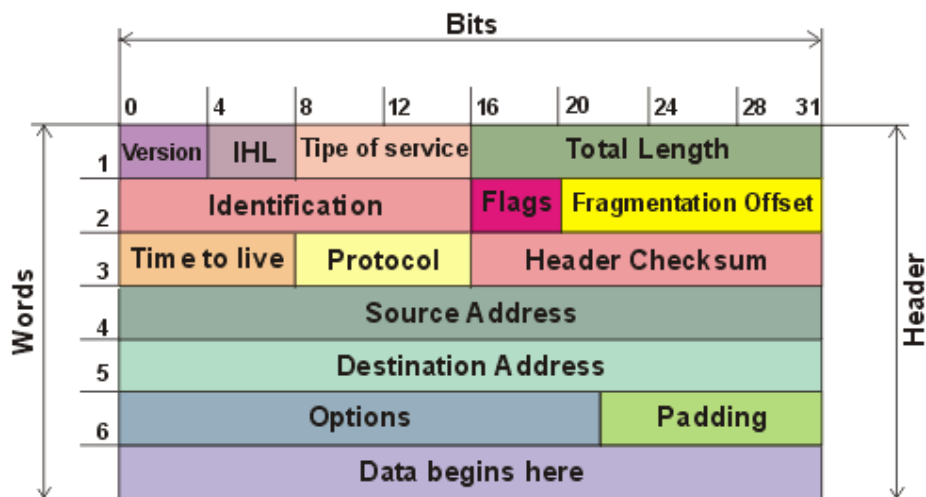


Figure 1.2: IP datagram format

1.1.2 Routing datagrams

In the Internet model, constituent networks are connected together by IP datagram forwarders which are called routers or IP routers. Historically, routers have been realized with packet-switching software executing on a general-purpose CPU. However, as custom hardware development becomes cheaper and as higher throughput is required, special purpose hardware is becoming increasingly common. This specification applies to routers regardless of how they are implemented.

Internet gateways are commonly (and perhaps more accurately) referred to as IP routers because they use Internet Protocol to route packets between networks. In traditional TCP/IP jargon, there are only two types of network devices - gateways and hosts. Gateways forward packets between networks, and hosts don't. However, if a host is connected to more than one network (called a multi-homed host), it can forward packets between the networks. When a multi-homed host forwards packets, it acts just like any other gateway and is considered to be a gateway. Current data communications terminology makes a distinction between gateways and routers, but we'll use the terms gateway and IP router interchangeably. In current terminology, a gateway moves data between different protocols and a router moves data between different networks. So a system that moves mail between TCP/IP and OSI is a gateway, but a traditional IP gateway is a router.

Figure 1.3 shows the use of gateways to forward packets. The hosts (or end systems) process packets through all four protocol layers, while the gateways (or intermediate systems) process the packets only up to the Internet Layer where the routing decisions are made.

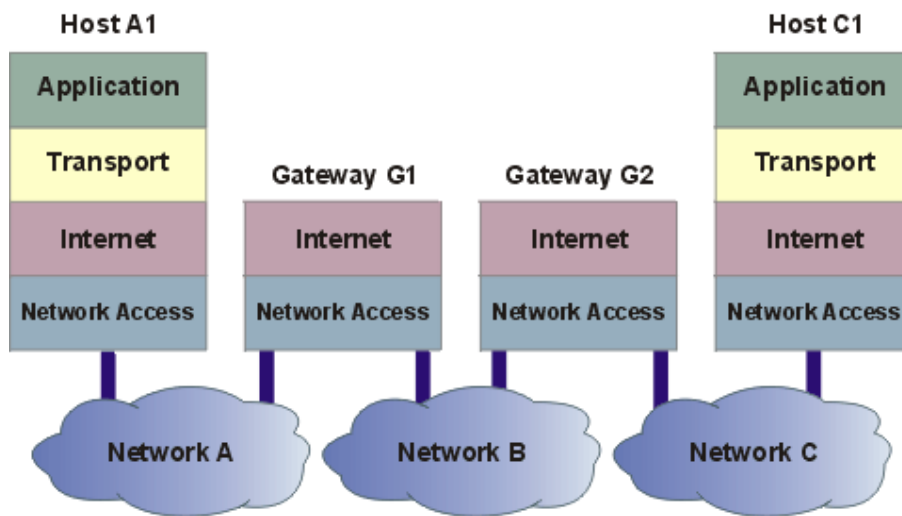


Figure 1.3: Routing through gateways

Systems can only deliver packets to other devices attached to the same physical network. Packets from A1 destined for host C1 are forwarded through gateways G1 and G2. Host A1 first delivers the packet to gateway G1, with which it shares network A. Gateway G1 delivers the packet to G2 over network B. Gateway G2 then delivers the packet directly to host C1, because they are both attached to network C. Host A1 has no knowledge of any gateways beyond gateway G1. It sends packets destined for both networks C and B to that local gateway, and then relies on that gateway to properly forward the packets along the path to their destinations. Likewise, host C1 would send its packets to G2, in order to reach a host on network A, as well as any host on network B.

1.2 Internet Control Message Protocol

ICMP is often considered part of the IP layer. It communicates error messages and other conditions that require attention. ICMP messages are usually acted on by either the IP layer

or the higher layer protocol (TCP or UDP). Some ICMP messages cause errors to be returned to user processes.

ICMP messages are transmitted within IP datagrams, as shown in Figure 1.4.

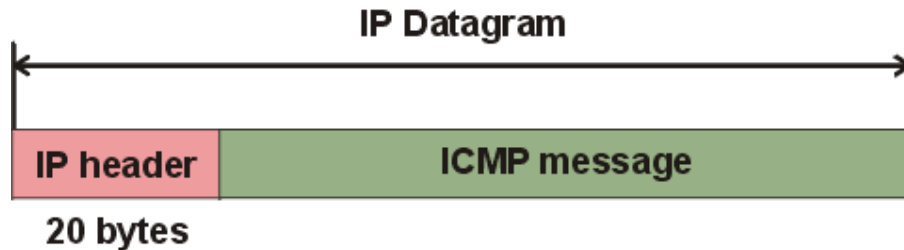


Figure 1.4 ICMP messages encapsulated within an IP datagram

We need to make this distinction because ICMP error messages are sometimes handled specially. For example, an ICMP error message is never generated in response to an ICMP error message. (If this were not the rule, we could end up with scenarios where an error generates an error, which generates an error, and so on, indefinitely).

When an ICMP error message is sent, the message always contains the IP header and the first 8 bytes of the IP datagram that caused the ICMP error to be generated. This lets the receiving ICMP module associate the message with one particular protocol (TCP or UDP from the protocol field in the IP header) and one particular user process (from the TCP or UDP port numbers that are in the TCP or UDP header contained in the first 8 bytes of the IP datagram).

The ICMP timestamp request allows a system to query another for the current time. The recommended value to be returned is the number of milliseconds since midnight, Coordinated Universal Time (UTC). (Older manuals refer to UTC as Greenwich Mean Time.) The nice feature of this ICMP message is that it provides millisecond resolution, whereas some other methods for obtaining the time from another host (such as the `rdate` command provided by some Unix systems) provide a resolution of seconds. The drawback is that only the time since midnight is returned-the caller must know the date from some other means.

1.3 Routers

Routing is the act of moving information across an internetwork from a source to a destination. Along the way, at least one intermediate node typically is encountered. Routing is often contrasted with bridging, which might seem to accomplish precisely the same thing to the casual observer. The primary difference between the two is that bridging occurs at Layer 2 (the data link layer) of the OSI reference model, whereas routing occurs at Layer 3 (the network layer). This distinction provides routing and bridging with different information to use in the process of moving information from source to destination, so the two functions accomplish their tasks in different ways.

A router connects to two or more logical interfaces (Figure 1.5), represented by IP subnets or unnumbered point to point lines. Thus, it has at least one physical interface. Forwarding an

IP datagram generally requires the router to choose the address and relevant interface of the next-hop router or (for the final hop) the destination host. This choice, called relaying or forwarding depends upon a route database within the router. The route database is also called a routing table or forwarding table. The term "router" derives from the process of building this route database; routing protocols and configuration interact in a process called routing.

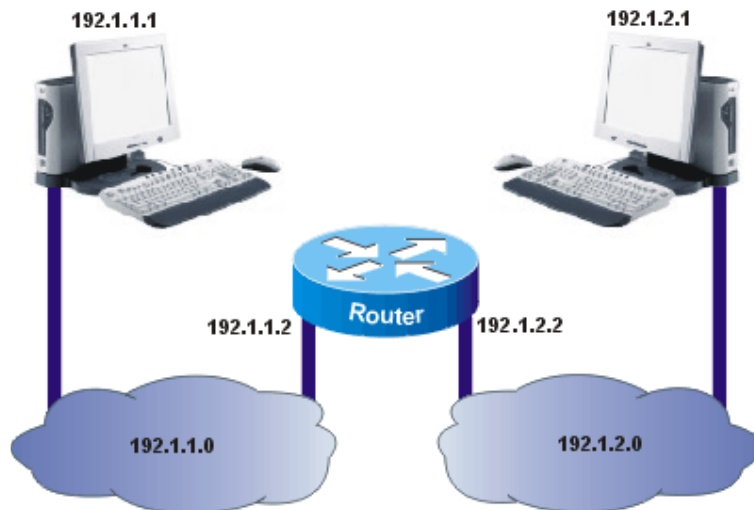


Figure 1.5

The routing database should be maintained dynamically to reflect the current topology of the Internet system. A router normally accomplishes this by participating in distributed routing and reachability algorithms with other routers.

Routers provide datagram transport only, and they seek to minimize the state information necessary to sustain this service in the interest of routing flexibility and robustness.

An Autonomous System (AS) is a connected segment of a network topology that consists of a collection of subnetworks (with hosts attached) interconnected by a set of routes. The subnetworks and the routers are expected to be under the control of a single operations and maintenance (O&M) organization. Within an AS routers may use one or more interior routing protocols, and sometimes several sets of metrics. An AS is expected to present to other ASs an appearance of a coherent interior routing plan, and a consistent picture of the destinations reachable through the AS. An AS is identified by an Autonomous System number.

2. Bridging With Routers

In order to improve the performance the following enhancements have been added to some routers:

- transparent bridging
- source-route bridging

Transparent bridges are found predominantly in Ethernet networks, and source-route bridges (SRBs) are found almost exclusively in Token Ring networks. Both transparent bridges and SRBs are popular.

2.1 Transparent Bridging

Transparent bridges were first developed at Digital Equipment Corporation (Digital) in the early 1980s. Digital submitted its work to the Institute of Electrical and Electronic Engineers (IEEE), which incorporated the work into the IEEE 802.1 standard. Transparent bridges are very popular in Ethernet/IEEE 802.3 networks. This chapter provides an overview of transparent bridging's handling of traffic and protocol components.

Transparent bridges are so named because their presence and operation are transparent to network hosts. When transparent bridges are powered on, they learn the workstation locations by analyzing the source address of incoming frames from all attached networks. For example, if a bridge sees a frame arrive on port 1 from Host A, the bridge concludes that Host A can be reached through the segment connected to port 1. Through this process, transparent bridges build a table (the learning process), such as the one in Figure 2.1.

Host address	Network number
15	1
17	1
12	2
13	2
18	1
9	1
14	3
.	.
.	.
.	.

Figure 2.1. Transparent Bridges Build a Table That Determines a Host's Accessibility

The bridge uses its table as the basis for traffic forwarding. When a frame is received on one of the bridge's interfaces, the bridge looks up the frame's destination address in its internal table. If the table contains an association between the destination address and any of the bridge's ports aside from the one on which the frame was received, the frame is forwarded out the indicated port. If no association is found, the frame is flooded to all ports except the inbound port. Broadcasts and multicasts also are flooded in this way.

Transparent bridges successfully isolate intrasegment traffic, thereby reducing the traffic seen on each individual segment. This is called filtering and occurs when the source and destination MAC addresses reside on the same bridge interface. Filtering usually improves network response times, as seen by the user. The extent to which traffic is reduced and response times are improved depends on the volume of intersegment traffic relative to the total traffic, as well as the volume of broadcast and multicast traffic.

2.2 Source-Route Bridging

The source-route bridging (SRB) algorithm was developed by IBM and was proposed to the IEEE 802.5 committee as the means to bridge between all LANs. Since its initial proposal, IBM has offered a new bridging standard to the IEEE 802 committee: the source-route transparent (SRT) bridging solution. SRT bridging eliminates pure SRBs, proposing that the two types of LAN bridges be transparent bridges and SRT bridges. Although SRT bridging has achieved support, SRBs are still widely deployed.

