1 GSW… Routeing Protocols

Routeing protocols have the job of working out the best routes to send packets, so they eventually get to their destination. The previous chapter described the difference between distance-vector protocols, link-state protocols and path-vector protocols, and introduced the Bellman-Ford and Dijkstra algorithms used in distance-vector protocols; this chapter describes some real routeing protocols, including some link-state protocols, and the algorithm that they often use: Dijkstra's algorithm.

The protocols I'll talk about are the most common ones in use at the moment: the routeing information protocol (RIP), the enhanced interior gateway routeing protocol (EIGRP), the open shortest path first (OSPF) protocol, and the border gateway protocol (BGP).

1.1 The Routeing Information Protocol

The *Routeing Information Protocol* (RIP) is a simple distance vector protocol. The protocol defines the packet formats for exchanging routeing tables between routers. It uses the Bellman-Ford algorithm to determine the optimum routes, broadcasting its routeing tables using UDP (using port 520) approximately every 30 seconds (reliable communication is not worth the overhead, and in any case you can't broadcast reliably) and deletes all routes from which no update has been received for the last three minutes. It uses split-horizons with poisoned reverse and triggered updates to speed up convergence (see the chapter on Routeing and Bellman-Ford for more details about these techniques).

RIP was one of the first routeing protocols on the Internet, replacing the time-consuming and difficult process of manually inputting and maintaining *static routes*[1]. RIP is fully described in RFC 1058.

The original version of RIP did not pass round network bit-masks, since at the time the Internet was still using classful addressing, and you could deduce the bit-mask from the network address[2]. It restricted networks to a maximum of 15 hops (a very large network in those days).

When classless addressing became popular, RIP was updated to RIPv2 (see RFC 2453) to allow bit-masks to be passed around as well, and a few other minor changes were made: for example, routeing updates were now multicast (to 224.0.0.9) rather than being broadcast, so that only other routers pick them up. There is also a version of RIPv2 called RIPng (see RFC 2080[3]) that supports the 128-bit IPv6 addresses.

Although a very simple routeing protocol to use, due to problems with the time it can take an RIP network to reconfigure (the *convergence time*), the size of the networks it can manage, and the amount of traffic required to support the protocol (routers are sending out their entire routeing tables every 30 seconds), most modern networks use other routeing protocols.

[1] Static route: one that is programmed into a router by hand, and that the router cannot update or change.

[2] See the chapter on IPv4 addressing for more details about classful addressing.

[3] You might note that 2080 is a smaller number than 2453, so does this mean RIPng came before RIPv2? Not really: RIPv2 was first specified in RFCs 1388 and 1723, and then updated to the latest version in RFC 2453.

1.1.1 RIPv2 Features

The maximum cost in any RIP network is 15. If all the costs for individual links are set to one (the most common configuration), then that limits the size of the network: any paths between two nodes with more than 15 routers between them will not be recognised. Since RIP was designed as an interior routeing protocol (to be used inside an individual organisation's network, not on the backbone of the Internet), that doesn't usually present a problem. If your network is bigger than this, don't use RIP.

For each entry in the routeing table, an RIP router needs to maintain the following information:

1. The address of the target network, and the network bit mask
2. The address of the first router along the path
3. The interface (port) out of which the packets should be sent
4. A metric (cost) indicating the total "distance" to the destination
5. A timer keeping a record of the time since the entry was last updated

Although in the examples quoted above I assumed that all routers send out their updates at the same time, in practice this is not true, and routers using RIP are free to send their routeing updates asynchronously[4]. This is fine, the Bellman-Ford algorithm will still work; in fact it usually works better, since each router gets more recent data.

You need to know the next router along in the path not only to know where to send the packets to, but also to determine when to increase the metric. A router receiving a routeing packet with a higher metric to reach a certain destination from the first router along the path cannot ignore the new cost. (Otherwise the router assumes this is information about a higher cost route, and ignores it.)

How do routers tell when a link is down? This can take some time. Since RIPv2 sends out routeing tables every 30 seconds, using UDP, some updates may get lost. Usually, a time-out value of 180 seconds is used: if nothing is heard from a router in that time, the route is marked as unavailable. This can mean several minutes of disruption for users as the network re-organises itself; this is a well-known weakness of RIP, and a key reason why other routeing protocols are more commonly used.

