Dynamic Routing Protocols with the Zebra GPL Software

# The Paths of the Zebra

In a network with multiple paths to a target, dynamic routing decides the path an IP packet will take. Sophisticated protocols assess the quality of a connection and make routing decisions based on this knowledge. The Zebra routing software implements multiple routing protocols. BY MICHAEL PETRY

Failproof networks are every admin's dream, but reality paints a different picture of loose contacts, crashing routers, and broken lines. Dynamic routing is at its best in this kind of scenario: if an alternative route to the target host does exist, Zebra [1], and similar routing programs, are sure to find it. Dynamic routing additionally provides load balancing across multiple connections.

The routing protocol uses a selection of algorithms to find the best path to the target. So-called metrics form the basis for making routing decisions. Metrics can vary depending on the protocol. They can use the hop count or the bandwidth across individual connections to evaluate alternative paths to the target. In order to adapt to the current situation on the network, each router recalculates its routing tables at fixed intervals, thus maintaining the consistency of the table. In contrast to this, static routing requires the system administrator to define the routing table. In larger networks a configuration of this kind soon becomes unmanageable, and this can lead to inconsistencies that can cut off whole branches of an internet.

Figure 1 shows an enterprise network that we will use as our examples in the following sections. Each router knows the networks directly connected to its interfaces. For router A these are the network 10.0.1.0/24 on interface E2, 10.0.2.0/24 on E0 and 10.0.3.0/24 on E1. The routing table has a static entry for each directly connected network.

## Basic Concepts of Dynamic Routing

Router A will forward IP packets destined for one of its directly connected networks via the correct interface. But router A has no knowledge of the networks to which it has indirect access via routers B and C. In simple cases a default route is used to forward packets whose target network is unknown to a default interface. Dynamic routing implies that the participating routers will use a routing protocol to inform their peers of the networks on which they can be reached. Thus, each router on a network can create its own map of the complete network. A router will incorporate the paths described by its neighbors in its own table, provided they are newer or cheaper than its own entries. Thus every router knows all the connections and target networks on an internet.

Figure 2 shows how routers exchange information. Following an update router A's routing table now contains new entries for a few previously unknown networks: 10.0.4.0/24, 10.0.5.0/24, and 10.0.6.0/24. If every router on an internet has a complete map of the network, and if these maps are consistent, this is called a convergent network.

**Zebra**

| | |
|---|---|
| **Developers:** | Kunihiro Ishiguro and IP Infusion |
| **License:** | GPL |
| **Current Version:** | 0.93b Status: Beta (1.0 final expected soon) |
| **Protocol:** | BGP 4 and 4+, RIP v1 and v2, RIPng, OSPF v2 and v3 |
| **Architecture:** | Single server for each routing protocol family, collated by central Zebra daemon |
| **Special features:** | Configurable via text file or by telnet via an interactive VTY interface |

Dynamic routing protocols can be divided into two groups: interior and exterior protocols, where interior or exterior refers to a so-called autonomous system (AS). An autonomous system is defined as an administrative domain, and could be an enterprise, or campus network, or any network that can be viewed as a contiguous internet. Multiple routing protocols can be used within an AS, and routers will have a map of all the paths to the networks that make up the autonomous system.

## Interior or exterior pathfinding

The global Internet comprises innumerable autonomous systems. To allow routers to access networks outside of their own AS, autonomous systems are connected via exterior routing protocols (see Figure 3).

The so-called border routers positioned at the borders of each AS connect to other autonomous systems. As these routers do not need to have knowledge of all the details of the interior network structure, they only pass a brief summary of this knowledge to other border routers.

The summary comprises the address blocks accessible via individual routers, and thus ensures that the routing tables stored on the border routers do not become too large.

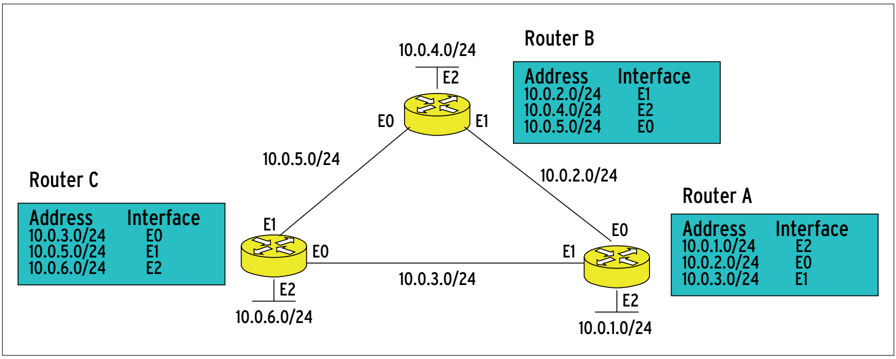
From the viewpoint of the exterior routing protocol, the whole AS is a single network. Having said that an internet connected BGP (Border Gateway Protocol) router still needs to manage about 100,000 entries in its routing table.

BGP is one of the major exterior routing protocols used, while RIP (Routing Information Protocol), OSPF (Open Shortest Path First), and ISIS (Intermediate System to Intermediate System) are typical of the interior routing protocols that we find today.