SRBs are so named because they assume that the complete source-to-destination route is placed in all inter-LAN frames sent by the source. SRBs store and forward the frames as indicated by the route appearing in the appropriate frame field.

SRBs are so named because they assume that the complete source-to-destination route is placed in all inter-LAN frames sent by the source. SRBs store and forward the frames as indicated by the route appearing in the appropriate frame field. Figure 25-1 illustrates a sample SRB network.

In Figure 2.2, assume that Host X wants to send a frame to Host Y. Initially, Host X does not know whether Host Y resides on the same LAN or a different LAN. To determine this, Host X sends out a test frame. If that frame returns to Host X without a positive indication that Host Y has seen it, Host X assumes that Host Y is on a remote segment.

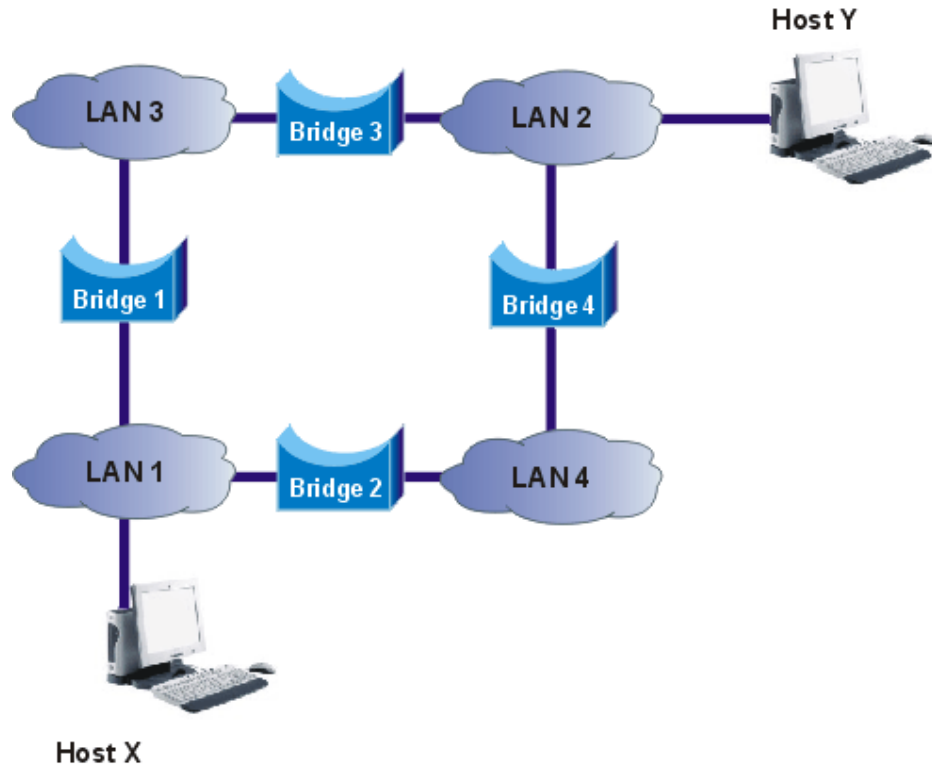


Figure 2.2: An SRB Network Contains LANs and Bridges

To determine the exact remote location of Host Y, Host X sends an explorer frame. Each bridge receiving the explorer frame (Bridges 1 and 2, in this example) copies the frame onto all outbound ports. Route information is added to the explorer frames as they travel through the internetwork. When Host X's explorer frames reach Host Y, Host Y replies to each individually, using the accumulated route information. Upon receipt of all response frames, Host X chooses a path based on some predetermined criteria.

In the example in Figure 25-1, this process will yield two routes:

- LAN 1 to Bridge 1 to LAN 3 to Bridge 3 to LAN 2
- LAN 1 to Bridge 2 to LAN 4 to Bridge 4 to LAN 2

Host X must select one of these two routes. The IEEE 802.5 specification does not mandate the criteria that Host X should use in choosing a route, but it does make several suggestions, including the following:

- First frame received
- Response with the minimum number of hops
- Response with the largest allowed frame size
- Various combinations of the preceding criteria

In most cases, the path contained in the first frame received is used.

After a route is selected, it is inserted into frames destined for Host Y in the form of a routing information field (RIF). A RIF is included only in those frames destined for other LANs. The presence of routing information within the frame is indicated by setting the most significant bit within the Source Address field, called the routing information indicator (RII) bit.

3. Static Routing

For routing between routers to work efficiently in an internetwork, routers must have knowledge of other network IDs or be configured with a default route. On large internetworks, the routing tables must be maintained so that the traffic always travels along optimal paths. How the routing tables are maintained defines the distinction between static and dynamic routing.

A router with manually configured routing tables is known as a *static router*. A network administrator, with knowledge of the internetwork topology, manually builds and updates the routing table, programming all routes in the routing table.

Static routers can work well for small internetworks but do not scale well to large or dynamically changing internetworks due to their manual administration. Static routers are not fault tolerant. The lifetime of a manually configured static route is infinite and, therefore, static routers do not sense and recover from crashed routers or broken links.

3.1 Router Characteristics

An Internet router performs the following functions:

1. Conforms to specific Internet protocols, including the Internet Protocol (IP), Internet Control Message Protocol (ICMP), and others as necessary.
2. Interfaces to two or more packet networks. For each connected network the router must implement the functions required by that network. These functions typically include:
 - Encapsulating and decapsulating the IP datagrams with the connected network framing (e.g., an Ethernet header and checksum),
 - Sending and receiving IP datagrams up to the maximum size supported by that network, this size is the network's Maximum Transmission Unit or MTU,
 - Translating the IP destination address into an appropriate network-level address for the connected network (e.g., an Ethernet hardware address), if needed, and
 - Responding to network flow control and error indications, if any.
3. Receives and forwards Internet datagrams. Important issues in this process are buffer management, congestion control, and fairness.
 - Recognizes error conditions and generates ICMP error and information messages as required.
 - Drops datagrams whose time-to-live fields have reached zero.
 - Fragments datagrams when necessary to fit into the MTU of the next network.
4. Chooses a next-hop destination for each IP datagram, based on the information in its routing database.
5. (Usually) supports an interior gateway protocol (IGP) to carry out distributed routing and reachability algorithms with the other routers in the same autonomous system. In addition, some routers will need to support an exterior gateway protocol (EGP) to exchange topological information with other autonomous systems.
6. Provides network management and system support facilities, including loading, debugging, status reporting, exception reporting and control.

3.2 Routing Table

Routers route data between networks; but all network devices, hosts as well as routers, must make routing decisions. For most hosts, the routing decisions are simple:

- If the destination host is on the local network, the data is delivered to the destination host.
- If the destination host is on a remote network, the data is forwarded to a local router.

Because routing is network-oriented, IP makes routing decisions based on the network portion of the address. The IP module determines the network part of the destination's IP address by applying the network mask to the address. If the destination network is the local network, the mask that is applied may be the local subnet mask. If no mask is provided with the address, the address class determines the network portion of the address.

After determining the destination network, the IP module looks up the network in the local routing table. Packets are routed toward their destination as directed by the routing table. The routing table may be built by the system administrator or by routing protocols, but the end result is the same; IP routing decisions are simple table look-ups.

A routing table consists of the following fields:

Destination

The destination network (or host).

Gateway

The gateway to use to reach the specified destination.

Flags

The flags describe certain characteristics of this route. The possible flag values are:

U

Indicates that the route is up and operational.

H

Indicates this is a route to a specific host (most routes are to networks).

G

Means the route uses a gateway. The system's network interfaces provide routes to directly connected networks. All other routes use remote gateways. Directly connected networks do not have the G flag set; all other routes do.

D

Means that this route was added because of an ICMP Redirect Message. When a system learns of a route via an ICMP Redirect, it adds the route to its routing table, so that additional packets bound for that destination will not need to be redirected. The system uses the D flag to mark these routes.

Ref

The number of times the route has been referenced to establish a connection.

Use

The number of packets transmitted via this route.

Interface

The name of the network interface used by this route.

Routing Table of a host:

Destination	Netmask	Gateway	Flags	Ref	Use	Interface
127.0.0.1	255.255.255.255	127.0.0.1	UH	1	298	eth0
192.1.1.0	255.255.255.0	192.1.1.1	U	6	73723	eth0
192.1.2.0	255.255.255.0	192.1.1.3	UG	4	5325	
default	0.0.0.0	192.1.1.1	UG	2	53627	

The first table entry is the loopback route for the local host. This is the loopback address is a reserved network number. Because every system uses the loopback route to send datagrams to itself, this entry is in every host's routing table. The H flag is set because it is a route to a specific host (127.0.0.1), not a route to an entire network (127.0.0.0).

Another unique entry in the routing table is the entry with the word "default" in the destination field. This entry is for the default route, and the gateway specified in this entry is the default gateway. The default route is an other reserved network number: 0.0.0.0. The default gateway is used whenever there is no specific route in the table for a destination network address. For example, this routing table has no entry for network 192.1.4.0. If IP receives any datagrams addressed to this network, it will send the datagram via the default gateway 192.1.1.1.

You can tell from the sample routing table display that this host is directly connected to network 192.1.1.0. The routing table entry for that network does not specify an external gateway; i.e., the routing table entry for 192.1.1.0 does not have the G flag set. Therefore, this computer must be directly connected to that network.

All of the gateways that appear in a routing table are on networks directly connected to the local system. In the sample shown above this means that, regardless of the destination address, the gateway addresses all begin with 192.1.1. This is the only network to which this computer is directly attached, and therefore it is the only network to which it can directly deliver data. The gateways that the computer uses to reach the rest of the Internet must be on its subnet.

In Figure 3.1 the IP layer of each host and gateway on our imaginary network is replaced by a small piece of a routing table, showing destination networks and the gateways used to reach those destinations. When the source host (192.1.1.1) sends data to the destination host (192.1.2.1), it first determines that 192.1.2.1 is the local network's official address and applies the subnet mask.

$$191.1.2.1 \text{ AND } 255.255.255.0 = 191.1.2.0$$

After applying the subnet mask, IP knows that the destination's network address is 192.1.2.0. The routing table in the source host shows that data bound for 192.1.2.0 should be sent to gateway 192.1.1.2. Gateway 192.1.1.2 makes direct delivery through its 192.1.2.2 interface. Examining the routing tables shows that all systems list only gateways on networks they are directly connected to. Note that 192.1.1.3 is the default gateway for both 192.1.1.1 and 192.1.1.2. But because 192.1.2.1 cannot reach network 192.1.1.0 directly, it has a different default route.

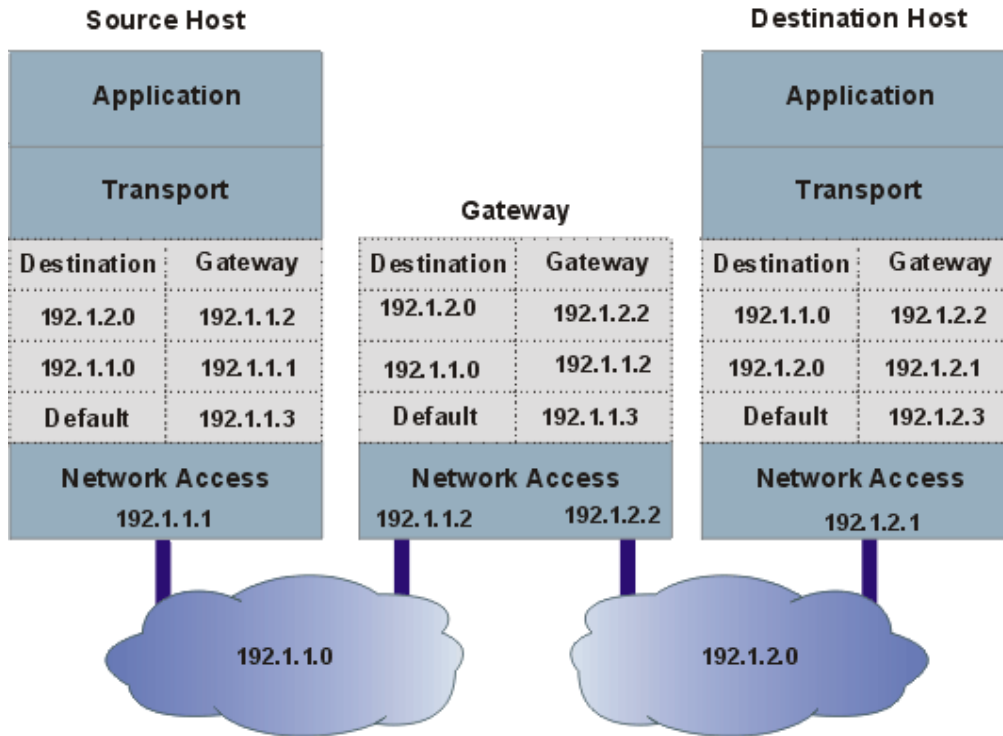


Figure 3.1: Table-based routing

A routing table does not contain end-to-end routes. A route points only to the next gateway, called the next hop, along the path to the destination network. The host relies on the local gateway to deliver the data, and the gateway relies on other gateways. As a datagram moves from one gateway to another, it should eventually reach one that is directly connected to its destination network. It is this last gateway that finally delivers the data to the destination host.