1.1.2 RIPv2 Frame Format

RIPv2 frames look like this:

[4] In fact they are encouraged to send their tables asynchronously. Getting all the routers in the network to send out their tables at the same time is a pretty bad idea: it can clog up the network at certain times, and it slows down convergence.

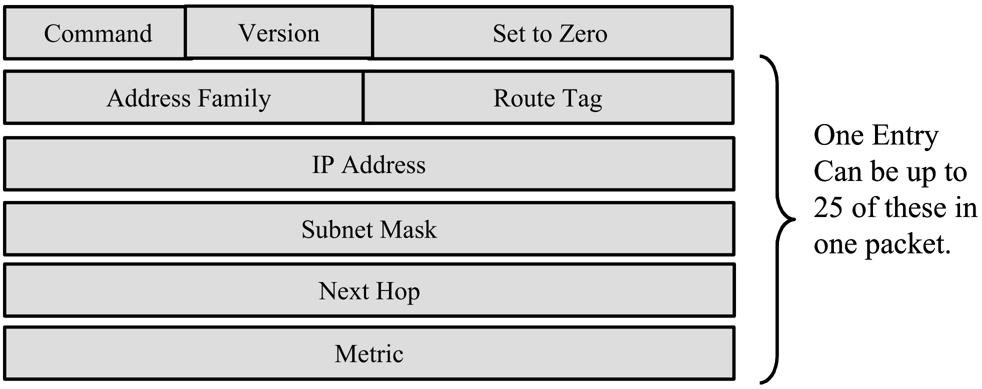

Figure 1-1 – Frame Format for RIPv2

Command can be either 'request' or 'response'. A 'request' can contain a list of routeing entries that the router wants to get a response for: in this case the response is sent immediately. Otherwise, regular 'responses' are sent approximately every 30 seconds.

The address family is set to AF_INET (2), the Internet address family[5]. The '*route tag*' field is used to highlight a route that has been imported from some other routeing protocol, and is not being updated by the RIP protocol. The '*next hop*' field allows a router to advertise a route to another network via a different router (not via itself). The other fields should be self-explanatory, I hope. (For full details, consult RFC 2453.)

1.2 The Enhanced Interior Gateway Routing Protocol

EIGRP, like RIP, is a distance-vector routeing algorithm that operates by requiring each router to tell its neighbours the content of its routeing table. It was designed to address the problems of RIP (and IGRP – the predecessor of EIGRP). In particular:

- Larger maximum hop-count, allowing use on larger networks (maximum hop-count of EIGRP is 224, as opposed to 15 using RIP).
- Allows the use of parameters such as delay[6], reliability, bandwidth, MTU[7] and load to calculate the path-cost metric, rather than just use a fixed static value (usually one)[8].
- The entire routeing table is not transmitted at each update interval – only changes to it (a copy of the neighbours routeing table is held in memory at each router).

[5] Or at least it is if RIP is being used to manage the routeing on IP networks. RIP is a flexible protocol and can be used to route other networks as well.

[6] There is a delay value assigned to each port on the router. It's a static value, not measured by the router itself. Network managers can change this parameter to affect the routes that EIGRP chooses.

[7] The maximum transmission unit (MTU) is the size of the largest packet that can be sent out of a particular port. If packets too large for a given port are queued for transmission, they are *fragmented* (split into smaller packets), and reassembled at the other end. This is expensive (in terms of time) and should be avoided wherever possible.

[8] The cost metric is very flexible, but the default is to just consider the delay and the bandwidth of the links only, and ignore parameters for reliability, MTU and load.

- *Load-balancing*. EIGRP can maintain up to six entries in the routeing table to get to the same destination, and can share traffic between them, so that not all packets going to the destination subnetwork have to go by the same route. This can also speed up reconfiguration: if one of these paths goes down, EIGRP can automatically start to use the other routes it already knows about. (RIP only remembers one route to each destination subnetwork.)