The routing protocol is not only responsible for propagating routing information, but also needs to prevent packets circulating in infinite routing loops.

The time required to propagate routes for an internet, that is, the time required for every router on an internet to compile a map of the paths on the internet is also an important criterion.

We then refer to the network as being in a convergent state. The time required



Router B

| Address | Interface |
|---|---|
| 10.0.2.0/24 | E1 |
| 10.0.4.0/24 | E2 |
| 10.0.5.0/24 | E0 |

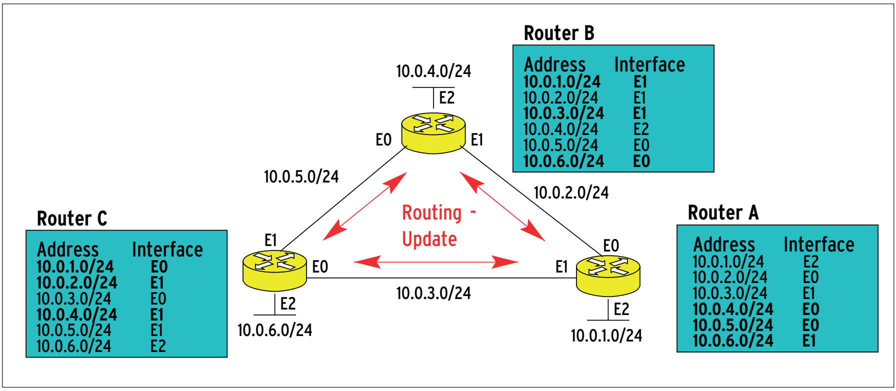
Router C

| Address | Interface |
|---|---|
| 10.0.3.0/24 | E0 |
| 10.0.5.0/24 | E1 |
| 10.0.6.0/24 | E2 |

Router A

| Address | Interface |
|---|---|
| 10.0.1.0/24 | E2 |
| 10.0.2.0/24 | E0 |
| 10.0.3.0/24 | E1 |

**Figure 1: In our sample network each of the three routers A, B, and C knows only the paths in its directly connected networks. The routing tables show the interfaces that provides access to the target networks**



Router B

| Address | Interface |
|---|---|
| **10.0.1.0/24** | **E1** |
| 10.0.2.0/24 | E1 |
| **10.0.3.0/24** | **E1** |
| 10.0.4.0/24 | E2 |
| 10.0.5.0/24 | E0 |
| **10.0.6.0/24** | **E0** |

Router C

| Address | Interface |
|---|---|
| **10.0.1.0/24** | **E0** |
| **10.0.2.0/24** | **E1** |
| 10.0.3.0/24 | E0 |
| **10.0.4.0/24** | **E1** |
| 10.0.5.0/24 | E1 |
| 10.0.6.0/24 | E2 |

Router A

| Address | Interface |
|---|---|
| 10.0.1.0/24 | E2 |
| 10.0.2.0/24 | E0 |
| 10.0.3.0/24 | E1 |
| **10.0.4.0/24** | **E0** |
| **10.0.5.0/24** | **E0** |
| **10.0.6.0/24** | **E1** |

**Figure 2: In a dynamic routing scenario routers exchange details of the networks connected to the internet. The entries added by updates are shown in bold type**

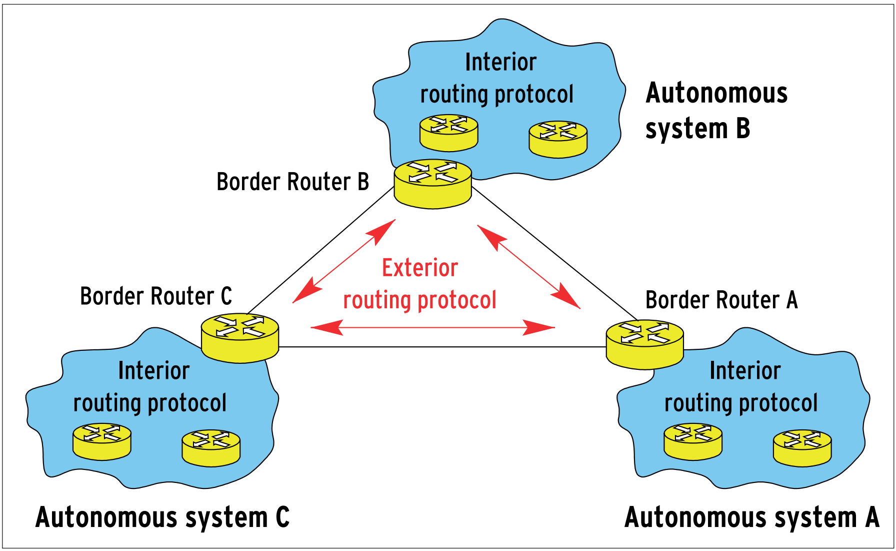


**Figure 3: Autonomous systems are connected via border routers. They communicate via an exterior routing protocol, whereas the routers within an autonomous system use an interior routing protocol**