4. Dynamic Routing

Dynamic routing occurs when routers talk to adjacent routers, informing each other of what networks each router is currently connected to. The routers must communicate using a *routing protocol*, of which there are many to choose from.

In a system such as the Internet, many different routing protocols are currently used. The Internet is organized into a collection of *autonomous systems* (ASs), each of which is normally administered by a single entity. A corporation or university campus often defines an autonomous system. For example, the NSFNET backbone of the Internet forms an autonomous system, because all the routers in the backbone are under a single administrative control.

Each autonomous system can select its own routing protocol to communicate between the routers in that autonomous system. This is called an *interior gateway protocol* (IGP) or *intradomain* routing protocol. The most popular IGP has been the *Routing Information Protocol* (RIP). A newer IGP is the *Open Shortest Path First* protocol (OSPF). It is intended as a replacement for RIP. An older IGP that has fallen out of use is HELLO—the IGP used on the original NSFNET backbone in 1986.

Separate routing protocols called *exterior gateway protocols* (EGPs) or *interdomain routing protocols* are used between the routers in different autonomous systems. Historically (and confusingly) the predominant EGP has been a protocol of the same name: EGP. A newer EGP is the Border Gateway Protocol (BGP) that is currently used between the NSFNET backbone and some of the regional networks that attach to the backbone. BGP is intended to replace EGP.

The success of dynamic routing depends on two basic router functions:

- maintenance of a routing table
- timely distribution of knowledge, in the form of routing updates, to other routers

Dynamic routing relies on a routing protocol to share knowledge among routers. A routing protocol defines the set of rules used by a router when it communicates with neighboring routers. For example, a routing protocol describes:

- how to send updates
- what knowledge is contained in these updates
- when to send this knowledge
- how to locate recipients of the updates

When a routing algorithm updates a routing table, its primary objective is to determine the best information to include in the table. Each routing algorithm interprets what is best in its own way. The algorithm generates a number, called the metric value, for each path through the network. Typically, the smaller the metric number, the better the path.

Routing algorithms have used many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. All the following metrics have been used:

- Path Length
- Reliability
- Delay
- Bandwidth
- Load
- Communication Cost

Path length is the most common routing metric. Some routing protocols allow network administrators to assign arbitrary costs to each network link. In this case, path length is the sum of the costs associated with each link traversed. Other routing protocols define *hop count*, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take en route from a source to a destination.

Reliability, in the context of routing algorithms, refers to the dependability (usually described in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. Any reliability factors can be taken into account in the assignment of the reliability ratings, which are arbitrary numeric values usually assigned to network links by network administrators.

Routing delay refers to the length of time required to move a packet from source to destination through the internetwork. Delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be travelled. Because delay is a conglomeration of several important variables, it is a common and useful metric.

Bandwidth refers to the available traffic capacity of a link. All other things being equal, a 10-Mbps Ethernet link would be preferable to a 64-kbps leased line. Although bandwidth is a rating of the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. If, for example, a faster link is busier, the actual time required to send a packet to the destination could be greater.

Load refers to the degree to which a network resource, such as a router, is busy. Load can be calculated in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can be resource-intensive itself.

Communication cost is another important metric, especially because some companies may not care about performance as much as they care about operating expenditures. Even though line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time. Most routing algorithms can be classified as one of two basic algorithms:

- distance vector; or
- link state.

The distance-vector routing approach determines the direction (vector) and distance to any link in the internetwork. The link-state (also called *shortest path first*) approach re-creates the exact topology of the entire internetwork (or at least the portion in which the router is situated).

The balanced hybrid approach combines aspects of the link-state and distance-vector algorithms. The next several pages cover procedures and problems for each of these routing algorithms and present techniques for minimizing the problems.

The routing algorithm is fundamental to dynamic routing. Whenever the topology of a network changes because of growth, reconfiguration, or failure, the network knowledge base must also change. The knowledge needs to reflect an accurate, consistent view of the new topology. This view is called *convergence*.

When all routers in an internetwork are operating with the same knowledge, the internetwork is said to have *converged*. Fast convergence is a desirable network feature because it reduces the period of time in which routers would continue to make incorrect/wasteful routing decisions.

4.1 Distance Vector Routing Protocols

The basic distance vector algorithm tries to solve the problem of how to reach the destination in the shortest number of 'hops' possible. A hop is whenever you pass through a node. So in the simple example network here the distance from router A to network D is 3.

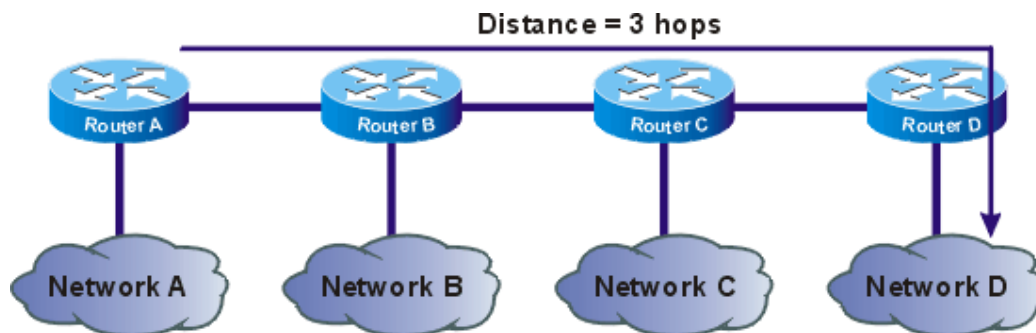


Figure 4.1 Distance from a router to a destination

The problem becomes more interesting if the network is not a long line. For example, the distance between A and D in the below network can be 3, 4, 5, 7 and more

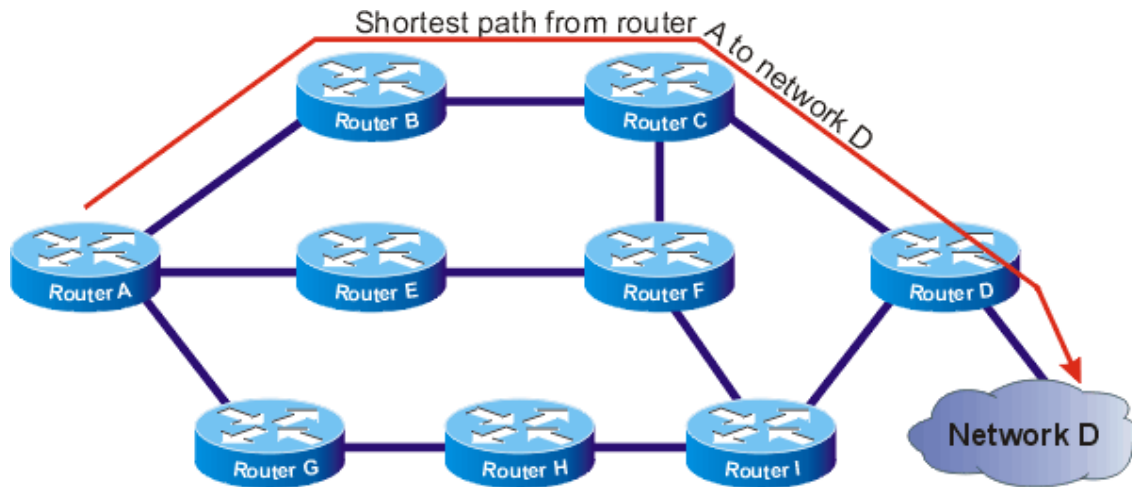


Figure 4.2 Shortest Path

In this example the path that would be desired is from A to B to K then to D. That the Bellman-Ford or Dijkstra algorithms do is find this path. Then each node in the network will know the shortest path from itself to each other node in the network.

A number of different approaches for finding routes between networks are possible. One useful way of categorizing these approaches is on the basis of the type of information the gateways need to exchange in order to be able to find routes. Distance vector algorithms are based on the exchange of only a small amount of information. Each entity (gateway or host) that participates in the routing protocol is assumed to keep information about all of the destinations within the system. Generally, information about all entities connected to one network is summarized by a single entry, which describes the route to all destinations on that network. This summarization is possible because as far as IP is concerned, routing within a network is invisible. Each entry in this routing database includes the next gateway to which datagrams destined for the entity should be sent. In addition, it includes a "metric" measuring the total distance to the entity. Distance is a somewhat generalized concept, which may cover the time delay in getting messages to the entity, the dollar cost of sending messages to it, etc. Distance vector algorithms get their name from the fact that it is possible to compute optimal routes when the only information exchanged is the list of these distances. Furthermore, information is only exchanged among entities that are adjacent, that is, entities that share a common network.

4.1.1 RIP: Routing Information Protocol

RIP is one protocol in a series of routing protocols based on the Bellman-Ford (or distance vector) algorithm. This algorithm has been used for routing computations in computer networks since the early days of the ARPANET.

This protocol is most useful as an "interior gateway protocol". In a nationwide network such as the current Internet, it is very unlikely that a single routing protocol will be used for the whole network. Rather, the network will be organized as a collection of "autonomous systems". An

autonomous system will in general be administered by a single entity, or at least will have some reasonable degree of technical and administrative control. Each autonomous system will have its own routing technology. This may well be different for different autonomous systems. The routing protocol used within an autonomous system is referred to as an interior gateway protocol, or "IGP". A separate protocol is used to interface among the autonomous systems. The earliest such protocol, still used in the Internet, is "EGP" (exterior gateway protocol). Such protocols are now usually referred to as inter-AS routing protocols. RIP was designed to work with moderate-size networks using reasonably homogeneous technology. Thus it is suitable as an IGP for many campuses and for regional networks using serial lines whose speeds do not vary widely. It is not intended for use in more complex environments.

RIP is one of a class of algorithms known as "distance vector algorithms". The earliest description of this class of algorithms known to the author is in Ford and Fulkerson. Because of this, they are sometimes known as Ford-Fulkerson algorithms. The term Bellman-Ford is also used. It comes from the fact that the formulation is based on Bellman's equation, the basis of "dynamic programming".

RIP is intended for use within the IP-based Internet. The Internet is organized into a number of networks connected by gateways. The networks may be either point-to-point links or more complex networks such as Ethernet or the ARPANET. Hosts and gateways are presented with IP datagrams addressed to some host. Routing is the method by which the host or gateway decides where to send the datagram. It may be able to send the datagram directly to the destination, if that destination is on one of the networks that are directly connected to the host or gateway. However, the interesting case is when the destination is not directly reachable. In this case, the host or gateway attempts to send the datagram to a gateway that is nearer the destination. The goal of a routing protocol is very simple: It is to supply the information that is needed to do routing.

4.1.1.1 Limitations of the protocol

This protocol does not solve every possible routing problem. As mentioned above, it is primarily intended for use as an IGP, in reasonably homogeneous networks of moderate size. In addition, the following specific limitations should be mentioned:

- The protocol is limited to networks whose longest path involves 15 hops. The designers believe that the basic protocol design is inappropriate for larger networks. Note that this statement of the limit assumes that a cost of 1 is used for each network. This is the way RIP is normally configured. If the system administrator chooses to use larger costs, the upper bound of 15 can easily become a problem.
- The protocol depends upon "counting to infinity" to resolve certain unusual situations. (This will be explained in the next section.) If the system of networks has several hundred networks, and a routing loop was formed involving all of them, the resolution of the loop would require either much time (if the frequency of routing updates were limited) or bandwidth (if updates were sent whenever changes were detected). Such a loop would consume a large amount of network bandwidth before the loop was corrected. We believe that in realistic cases, this will not be a problem except on slow lines. Even then, the problem will be fairly unusual, since various precautions are taken that should prevent these problems in most cases.
- This protocol uses fixed "metrics" to compare alternative routes. It is not appropriate for situations where routes need to be chosen based on real-time parameters such as a measured delay, reliability, or load. The obvious extensions to allow metrics of this

type are likely to introduce instabilities of a sort that the protocol is not designed to handle.