- *Automatic Route Aggregation*. EIGRP is clever enough to work out when a set of subnetworks all lie in the same direction, and then just advertise an aggregated route. For example, consider the following topology:

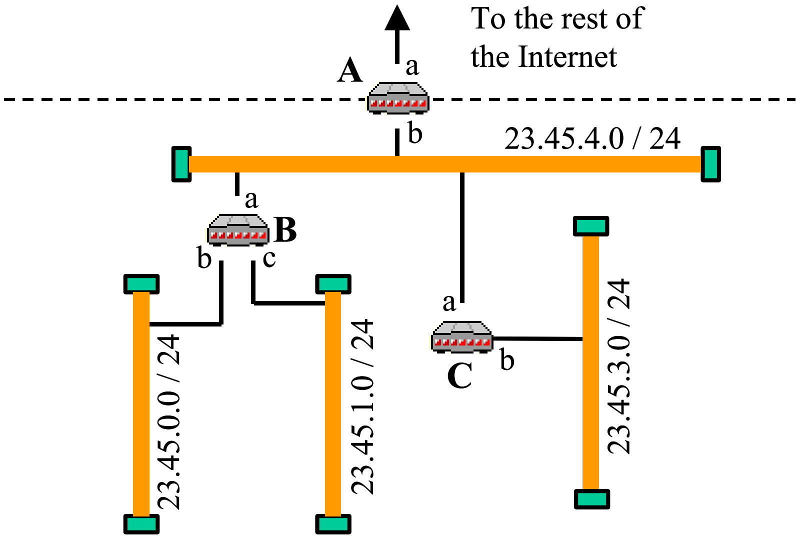

Figure 1-2 Example Network Showing Route Aggregation

In this network, there is no need for routers **A** or **C** to know that router **B** has two subnetworks attached. Router **B** could aggregate these subnetworks into the single range 23.45.0.0 / 23, and just advertise this network out from port **a**. This saves network capacity (shortening the update messages), and reduces the size of the routeing tables in routers **A** and **C**.

In this way, it can operate more efficiently over hierarchical networks than RIP.

One important point to bear in mind about EIGRP is that it is Cisco-proprietary: it only operates on Cisco equipment, and is not specified in an RFC.

1.3 Open Shortest Path First

OSPF is currently the routeing protocol of choice in small networks. It's an open standard, defined in the RFCs (originally RFC 1131, now see OSPFv2 in RFC 2328 and OSPFv3 (including support for IPv6) in RFC 2740). It's a link-state algorithm: each router builds up a map of the entire network (known as the link-state database or LSDB), and uses Dijkstra's algorithm to calculate the best routes.

While this can achieve very fast convergence (the only information that needs to be passed around the network is whether certain links are working or not: given this information, all the routers can work out their own routeing tables), it is more processor intensive.

Like EIGRP, OSPF sends out 'hello' packets at regular intervals (usually once every ten seconds) so that routers can determine when links go down, and otherwise only transmit information when requested, or when there has been a change in the network topology.

1.3.1 Hierarchical Network Design

OSPF includes the concept of hierarchy in network design, introducing the concept of a *routeing area* to limit the amount of routeing information that passes across a network. The idea is that a small change in the network probably only affects routeing tables in routers close to the disruption, and no change needs to be made in routeing tables some distance away. Therefore, there is no need to tell these more distant routers that anything has happened.

All OSPF networks have a backbone (area 0), and may have up to $2^{32} - 1 = 4.3$ billion other areas attached to the backbone (sometimes indirectly, via other areas). Each area has a distinct set of IP addresses, which are further sub-divided amongst the subnetworks in each area. Routers in one (non-backbone) area are only connected to routers in other (non-backbone) areas through the backbone, and therefore do not need to know the details of the sub-networks in any other area.