4.1.1.2 Specifications for the protocol

RIP is intended to allow hosts and gateways to exchange information for computing routes through an IP-based network. RIP is a distance vector protocol. RIP may be implemented by both hosts and gateways. As in most IP documentation, the term "host" will be used here to cover either. RIP is used to convey information about routes to "destinations", which may be individual hosts, networks, or a special destination used to convey a default route.

Any host that uses RIP is assumed to have interfaces to one or more networks. These are referred to as its "directly-connected networks". The protocol relies on access to certain information about each of these networks. The most important is its metric or "cost". The metric of a network is an integer between 1 and 15 inclusive. It is set in some manner not specified in this protocol. Most existing implementations always use a metric of 1. New implementations should allow the system administrator to set the cost of each network. In addition to the cost, each network will have an IP network number and a subnet mask associated with it. These are to be set by the system administrator in a manner not specified in this protocol.

It is assumed that there is a single subnet mask applying to each IP network, and that only the subnet masks for directly-connected networks are known. There may be systems that use different subnet masks for different subnets within a single network. There may also be instances where it is desirable for a system to know the subnets masks of distant networks. However, such situations will require modifications of the rules which govern the spread of subnet information. Such modifications raise issues of interoperability, and thus must be viewed as modifying the protocol.

Each host that implements RIP is assumed to have a routing table. This table has one entry for every destination that is reachable through the system described by RIP. Each entry contains at least the following information:

- The IP address of the destination.
- A metric, which represents the total cost of getting a datagram from the host to that destination. This metric is the sum of the costs associated with the networks that would be traversed in getting to the destination.
- The IP address of the next gateway along the path to the destination. If the destination is on one of the directly-connected networks, this item is not needed.
- A flag to indicate that information about the route has changed recently. This will be referred to as the "route change flag."
- Various timers associated with the route.

The entries for the directly-connected networks are set up by the host, using information gathered by means not specified in this protocol. The metric for a directly-connected network is set to the cost of that network. In existing RIP implementations, 1 is always used for the cost. In that case, the RIP metric reduces to a simple hop-count. More complex metrics may be used when it is desirable to show preference for some networks over others, for example because of differences in bandwidth or reliability.

Implementors may also choose to allow the system administrator to enter additional routes. These would most likely be routes to hosts or networks outside the scope of the routing system.

Entries for destinations other these initial ones are added and updated by the algorithms described in the following sections.

In order for the protocol to provide complete information on routing, every gateway in the system must participate in it. Hosts that are not gateways need not participate, but many implementations make provisions for them to listen to routing information in order to allow them to maintain their routing tables.

4.1.1.3 Message formats

RIP is a UDP-based protocol. Each host that uses RIP has a routing process that sends and receives datagrams on UDP port number 520. All communications directed at another host's RIP processor are sent to port 520. All routing update messages are sent from port 520. Unsolicited routing update messages have both the source and destination port equal to 520. Those sent in response to a request are sent to the port from which the request came. Specific queries and debugging requests may be sent from ports other than 520, but they are directed to port 520 on the target machine.

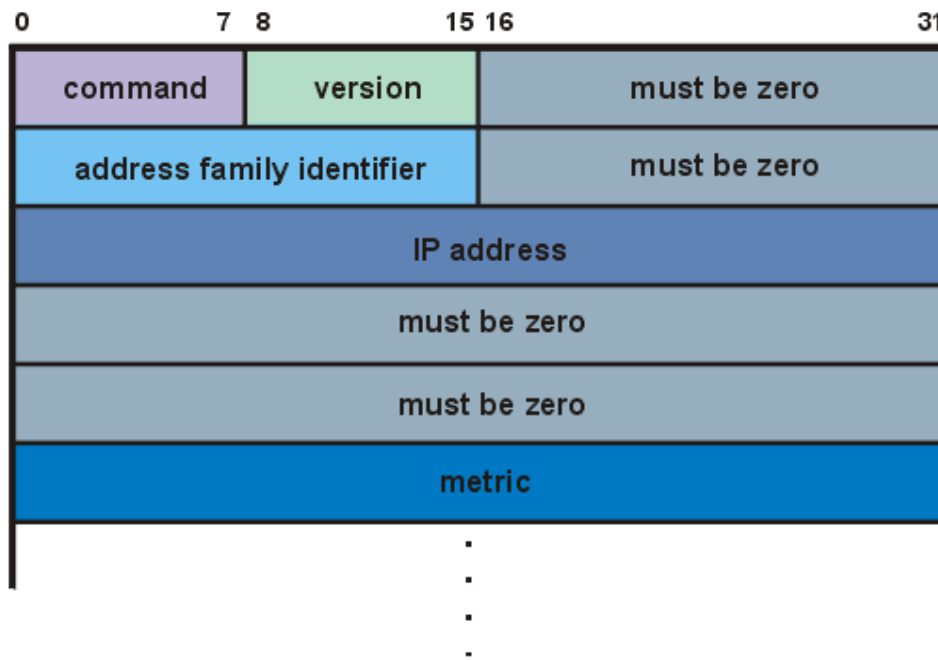


Figure 4.3. Packet format (RIP version 1)

The portion of the datagram from address family identifier through metric may appear up to 25 times. IP address is the usual 4-octet Internet address, in network order.

There are provisions in the protocol to allow "silent" RIP processes. A silent process is one that normally does not send out any messages. However, it listens to messages sent by others. A silent RIP might be used by hosts that do not act as gateways, but wish to listen to routing updates in order to monitor local gateways and to keep their internal routing tables up to date. A gateway that has lost contact with all but one of its networks might choose to become silent, since it is effectively no longer a gateway.

However, this should not be done if there is any chance that neighboring gateways might depend upon its messages to detect that the failed network has come back into operation.

The packet format is shown in Figure 4.3.

Every datagram contains a command, a version number, and possible arguments. This document describes version 1 of the protocol. The command field is used to specify the purpose of this datagram. Here is a summary of the commands implemented in version 1:

<i>1 - request</i>	<i>A request for the responding system to send all or part of its routing table.</i>
<i>2 - response</i>	<i>A message containing all or part of the sender's routing table. This message may be sent in response to a request or poll, or it may be an update message generated by the sender.</i>
<i>3 - traceon</i>	<i>Obsolete. Messages containing this command are to be ignored.</i>
<i>4 - traceoff</i>	<i>Obsolete. Messages containing this command are to be ignored.</i>
<i>5 - reserved</i>	<i>This value is used by Sun Microsystems for its own purposes. If new commands are added in any succeeding version, they should begin with 6. Messages containing this command may safely be ignored by implementations that do not choose to respond to it.</i>

For request and response, the rest of the datagram contains a list of destinations, with information about each. Each entry in this list contains a destination network or host, and the metric for it. The packet format is intended to allow RIP to carry routing information for several different protocols. Thus, each entry has an address family identifier to indicate what type of address is specified in that entry. This document only describes routing for Internet networks. The address family identifier for IP is 2. However, to allow for future development, implementations are required to skip entries that specify address families that are not supported by the implementation. (The size of these entries will be the same as the size of an entry specifying an IP address.) Processing of the message continues normally after any unsupported entries are skipped. The IP address is the usual Internet address, stored as 4 octets in network order. The metric field must contain a value between 1 and 15 inclusive, specifying the current metric for the destination, or the value 16, which indicates that the destination is not reachable. Each route sent by a gateway supercedes any previous route to the same destination from the same gateway.

The maximum datagram size is 512 octets. This includes only the portions of the datagram described above. It does not count the IP or UDP headers. The commands that involve network information allow information to be split across several datagrams. No special provisions are needed for continuations, since correct results will occur if the datagrams are processed individually.

RIP 2 Packet Format

The RIP 2 specification (described in RFC 1723) allows more information to be included in RIP packets and provides a simple authentication mechanism that is not supported by RIP. Figure 4.4 shows the IP RIP 2 packet format.

1-octet command field	1-octet version number field	2-octet unused field	2-octet AFI field	2-octet route tag field	4-octet network address field	4-octet subnet mask field	4-octet next hop field	4-octet metric field
-----------------------------	---------------------------------------	----------------------------	-------------------------	----------------------------------	--	------------------------------------	---------------------------------	----------------------------

Figure 4.4 RIP 2 Packet Format

The following descriptions summarize the IP RIP 2 packet format fields illustrated in Figure 4.4:

- **Command**—Indicates whether the packet is a request or a response. The request asks that a router send all or a part of its routing table. The response can be an unsolicited regular routing update or a reply to a request. Responses contain routing table entries. Multiple RIP packets are used to convey information from large routing tables.
- **Version**—Specifies the RIP version used. In a RIP packet implementing any of the RIP 2 fields or using authentication, this value is set to 2.
- **Unused**—Has a value set to zero.
- **Address-family identifier (AFI)**—Specifies the address family used. IPv2's AFI field functions identically to RFC 1058 RIP's AFI field, with one exception: If the AFI for the first entry in the message is 0xFFFF, the remainder of the entry contains authentication information. Currently, the only authentication type is simple password.
- **Route tag**—Provides a method for distinguishing between internal routes (learned by RIP) and external routes (learned from other protocols).
- **IP address**—Specifies the IP address for the entry.
- **Subnet mask**—Contains the subnet mask for the entry. If this field is zero, no subnet mask has been specified for the entry.
- **Next hop**—Indicates the IP address of the next hop to which packets for the entry should be forwarded.
- **Metric**—Indicates how many internetwork hops (routers) have been traversed in the trip to the destination. This value is between 1 and 15 for a valid route, or 16 for an unreachable route.

4.1.1.4 Addressing considerations

As indicated earlier, distance vector routing can be used to describe routes to individual hosts or to networks. The RIP protocol allows either of these possibilities. The destinations appearing in request and response messages can be networks, hosts, or a special code used to indicate a default address. In general, the kinds of routes actually used will depend upon the routing strategy used for the particular network. Many networks are set up so that routing information for individual hosts is not needed. If every host on a given network or subnet is accessible through the same gateways, then there is no reason to mention individual hosts in the routing tables. However, networks that include point to point lines sometimes require gateways to keep track of routes to certain hosts. Whether this feature is required depends upon the addressing and routing approach used in the system. Thus, some implementations may choose not to support host routes. If host routes are not supported, they are to be dropped when they are received in response messages.

The RIP packet formats do not distinguish among various types of address. Fields that are labeled "address" can contain any of the following:

- host address
- subnet number
- network number
- 0, indicating a default route

Entities that use RIP are assumed to use the most specific information available when routing a datagram. That is, when routing a datagram, its destination address must first be checked against the list of host addresses. Then it must be checked to see whether it matches any known subnet or network number. Finally, if none of these match, the default route is used.

When a host evaluates information that it receives via RIP, its interpretation of an address depends upon whether it knows the subnet mask that applies to the net. If so, then it is possible to determine the meaning of the address. For example, consider net 128.6. It has a subnet mask of 255.255.255.0. Thus 128.6.0.0 is a network number, 128.6.4.0 is a subnet number, and 128.6.4.1 is a host address. However, if the host does not know the subnet mask, evaluation of an address may be ambiguous. If there is a non-zero host part, there is no clear way to determine whether the address represents a subnet number or a host address. As a subnet number would be useless without the subnet mask, addresses are assumed to represent hosts in this situation. In order to avoid this sort of ambiguity, hosts must not send subnet routes to hosts that cannot be expected to know the appropriate subnet mask. Normally hosts only know the subnet masks for directly-connected networks. Therefore, unless special provisions have been made, routes to a subnet must not be sent outside the network of which the subnet is a part.

This filtering is carried out by the gateways at the "border" of the subnetted network. These are gateways that connect that network with some other network. Within the subnetted network, each subnet is treated as an individual network. Routing entries for each subnet are circulated by RIP. However, border gateways send only a single entry for the network as a whole to hosts in other networks. This means that a border gateway will send different information to different neighbors. For neighbors connected to the subnetted network, it generates a list of all subnets to which it is directly connected, using the subnet number. For neighbors connected to other networks, it makes a single entry for the network as a whole, showing the metric associated with that network. (This metric would normally be the smallest metric for the subnets to which the gateway is attached.)

Similarly, border gateways must not mention host routes for hosts within one of the directly-connected networks in messages to other networks. Those routes will be subsumed by the single entry for the network as a whole. We do not specify what to do with host routes for "distant" hosts (i.e., hosts not part of one of the directly-connected networks). Generally, these routes indicate some host that is reachable via a route that does not support other hosts on the network of which the host is a part.

The special address 0.0.0.0 is used to describe a default route. A default route is used when it is not convenient to list every possible network in the RIP updates, and when one or more closely-connected gateways in the system are prepared to handle traffic to the networks that are not listed explicitly. These gateways should create RIP entries for the address 0.0.0.0, just as if it were a network to which they are connected. The decision as to how gateways create

entries for 0.0.0.0 is left to the implementor. Most commonly, the system administrator will be provided with a way to specify which gateways should create entries for 0.0.0.0. However, other mechanisms are possible. For example, an implementor might decide that any gateway that speaks EGP should be declared to be a default gateway. It may be useful to allow the network administrator to choose the metric to be used in these entries. If there is more than one default gateway, this will make it possible to express a preference for one over the other. The entries for 0.0.0.0 are handled by RIP in exactly the same manner as if there were an actual network with this address. However, the entry is used to route any datagram whose destination address does not match any other network in the table. Implementations are not required to support this convention. However, it is strongly recommended. Implementations that do not support 0.0.0.0 must ignore entries with this address. In such cases, they must not pass the entry on in their own RIP updates. System administrators should take care to make sure that routes to 0.0.0.0 do not propagate further than is intended. Generally, each autonomous system has its own preferred default gateway. Thus, routes involving 0.0.0.0 should generally not leave the boundary of an autonomous system. The mechanisms for enforcing this are not specified in this document.

4.1.1.5 Timers

This section describes all events that are triggered by timers. Every 30 seconds, the output process is instructed to generate a complete response to every neighboring gateway. When there are many gateways on a single network, there is a tendency for them to synchronize with each other such that they all issue updates at the same time. This can happen whenever the 30 second timer is affected by the processing load on the system. It is undesirable for the update messages to become synchronized, since it can lead to unnecessary collisions on broadcast networks. Thus, implementations are required to take one of two precautions.

- The 30-second updates are triggered by a clock whose rate is not affected by system load or the time required to service the previous update timer.
- The 30-second timer is offset by addition of a small random time each time it is set.

There are two timers associated with each route, a "timeout" and a "garbage-collection time". Upon expiration of the timeout, the route is no longer valid. However, it is retained in the table for a short time, so that neighbors can be notified that the route has been dropped. Upon expiration of the garbage-collection timer, the route is finally removed from the tables.

The timeout is initialized when a route is established, and any time an update message is received for the route. If 180 seconds elapse from the last time the timeout was initialized, the route is considered to have expired, and the deletion process which we are about to describe is started for it.

Deletions can occur for one of two reasons: (1) the timeout expires, or (2) the metric is set to 16 because of an update received from the current gateway. In either case, the following events happen:

- The garbage-collection timer is set for 120 seconds.

- The metric for the route is set to 16 (infinity). This causes the route to be removed from service.
- A flag is set noting that this entry has been changed, and the output process is signalled to trigger a response.

Until the garbage-collection timer expires, the route is included in all updates sent by this host, with a metric of 16 (infinity). When the garbage-collection timer expires, the route is deleted from the tables.

Should a new route to this network be established while the garbage-collection timer is running, the new route will replace the one that is about to be deleted. In this case the garbage-collection timer must be cleared.

4.1.1.6 Dealing with changes in topology

In practice, gateways and lines often fail and come back up. The theoretical version of the algorithm involved a minimum over all immediate neighbors. If the topology changes, the set of neighbors changes. Therefore, the next time the calculation is done, the change will be reflected. However, as mentioned above, actual implementations use an incremental version of the minimization. Only the best route to any given destination is remembered. If the gateway involved in that route should crash, or the network connection to it break, the calculation might never reflect the change. The algorithm depends upon a gateway notifying its neighbors if its metrics change. If the gateway crashes, then it has no way of notifying neighbors of a change.

In order to handle problems of this kind, distance vector protocols must make some provision for timing out routes. The details depend upon the specific protocol. As an example, in RIP every gateway that participates in routing sends an update message to all its neighbors once every 30 seconds. Suppose the current route for network N uses gateway G. If we don't hear from G for 180 seconds, we can assume that either the gateway has crashed or the network connecting us to it has become unusable. Thus, we mark the route as invalid. When we hear from another neighbor that has a valid route to N, the valid route will replace the invalid one. Note that we wait for 180 seconds before timing out a route even though we expect to hear from each neighbor every 30 seconds. Unfortunately, messages are occasionally lost by networks. Thus, it is probably not a good idea to invalidate a route based on a single missed message.

As we will see below, it is useful to have a way to notify neighbors that there currently isn't a valid route to some network. RIP, along with several other protocols of this class, does this through a normal update message, by marking that network as unreachable. A specific metric value is chosen to indicate an unreachable destination; that metric value is larger than the largest valid metric that we expect to see. In the existing implementation of RIP, 16 is used. This value is normally referred to as "infinity", since it is larger than the largest valid metric. 16 may look like a surprisingly small number. It is chosen to be this small for reasons that we will see shortly. In most implementations, the same convention is used internally to flag a route as invalid.

4.1.1.7 Counting to infinity

The algorithm as presented up to this point will always allow a host or gateway to calculate a correct routing table. However, that is still not quite enough to make it useful in practice. The proofs referred to above only show that the routing tables will converge to the correct values in finite time. They do not guarantee that this time will be small enough to be useful, nor do they say what will happen to the metrics for networks that become inaccessible.

It is easy enough to extend the mathematics to handle routes becoming inaccessible. The convention suggested above will do that. We choose a large metric value to represent "infinity". This value must be large enough that no real metric would ever get that large. For the purposes of this example, we will use the value 16. Suppose a network becomes inaccessible. All of the immediately neighboring gateways time out and set the metric for that network to 16. For purposes of analysis, we can assume that all the neighboring gateways have gotten a new piece of hardware that connects them directly to the vanished network, with a cost of 16. Since that is the only connection to the vanished network, all the other gateways in the system will converge to new routes that go through one of those gateways. It is easy to see that once convergence has happened, all the gateways will have metrics of at least 16 for the vanished network. Gateways one hop away from the original neighbors would end up with metrics of at least 17; gateways two hops away would end up with at least 18, etc. As these metrics are larger than the maximum metric value, they are all set to 16. It is obvious that the system will now converge to a metric of 16 for the vanished network at all gateways.

Unfortunately, the question of how long convergence will take is not amenable to quite so simple an answer. Before going any further, it will be useful to look at an example (Figure 4.5). Note, by the way, that what we are about to show will not happen with a correct implementation of RIP. We are trying to show why certain features are needed.

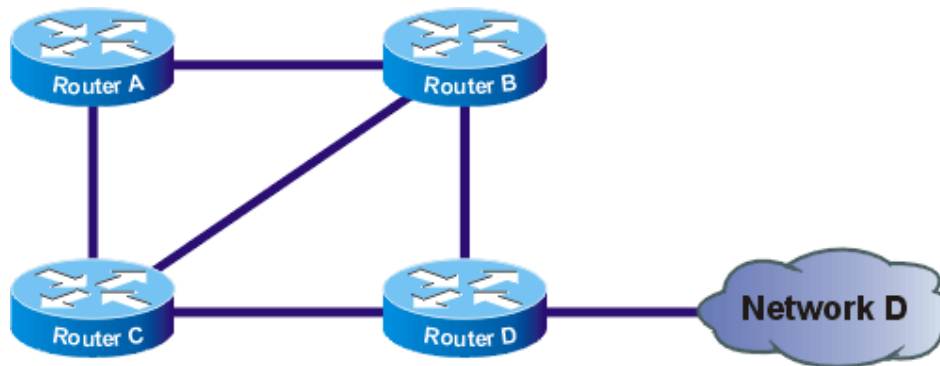


Figure 4.5

All networks have cost 1, except for the direct link from C to D, which has cost 10.

Each router (gateway) will have a table showing a route to each network.

However, for purposes of this illustration, we show only the routes from each gateway to the network D.

- D: directly connected, metric 1
- B: route via D, metric 2
- C: route via B, metric 3
- A: route via B, metric 3

Now suppose that the link from B to D fails. The routes should now adjust to use the link from C to D. Unfortunately, it will take a while for this to happen. The routing changes start when B notices that the route to D is no longer usable. For simplicity, the chart below assumes that all gateways send updates at the same time. The chart shows the metric for the target network, as it appears in the routing table at each gateway.

time ----->

	Next hop	Dist.	Next hop	Dist.	Next hop	Dist.	Next hop	Dist.	Next hop	Dist.	Next hop	Dist.
D	Dir	1	Dir	1	Dir	1	Dir	1		Dir	1	Dir	1
B	Unr	-	C	4	C	5	C	6		C	11	C	12
C	B	3	A	4	A	5	A	6		A	11	D	11
A	B	3	C	4	C	5	C	6		C	11	C	12

Dir = directly connected

Unr = unreachable

Here's the problem: B is able to get rid of its failed route using a timeout mechanism. But vestiges of that route persist in the system for a long time. Initially, A and C still think they can get to D via B. So, they keep sending updates listing metrics of 3. In the next iteration, B will then claim that it can get to D via either A or C. Of course, it can't. The routes being claimed by A and C are now gone, but they have no way of knowing that yet. And even when they discover that their routes via B have gone away, they each think there is a route available via the other. Eventually the system converges, as all the mathematics claims it must. But it can take some time to do so. The worst case is when a network becomes completely inaccessible from some part of the system. In that case, the metrics may increase slowly in a pattern like the one above until they finally reach infinity. For this reason, the problem is called "counting to infinity".

You should now see why "infinity" is chosen to be as small as possible. If a network becomes completely inaccessible, we want counting to infinity to be stopped as soon as possible. Infinity must be large enough that no real route is that big. But it shouldn't be any bigger than required. Thus the choice of infinity is a tradeoff between network size and speed of convergence in case counting to infinity happens. The designers of RIP believed that the protocol was unlikely to be practical for networks with a diameter larger than 15.

There are several things that can be done to prevent problems like this. The ones used by RIP are called "split horizon with poisoned reverse", and "triggered updates".

4.1.1.8 Split horizon

Note that some of the problem above is caused by the fact that A and C are engaged in a pattern of mutual deception. Each claims to be able to get to D via the other. This can be prevented by being a bit more careful about where information is sent. In particular, it is never useful to claim reachability for a destination network to the neighbor(s) from which the route was learned. "Split horizon" is a scheme for avoiding problems caused by including routes in updates sent to the gateway from which they were learned. The "simple split

horizon" scheme omits routes learned from one neighbor in updates sent to that neighbor. "Split horizon with poisoned reverse" includes such routes in updates, but sets their metrics to infinity.

If A thinks it can get to D via C, its messages to C should indicate that D is unreachable. If the route through C is real, then C either has a direct connection to D, or a connection through some other gateway. C's route can't possibly go back to A, since that forms a loop. By telling C that D is unreachable, A simply guards against the possibility that C might get confused and believe that there is a route through A. This is obvious for a point to point line. But consider the possibility that A and C are connected by a broadcast network such as an Ethernet, and there are other gateways on that network. If A has a route through C, it should indicate that D is unreachable when talking to any other gateway on that network. The other gateways on the network can get to C themselves. They would never need to get to C via A. If A's best route is really through C, no other gateway on that network needs to know that A can reach D. This is fortunate, because it means that the same update message that is used for C can be used for all other gateways on the same network. Thus, update messages can be sent by broadcast.

In general, split horizon with poisoned reverse is safer than simple split horizon. If two gateways have routes pointing at each other, advertising reverse routes with a metric of 16 will break the loop immediately. If the reverse routes are simply not advertised, the erroneous routes will have to be eliminated by waiting for a timeout. However, poisoned reverse does have a disadvantage: it increases the size of the routing messages. Consider the case of a campus backbone connecting a number of different buildings. In each building, there is a gateway connecting the backbone to a local network. Consider what routing updates those gateways should broadcast on the backbone network. All that the rest of the network really needs to know about each gateway is what local networks it is connected to. Using simple split horizon, only those routes would appear in update messages sent by the gateway to the backbone network. If split horizon with poisoned reverse is used, the gateway must mention all routes that it learns from the backbone, with metrics of 16. If the system is large, this can result in a large update message, almost all of whose entries indicate unreachable networks.