For example, consider the network below:

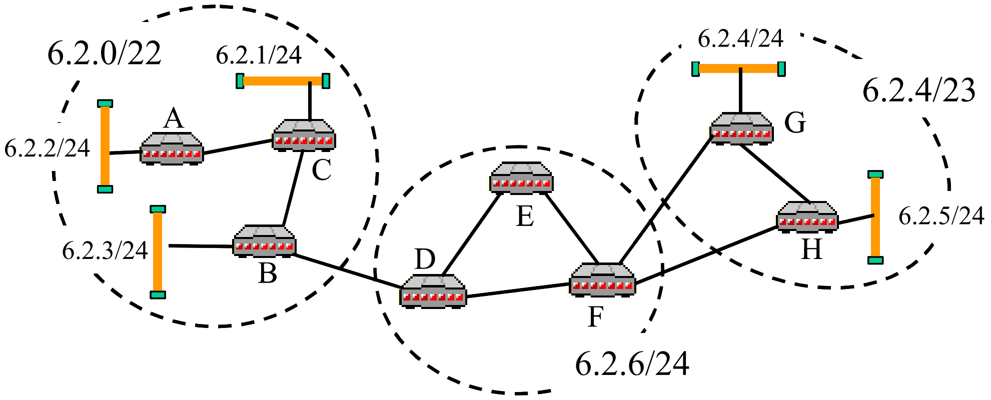

Figure 1-3 – Internal, Border and Backbone Routers

aThere are three[9] main types of routers in OSPF: internal routers (e.g. **A**, **B**, **C**, **G** and **H** in the figure above), area border routers (**D** and **F**) and backbone routers (**E**). Internal routers maintain a map of just their local area; area border routers maintain a map for the backbone and the local areas they are attached to; backbone routers just maintain a map of the backbone.

As far as the routers **A**, **B** and **C** are concerned (in region 6.2.0/22), they have no need to know that network 6.2.4/23 is split into two subnetworks. They can have a single entry in their routeing tables for 6.2.4/23, and send all packets for that range towards the backbone. OSPF is flexible enough to be able to cope with this hierarchy in the same routeing protocol, and it saves a lot of complexity in the network maps in each router (and hence processor time

[9] There's a fourth type: an autonomous system boundary router, that allows different areas to be connected together by means other than the backbone. This can be useful when there are different routeing protocols running in the same network, or there are ways to get to other places in the network other than via the backbone (area zero).

working out the optimum routes, and network capacity in passing the routeing messages around).

1.4 Border Gateway Protocol

BGP version 4 (see RFC 4271) is the routeing protocol run in the core of the Internet. It allows smaller networks known as *Autonomous Systems* (ASs) (the individual networks run by Internet Service Providers (ISPs) or large companies) to be connected together and exchange information. In BGP, all these ASs have numbers, known as *Autonomous System Numbers* (ASNs), as do all routers in the backbone of the Internet. The task of BGP is to determine the best route through all of these ASs until they reach the gateway into the target autonomous system.

Each AS can be configured to allow, or not allow, transit traffic. This allows the majority of the traffic on the Internet to be routed along central backbone links, and not through an AS that just happens to be multi-homed (i.e. have two connections to the Internet backbone).

BGP is a path-vector protocol. It doesn't store just the next hop in the routeing tables, but a sequence of the next hops through the ASs between it and the destination address. It is the characteristics of the entire path to the destination that is considered when making a routeing decision. This dramatically reduces the possibility of a routeing loop: it would be obvious to a BGP router if it was about to send a packet into a loop, since it would see its own ASN somewhere in the path. It also allows routers to be configured to avoid certain ASs.

One problem with BGP is that while it connects together ASs, it doesn't know the details of the internal architecture of the ASs. It might pick a route through one large slow AS in preference to a route through three small, fast ASs. There is no general cost of distance path metric in BGP, the routeing policies are often adjusted (sometimes manually) in response to changing traffic patterns).

1.5 Tutorial Questions

1) Explain the benefits provided by the main differences between EIGRP and RIP.

**2) The next hop field in the RIPv2 header allows a router to advertise a route via another router, not via itself. Why is this field useful? In what situations can it help?

**3) EIGRP and OSPF both use a form of triggered updates (sending information when the state of the network changes, rather than at regular intervals). In what circumstances might this be a bad idea, and reduce the performance of the network?

4) Give reasons why RIP, EIGRP and OSPF are not suitable protocols for the backbone of the Internet.