In a static sense, advertising reverse routes with a metric of 16 provides no additional information. If there are many gateways on one broadcast network, these extra entries can use significant bandwidth. The reason they are there is to improve dynamic behavior. When topology changes, mentioning routes that should not go through the gateway as well as those that should can speed up convergence. However, in some situations, network managers may prefer to accept somewhat slower convergence in order to minimize routing overhead. Thus implementors may at their option implement simple split horizon rather than split horizon with poisoned reverse, or they may provide a configuration option that allows the network manager to choose which behavior to use. It is also permissible to implement hybrid schemes that advertise some reverse routes with a metric of 16 and omit others. An example of such a scheme would be to use a metric of 16 for reverse routes for a certain period of time after routing changes involving them, and thereafter omitting them from updates.

4.1.1.9. Triggered updates

Split horizon with poisoned reverse will prevent any routing loops that involve only two gateways. However, it is still possible to end up with patterns in which three gateways are

engaged in mutual deception. For example, A may believe it has a route through B, B through C, and C through A. Split horizon cannot stop such a loop. This loop will only be resolved when the metric reaches infinity and the network involved is then declared unreachable. Triggered updates are an attempt to speed up this convergence. To get triggered updates, we simply add a rule that whenever a gateway changes the metric for a route, it is required to send update messages almost immediately, even if it is not yet time for one of the regular update message. (The timing details will differ from protocol to protocol. Some distance vector protocols, including RIP, specify a small time delay, in order to avoid having triggered updates generate excessive network traffic.) Note how this combines with the rules for computing new metrics. Suppose a gateway's route to destination N goes through gateway G. If an update arrives from G itself, the receiving gateway is required to believe the new information, whether the new metric is higher or lower than the old one. If the result is a change in metric, then the receiving gateway will send triggered updates to all the hosts and gateways directly connected to it. They in turn may each send updates to their neighbors. The result is a cascade of triggered updates. It is easy to show which gateways and hosts are involved in the cascade. Suppose a gateway G times out a route to destination N. G will send triggered updates to all of its neighbors. However, the only neighbors who will believe the new information are those whose routes for N go through G. The other gateways and hosts will see this as information about a new route that is worse than the one they are already using, and ignore it. The neighbors whose routes go through G will update their metrics and send triggered updates to all of their neighbors. Again, only those neighbors whose routes go through them will pay attention. Thus, the triggered updates will propagate backwards along all paths leading to gateway G, updating the metrics to infinity. This propagation will stop as soon as it reaches a portion of the network whose route to destination N takes some other path.

If the system could be made to sit still while the cascade of triggered updates happens, it would be possible to prove that counting to infinity will never happen. Bad routes would always be removed immediately, and so no routing loops could form.

Unfortunately, things are not so nice. While the triggered updates are being sent, regular updates may be happening at the same time. Gateways that haven't received the triggered update yet will still be sending out information based on the route that no longer exists. It is possible that after the triggered update has gone through a gateway, it might receive a normal update from one of these gateways that hasn't yet gotten the word. This could reestablish an orphaned remnant of the faulty route. If triggered updates happen quickly enough, this is very unlikely. However, counting to infinity is still possible.

4.1.2 IGRP Interior Gateway Routing Protocol

IGRP is a protocol that allows a number of gateways to coordinate their routing. Its goals are

- stable routing even in very large or complex networks. No routing loops should occur, even as transients.
- fast response to changes in network topology
- low overhead. That is, IGRP itself should not use more bandwidth than what is actually needed for its task.
- splitting traffic among several parallel routes when they are of roughly equal desirability.
- taking into account error rates and level of traffic on different paths
- the ability to handle multiple "types of service" with a single set of information.

The current implementation of IGRP handles routing for TCP/IP. However, the basic design is intended to be able to handle a variety of protocols.

During the last few years, routing has suddenly become a more difficult problem than it used to be. A few years ago, protocols such as RIP were sufficient to handle most real networks. However, growth in the Internet, and decentralization of control of its structure, have now resulted in a system of networks that is nearly beyond our capabilities to manage. Similar situations are occurring in large corporate networks as well. IGRP is one tool intended to help attack this problem.

No one tool is going to solve all routing problems. Conventionally the routing problem is broken into several pieces. Protocols such as IGRP are called "internal gateway protocols" (IGPs). They are intended for use within a single set of networks, either under a single management or closely coordinated managements. Such sets of networks are connected by "external gateway protocols" (EGPs). An IGP is designed to keep track of a good deal of detail about network topology. Priority in designing an IGP is placed on producing optimal routes and responding quickly to changes. An EGP is intended to protect one system of networks against errors or intentional misrepresentation by other systems. Priority in designing an EGP is on stability and administrative controls. Often it is sufficient for an EGP to produce a reasonable route, rather than the optimal route. In fact, there are features in Cisco's implementation that allow IGRP to be used as an EGP in some circumstances. However, the emphasis in its design is on use as an IGP.

IGRP has some similarities to older protocols such as Xerox's Routing Information Protocol, Berkeley's RIP, and Dave Mills' Hello. It differs from these protocols primarily in being designed for larger and more complex networks. Section 4 gives a more detailed comparison with RIP, which is the most widely used of the older generation of protocols.

Like these older protocols, IGRP is a distance vector protocol. In such a protocol, gateways exchange routing information only with adjacent gateways. This routing information contains a summary of information about the rest of the network. It can be shown mathematically that all of the gateways taken together are solving an optimization problem by what amounts to a distributed algorithm. Each gateway only needs to solve part of the problem, and it only has to receive a portion of the total data.

4.1.2.1 The Routing Problem

IGRP is intended for use in gateways connecting several networks. We assume that the networks use packet-based technology. In effect the gateways act as packet switches. When a system connected to one network wants to send a packet to a system on a different network, it addresses the packet to a gateway. If the destination is on one of the networks connected to the gateway, the gateway will forward the packet to the destination. If the destination is more distant, the gateway will forward the packet to another gateway that is closer to the destination. Gateways use routing tables to help them decide what to do with packets. Here is a simple example routing table. (Note that the basic routing problem is similar for other protocols as well, but this description will assume that IGRP is being used for routing IP.)

Network	Gateway	Interface
128.6.4	none	Ethernet 0
128.6.5	none	Ethernet 1
128.6.21	128.6.4.1	Ethernet 0
128.121	128.6.5.4	Ethernet 1
10	128.6.5.4	Ethernet 1

(Actual IGRP routing tables have additional information for each gateway, as we will see.) This gateway is connected to two Ethernets, called 0 and 1. They have been given IP network numbers (actually subnet numbers) 128.6.4 and 128.6.5. Thus packets addressed for these specific networks can be sent directly to the destination, simply by using the appropriate Ethernet interface. There are two nearby gateways, 128.6.4.1 and 128.6.5.4. Packets for networks other than 128.6.4 and 128.6.5 will be forwarded to one or the other of those gateways. The routing table indicates which gateway should be used for which network. For example, packets addressed to a host on network 10 should be forwarded to gateway 128.6.5.4. One hopes that this gateway is closer to network 10, i.e. that the best path to network 10 goes through this gateway. The primary purpose of IGRP is allow the gateways to build and maintain routing tables like this.

4.1.2.2 Summary of IGRP

As mentioned above, IGRP is a protocol that allows gateways to build up their routing table by exchanging information with other gateways. A gateway starts out with entries for all of the networks that are directly connected to it. It gets information about other networks by exchanging routing updates with adjacent gateways. In the simplest case, the gateway will find one path that represents the best way to get to each network. A path is characterized by the next gateway to which packets should be sent, the network interface that should be used, and metric information. Metric information is a set of numbers that characterize how good the path is. This allows the gateway to compare paths that it has heard from various gateways and decide which one to use. There are often cases where it makes sense to split traffic between two or more paths. IGRP will do this whenever two or more paths are equally good. The user can also configure it to split traffic when paths are almost equally good. In this case more traffic will be sent along the path with the better metric. The intent is that traffic can be split between a 9600 bps line and a 19200 BPS line, and the 19200 line will get roughly twice as much traffic as the 9600 BPS line.

The metric used by IGRP includes:

- the topological delay time
- the bandwidth of the narrowest bandwidth segment of the path
- the channel occupancy of the path
- the reliability of the path.

Topological delay time is the amount of time it would take to get to the destination along that path, assuming an unloaded network. Of course there is additional delay when the network is loaded. However, load is accounted for by using the channel occupancy figure, not by attempting to measure actual delays. The path bandwidth is simply the bandwidth in bits per second of the slowest link in the path. Channel occupancy indicates how much of that bandwidth is currently in use. It is measured, and will change with load. Reliability indicates the current error rate. It is the fraction of packets that arrive at the destination undamaged. It is measured.

Although they are not used as part of the metric, two additional pieces of information are passed with it: hop count and MTU. The hop count is simply the number of gateways that a packet will have to go through to get to the destination. MTU is the maximum packet size that can be sent along the entire path without fragmentation. (That is, it is the minimum of the MTUs of all the networks involved in the path.)

Based on the metric information, a single "composite metric" is calculated for the path. The composite metric combines the effect of the various metric components into a single number representing the "goodness" of that path. It is the composite metric that is actually used to decide on the best path.

Periodically each gateway broadcasts its entire routing table (with some censoring because of the split horizon rule) to all adjacent gateways. When a gateway gets this broadcast from another gateway, it compares the table with its existing table. Any new destinations and paths are added to the gateway's routing table. Paths in the broadcast are compared with existing paths. If a new path is better, it may replace the existing one. Information in the broadcast is also used to update channel occupancy and other information about existing paths. This general procedure is similar to that used by all distance vector protocols. It is referred to in the mathematical literature as the Bellman-Ford algorithm. For a detailed development of the basic procedure, see RFC 1058, which describes RIP, an older distance vector protocol.

In IGRP, the general Bellman-Ford algorithm is modified in three critical aspects. First, instead of a simple metric, a vector of metrics is used to characterize paths. Second, instead of picking a single path with the smallest metric, traffic is split among several paths, whose metrics fall into a specified range. Third, several features are introduced to provide stability in situations where the topology is changing.

The best path is selected based on a composite metric:

$$[(K1 / Be) + (K2 * Dc)]^r$$

Where:

K1, K2 = constants
Be = unloaded path bandwidth x (1 - channel occupancy)
Dc = topological delay
r = reliability.

The path having the smallest composite metric will be the best path. Where there are multiple paths to the same destination, the gateway can route the packets over more than one path. This is done in accordance with the composite metric for each data path. For instance, if one path has a composite metric of 1 and another path has a composite metric of 3, three times as many packets will be sent over the data path having the composite metric of 1. However, only paths whose composite metrics are within a certain range of the smallest composite metric will be used. K1 and K2 indicate the weight to be assigned to bandwidth and delay. These will depend upon the "type of service". For example, interactive traffic would normally place a higher weight on delay, and file transfer on bandwidth.

There are two advantages to using a vector of metric information. The first is that it provides the ability to support multiple types of service from the same set of data. The second advantage is improved accuracy. When a single metric is used, it is normally treated as if it were a delay. Each link in the path is added to the total metric. If there is a link with a low bandwidth, it is normally represented by a large delay. However, bandwidth limitations don't really cumulate the way delays do. By treating bandwidth as a separate component, it can be handled correctly. Similarly, load can be handled by a separate channel occupancy number.

IGRP provides a system for interconnecting computer networks which can stably handle a general graph topology including loops. The system maintains full path metric information, i.e., it knows the path parameters to all other networks to which any gateway is connected. Traffic can be distributed over parallel paths and multiple path parameters can be simultaneously computed over the entire network.

4.1.2.3 Detailed Description

When a gateway is first turned on, its routing table is initialized. This may be done by an operator from a console terminal, or by reading information from configuration files. A description of each network connected to the gateway is provided, including the topological delay along the link (i.e., how long it takes a single bit to transverse the link) and the bandwidth of the link.

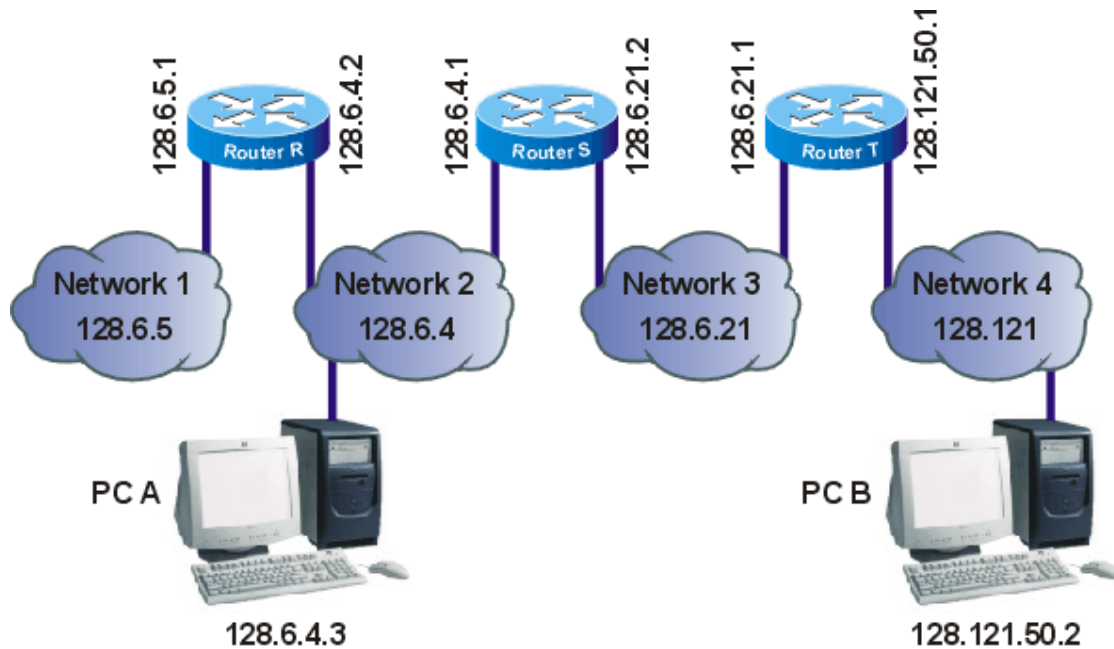


Figure 4.6 A simple example network

For instance, in Fig. 4.6, gateway S would be told that it is connected to networks 2 and 3 via the corresponding interfaces. Thus, initially, gateway 2 only knows that it can reach any destination computer in networks 2 and 3. All the gateways are programmed to periodically transmit to their neighboring gateways the information that they have been initialized with, as well as information gathered from other gateways. Thus, gateway S would receive updates from gateways R and T and learn that it can reach computers in network 1 through gateway R and computers in network 4 through gateway T. Since gateway S sends its entire routing table, in the next cycle gateway T will learn that it can get to network 1 through gateway S. It is easy to see that information about every network in the system will eventually reach every gateway in the system, providing only that the network is fully connected.

Each gateway computes a composite metric to determine the desirability of the data paths to destination computers. For instance, in Fig 4.7, for a destination in network 6, gateway A would compute metric functions for two paths, via gateways B and C. Note that paths are defined simply by the next hop. There are actually three possible routes from A to network 6:

- Direct to B
- To C and then to B
- To C and then to D

However, gateway A need not choose between the two routes involving C. The routing table in A has a single entry representing the path to C. Its metric represents the best way of getting from C to the final destination. If A sends a packet to C, it is up to C to decide whether to use B or D.

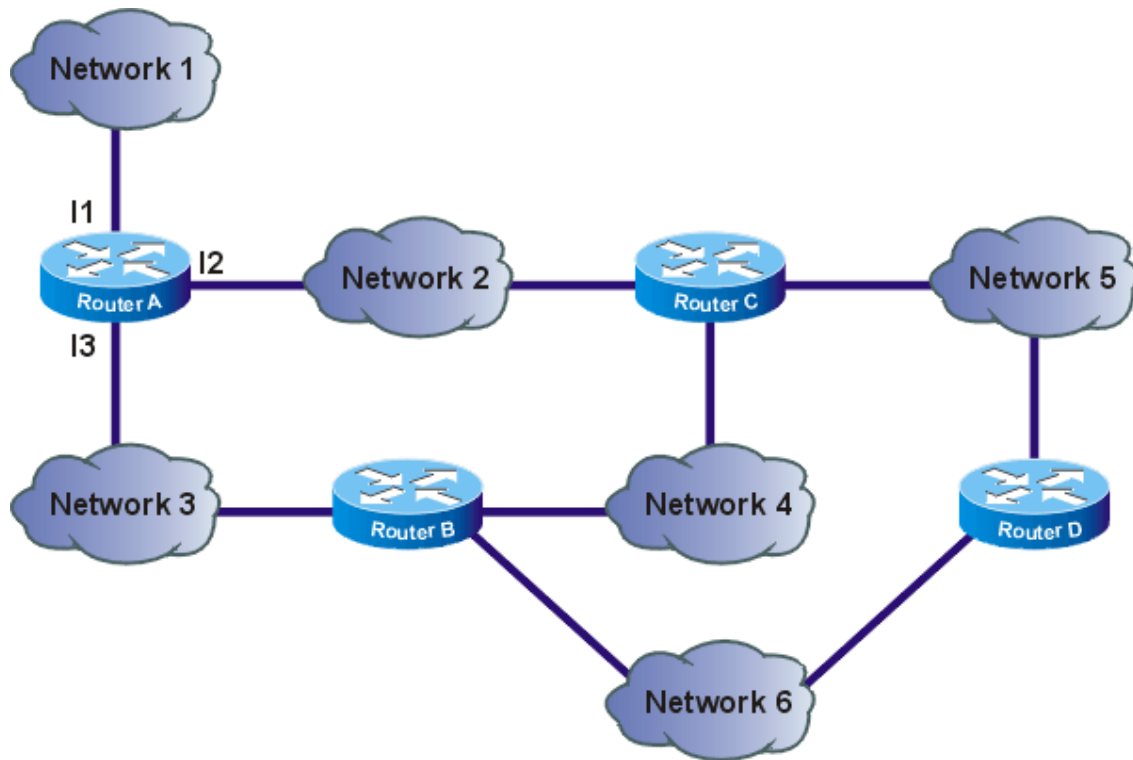


Figure 4.7 Example of alternate paths

The composite metric function computed for each data path is as follows:

$$[(K1 / Be) + (K2 * Dc)] r \quad \text{Eq. 1}$$

Where:

r = fractional reliability
 (% of transmissions that are successfully received at the next hop.)

Dc = composite delay;

Be = effective bandwidth: unloaded bandwidth x
 (1 - channel occupancy)

K1, K2 = constants.

In principle the composite delay, Dc, could be determined as follows:

$$Dc = Ds + Dcir + Dt \quad \text{Eq. 2}$$

Where:

Ds = switching delay;

Dcir = circuit delay (propagation delay of 1 bit); and

Dt = transmission delay (no-load delay for a 1500 bit message).

However, in practice a standard delay figure is used for each type of network technology. For instance, there will be a standard delay figure for Ethernet, and for serial lines at any particular bit rate.

Here is an example of how gateway A's routing table might look in the case of Fig 4.7. (Note that individual components of the metric vector are not shown, for simplicity.)

	Interface	Next hop	Metric
Network 1	I1	none	directly connected
Network 2	I2	none	directly connected
Network 3	I3	none	directly connected
Network 4	I2	C	1270
	I3	B	1180
Network 5	I2	C	1270
	I3	B	2130
Network 6	I2	C	2040
	I3	B	1180

Fig 4.8 An example routing table

The basic process of building up a routing table by exchanging information with neighbors is described by the Bellman-Ford algorithm. The algorithm has been used in earlier protocols such as RIP (RFC 1058). In order to deal with more complex networks, IGRP adds three features to the basic Bellman-Ford algorithm:

1. Instead of a simple metric, a vector of metrics is used to characterize paths. A single composite metric can be computed from this vector according to equation 1. Use of a vector allows the gateway to accommodate different types of service, by using several different coefficients in Eq. 1. It also allows a more accurate representation of the characteristics of the network than a single metric.
2. Instead of picking a single path with the smallest metric, traffic is split among several paths with metrics falling into a specified range. This allows several routes to be used in parallel, providing a greater effective bandwidth than any single route. A variance V is specified by the network administrator. All paths with minimal composite metric M are kept. In addition, all paths whose metric is less than $V \times M$ are kept. Traffic is distributed among multiple paths in inverse proportion to the composite metrics.
3. There are some problems with this concept of variance. It is difficult to come up with strategies that make use of variance values greater than 1, and do not also lead to packets looping. In Cisco release 8.2, the variance feature is not implemented. The effect of this is to set the variance permanently to 1.
4. Several features are introduced to provide stability in situations where the topology is changing. These features are intended to prevent routing loops and "counting to infinity," which have characterized previous attempts to use Ford-type algorithms for this type of application. The primary stability features are "holddowns", "triggered updates", "split horizon," and "poisoning". These will be discussed in more detail below.

Traffic splitting (point 2) raises a rather subtle danger. The variance V is designed to allow gateways to use parallel paths of different speeds. For example, there might be a 9600 BPS line running in parallel with a 19200 BPS line, for redundancy. If the variance V is 1, only

the best path will be used. So the 9600 BPS line will not be used if the 19200 BPS line has a reasonable reliability. (However, if several paths are the same, the load will be shared among them.) By raising the variance, we can allow traffic to be split between the best route and other routes that are nearly as good. With a large enough variance, traffic will be split between the two lines. The danger is that with a large enough variance, paths become allowed that aren't just slower, but are actually "in the wrong direction". Thus there should be an additional rule to prevent traffic from being sent "upstream": No traffic is sent along paths whose remote composite metric (the composite metric calculated at the next hop) is greater than the composite metric calculated at the gateway. In general system administrators are encouraged not to set the variance above 1 except in specific situations where parallel paths need to be used. In this case, the variance is carefully set to provide the "right" results.

IGRP is intended to handle multiple "types of service," and multiple protocols. Type of service is a specification in a data packet that modifies the way paths are to be evaluated. For example, the TCP/IP protocol allows the packet to specify the relative importance of high bandwidth, low delay, or high reliability. Generally, interactive applications will specify low delay, whereas bulk transfer applications will specify high bandwidth. These requirements determine the relative values of K1 and K2 that are appropriate for use in Eq. 1. Each combination of specifications in the packet that is to be supported is referred to as a "type of service". For each type of service, a set of parameters K1 and K2 must be chosen. A routing table is kept for each type of service. This is done because paths are selected and ordered according to the composite metric defined by Eq. 1. This is different for each type of service. Information from all of these routing tables is combined to produce the routing update messages exchanged by the gateways.

4.2 Link state Routing Protocols

4.2.1 General overview

Routers using link state-based routing protocols exchange link state advertisements (LSAs) which consist of a router's attached network IDs and interface costs. LSAs are advertised upon startup and when changes in the internetwork topology are sensed. LSAs are sent using directed or multicast traffic rather than broadcasting. Link state routers build a database of LSAs and use the database to calculate optimal routes which are added to the routing table. Routing information exchanged between link state-based routers is synchronized and acknowledged. The following table lists some link-state routing protocols:

Routeable Protocol	Link State-based Routing Protocol
IP	OSPF (Open Shortest Path First)
IPX	NLSP (NetWare Link Services Protocol)

Advantages of link state routing protocols:

- **Smaller routing tables.** Only a single optimal route for each network ID is stored in the routing table.

- **Low network overhead.** Link state-based routers do not exchange any routing information when the internetwork has converged.
- **Ability to scale.** Between the smaller routing tables and low overhead, link state-based routing protocols scale well to large and very large internetworks.
- **Lower convergence time.** Link state-based routing protocols have a much lower convergence time and the internetwork is converged without routing loops.

Disadvantages of link state routing protocols:

- **Complex.** Link state-based routing protocols are much more complex and difficult to understand and troubleshoot than distance vector-based routing protocols.
- **More difficult to configure.** A link state-based routing protocol implementation requires additional planning and configuration.

4.2.2 Open Shortest Path First (OSPF)

OSPF routes IP packets based solely on the destination IP address and IP Type of Service found in the IP packet header. IP packets are routed "as is" -- they are not encapsulated in any further protocol headers as they transit the Autonomous System. OSPF is a dynamic routing protocol. It quickly detects topological changes in the AS (such as router interface failures) and calculates new loop-free routes after a period of convergence. This period of convergence is short and involves a minimum of routing traffic.

In a link-state routing protocol, each router maintains a database describing the Autonomous System's topology. Each participating router has an identical database. Each individual piece of this database is a particular router's local state (e.g., the router's usable interfaces and reachable neighbors). The router distributes its local state throughout the Autonomous System by flooding.

All routers run the exact same algorithm, in parallel. From the topological database, each router constructs a tree of shortest paths with itself as root. This shortest-path tree gives the route to each destination in the Autonomous System. Externally derived routing information appears on the tree as leaves.

OSPF calculates separate routes for each Type of Service (TOS). When several equal-cost routes to a destination exist, traffic is distributed equally among them. The cost of a route is described by a single dimensionless metric.

OSPF allows sets of networks to be grouped together. Such a grouping is called an area. The topology of an area is hidden from the rest of the Autonomous System. This information hiding enables a significant reduction in routing traffic. Also, routing within the area is determined only by the area's own topology, lending the area protection from bad routing data. An area is a generalization of an IP subnetted network.

OSPF enables the flexible configuration of IP subnets. Each route distributed by OSPF has a destination and mask. Two different subnets of the same IP network number may have different sizes (i.e., different masks). This is commonly referred to as variable length subnetting. A packet is routed to the best (i.e., longest or most specific) match. Host routes are considered to be subnets whose masks are "all ones" (0xffffffff).

All OSPF protocol exchanges are authenticated. This means that only trusted routers can participate in the Autonomous System's routing. A variety of authentication schemes can be used; a single authentication scheme is configured for each area. This enables some areas to use much stricter authentication than others.

Externally derived routing data (e.g., routes learned from the Exterior Gateway Protocol (EGP)) is passed transparently throughout the Autonomous System. This externally derived data is kept separate from the OSPF protocol's link state data. Each external route can also be tagged by the advertising router, enabling the passing of additional information between routers on the boundaries of the Autonomous System.

4.2.2.1 The Topological Database

The Autonomous System's topological database describes a directed graph. The vertices of the graph consist of routers and networks. A graph edge connects two routers when they are attached via a physical point-to-point network. An edge connecting a router to a network indicates that the router has an interface on the network.

The vertices of the graph can be further typed according to function. Only some of these types carry transit data traffic; that is, traffic that is neither locally originated nor locally destined. Vertices that can carry transit traffic are indicated on the graph by having both incoming and outgoing edges.

Vertex type	Vertex name	Transit
1	Router	yes
2	Network	yes
3	Stub network	no

OSPF supports the following types of physical networks:

Point-to-point networks

A network that joins a single pair of routers. A 56Kb serial line is an example of a point-to-point network.

Broadcast networks

Networks supporting many (more than two) attached routers, together with the capability to address a single physical message to all of the attached routers (broadcast). Neighboring routers are discovered dynamically on these nets using OSPF's Hello Protocol. The Hello Protocol itself takes advantage of the broadcast capability. The protocol makes further use of multicast capabilities, if they exist. An ethernet is an example of a broadcast network.

Non-broadcast networks

Networks supporting many (more than two) routers, but having no broadcast capability. Neighboring routers are also discovered on these nets using OSPF's Hello Protocol. However, due to the lack of broadcast capability, some configuration information is necessary for the correct operation of the Hello Protocol. On these networks, OSPF protocol packets that are normally multicast need to be sent to each neighboring router, in turn. An X.25 Public Data Network (PDN) is an example of a non-broadcast network.

The neighborhood of each network node in the graph depends on whether the network has multi-access capabilities (either broadcast or non-broadcast) and, if so, the number of routers

having an interface to the network. The three cases are depicted in Figure 4.9. Rectangles indicate routers. Circles and oblongs indicate multi-access networks. Router names are prefixed with the letters RT and network names with the letter N. Router interface names are prefixed by the letter I. Lines between routers indicate point-to-point networks. On top of the figure is shown a network with its connected routers, with the resulting graph shown below.

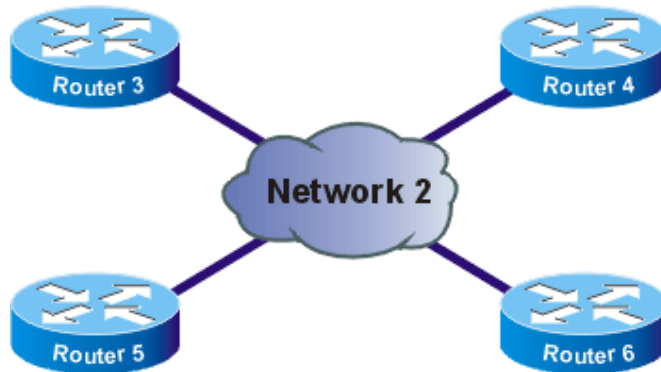
Two routers joined by a point-to-point network are represented in the directed graph as being directly connected by a pair of edges, one in each direction. Interfaces to physical point-to-point networks need not be assigned IP addresses. Such a point-to-point network is called unnumbered. The graphical representation of point-to-point networks is designed so that unnumbered networks can be supported naturally. When interface addresses exist, they are modeled as stub routes. Note that each router would then have a stub connection to the other router's interface address (see Figure 4.9).

When multiple routers are attached to a multi-access network, the directed graph shows all routers bidirectionally connected to the network vertex (again, see Figure 4.9). If only a single router is attached to a multi-access network, the network will appear in the directed graph as a stub connection.



		FROM	
		RT1	RT2
T O	RT1		X
	RT2	X	
	Ia		X
	Ib	X	

Physical point-to-point networks



		FROM				
		RT3	RT4	RT5	RT2	N2
T O	RT3					X
	RT4					X
	RT5					X
	RT6					X
	N2	X	X	X	X	

Multi-access networks



		FROM	
T O		RT7	N3
	RT7		X
	N3	X	

Stub multi-access networks

Figure 4.9 Network map components

Each network (stub or transit) in the graph has an IP address and associated network mask. The mask indicates the number of nodes on the network. Hosts attached directly to routers (referred to as host routes) appear on the graph as stub networks. The network mask for a host route is always 0xffffffff, which indicates the presence of a single node.

4.2.2.2 The shortest-path tree

When no OSPF areas are configured, each router in the Autonomous System has an identical topological database, leading to an identical graphical representation. A router generates its routing table from this graph by calculating a tree of shortest paths with the router itself as root. Obviously, the shortest-path tree depends on the router doing the calculation.

After the tree is created the external routing information is examined. This external routing information may originate from another routing protocol such as EGP, or be statically configured (static routes). Default routes can also be included as part of the Autonomous System's external routing information. External routing information is flooded unaltered throughout the AS.

OSPF supports two types of external metrics. Type 1 external metrics are equivalent to the link state metric. Type 2 external metrics are greater than the cost of any path internal to the AS. Use of Type 2 external metrics assumes that routing between AS'es is the major cost of routing a packet, and eliminates the need for conversion of external costs to internal link state metrics.

OSPF allows collections of contiguous networks and hosts to be grouped together. Such a group, together with the routers having interfaces to any one of the included networks, is called an area. Each area runs a separate copy of the basic link-state routing algorithm. This means that each area has its own topological database and corresponding graph.

The topology of an area is invisible from the outside of the area. Conversely, routers internal to a given area know nothing of the detailed topology external to the area. This isolation of knowledge enables the protocol to effect a marked reduction in routing traffic as compared to treating the entire Autonomous System as a single link-state domain.

With the introduction of areas, it is no longer true that all routers in the AS have an identical topological database. A router actually has a separate topological database for each area it is

connected to. (Routers connected to multiple areas are called area border routers). Two routers belonging to the same area have, for that area, identical area topological databases. Routing in the Autonomous System takes place on two levels, depending on whether the source and destination of a packet reside in the same area (intra-area routing is used) or different areas (inter-area routing is used). In intra-area routing, the packet is routed solely on information obtained within the area; no routing information obtained from outside the area can be used. This protects intra-area routing from the injection of bad routing information.

4.3. Autonomous Systems

In very large internetworks, it is necessary to divide the internetwork into separate entities known as autonomous systems. An *autonomous system* (AS) is a portion of the internetwork under the same administrative authority. The administrative authority can be an institution or corporation but can also be defined by the use of a routing protocol such as OSPF. The contiguous portion of an IP internetwork that is using OSPF to distribute routing information is under OSPF administrative authority and is, therefore, an OSPF AS. The AS may be further divided into domains, regions, or areas that define a hierarchy within the AS.

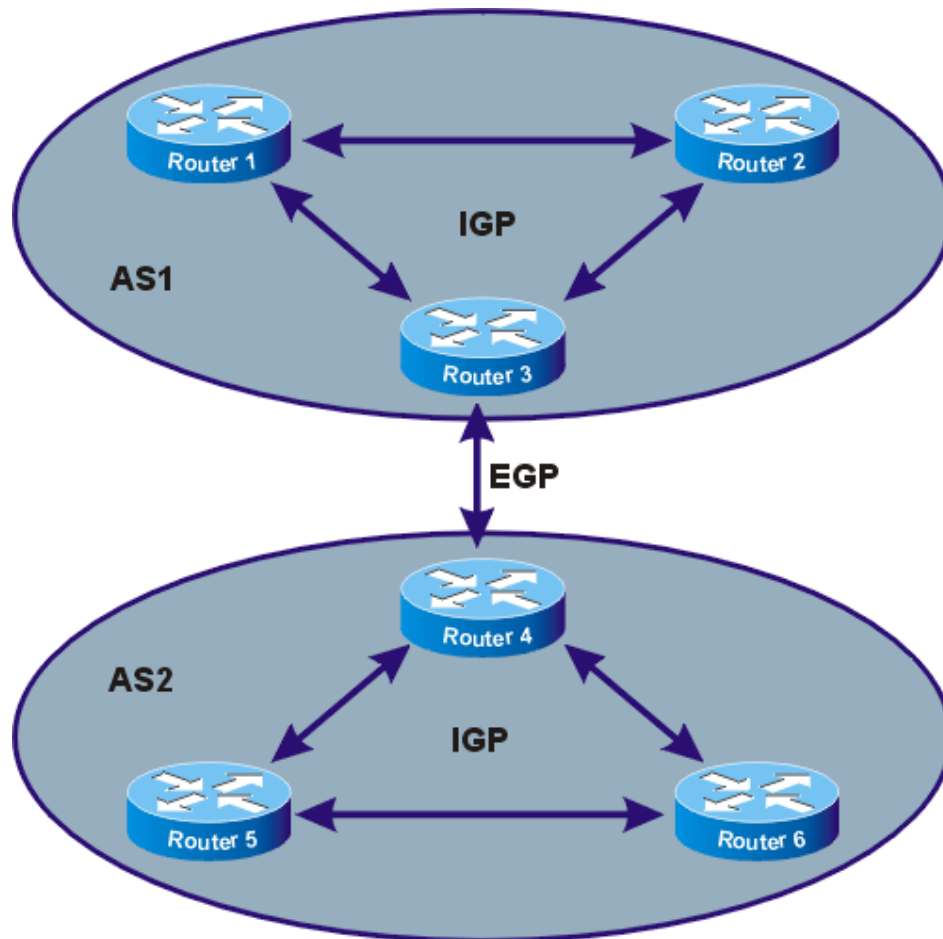


Figure 4.10 Autonomous systems, interior gateway protocols, and exterior gateway protocols

The protocols used to distribute routing information within an AS are known as *Interior Gateway Protocols* (IGPs). The protocols used to distribute routing information between ASs are known as *Exterior Gateway Protocols* (EGPs).

Interior Gateway Protocols (IGPs)

IGPs are intra-AS routing protocols. IGPs distribute routes within the AS in either a flat or hierarchical manner.

Examples of IGPs for IP internetworks are:

- **RIP for IP.** An RFC-based distance vector IGP.
- **OSPF.** An RFC-based link state IGP.
- **Interior Gateway Routing Protocol (IGRP).** A distance vector IGP developed by Cisco Systems, Inc.

Exterior Gateway Protocols (EGPs)

EGPs are inter-AS routing protocols. EGPs define the way that all of the networks within the AS are advertised outside of the AS. This can include a list of network routes in a flat routing infrastructure or a list of summarized network routes in a hierarchical routing infrastructure. EGPs are independent of the IGPs used within the AS. EGPs can facilitate the exchange of routes between ASs that use different IGPs.

Examples of EGPs for IP internetworks are:

- **Exterior Gateway Protocol (EGP).** An RFC-based EGP that was developed for use between ASs on the Internet. EGP is no longer used on the Internet due to its lack of support for complex, multi-path environments and Classless Inter-Domain Routing (CIDR).

Border Gateway Protocol (BGP). An RFC-based EGP that is currently used between ASs on the Internet. BGP overcomes the weaknesses of EGP. The current version of BGP being used on Internet backbone routers is BGP4.