**Table 1: Topology database for SPF**

| Router | Neighboring Router | Cost |
|---|---|---|
| A | B | 20 |
| A | C | 3 |
| A | E | 5 |
| B | A | 20 |
| B | C | 5 |
| C | A | 3 |
| C | B | 5 |
| C | D | 2 |
| D | C | 2 |
| D | E | 10 |
| E | A | 5 |
| E | D | 10 |

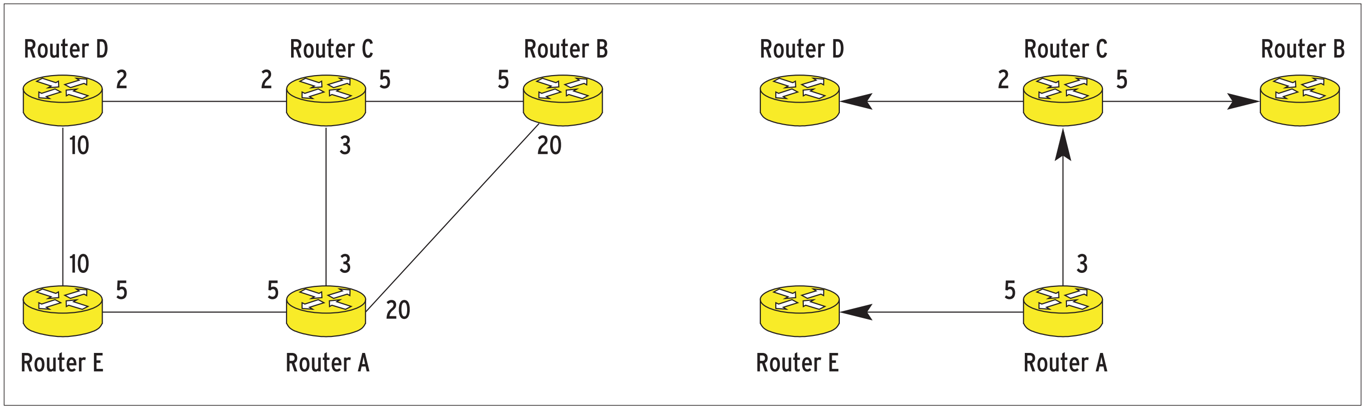


**Figure 4: A network before (left) and after (right) searching for paths with the Shortest Path First algorithm. The costs for the individual links are indicated**

to reach this state is known as a convergence period.

## Open Shortest Path First

OSPF is a link state protocol developed by the IETF (Internet Engineering Task Force). In contrast to RIP that only knows the next hop to a specific network OSPF has a map of the complete network. The RIP mechanism is comparable to a roadside, whereas OSPF has a complete atlas.

OSPF divides the autonomous system into the areas, where area 0 plays a special role: area 0 is the backbone area and is connected to every other area (see Figure 5). The advantage of this mechanism is that network convergence will happen more quickly, the less routers you have, the smaller the map. The area border router passes a list of the networks available within an area to the backbone area, thus ensuring that all the networks within the AS are reachable by the backbone. In production internets a maximum of 50 routers per area should not be exceeded in order to allow for reasonable convergence times. And it is feasible to operate smaller networks within a single area.

Neighboring routers exchange HELLO packets to maintain a constant dialog. Failures are easily recognizable as the neighboring router will not react to a HELLO packet. Each router transmits a HELLO packet to each of its neighbors every ten seconds. If four messages in succession remain unanswered, that is if a neighboring router does not react for 40 seconds, the router will assume that its neighbor is down and delete the corresponding entry from the routing table. Thus, HELLO packets are used to keep alive the connection between routers; this is why they are commonly referred to as keep alive messages.

**Table 2: Routing with SPF**

| Candidate Database | Cost | Routing Table | Cost | Description |
|---|---|---|---|---|
| **Step 1** | | | | |
| A->B | 20 | | | All neighbors of router A are added to the candidate database. |
| A->C | 3 | | | |
| A->E | 5 | | | |
| **Step 2** | | | | |
| A->B | 20 | A->C | 3 | The edge with the lowest costs (A->C) is added to the routing table. The neighbors of C are added to the candidate database. |
| A->E | 5 | | | |
| C->B | 5 | | | |
| C->D | 2 | | | |
| **Step 3** | | | | |
| A->B | 20 | A->C | 3 | The edge with the lowest costs (C->D) is added to the routing table. The neighbors of D are added to the candidate database. |
| A->E | 5 | C->D | 2 | |
| C->B | 5 | | | |
| D->E | 10 | | | |
| **Step 4** | | | | |
| A->B | 20 | A->C | 3 | The edges with the lowest costs (A->E andC->B) are added to the routing table. The algorithm stops here, as all the nodes on the network are reachable. |
| D->E | 10 | C->D | 2 | |
| | 5 | A->E | 5 | |
| | | C->B | 5 | |

## Good Neighbors

Adjacencies can occur between neighboring routers. Neighboring routers are said to be adjacent if they reside in the same area and use the same authentication string. Whether a neighboring router is adjacent or not also depends on

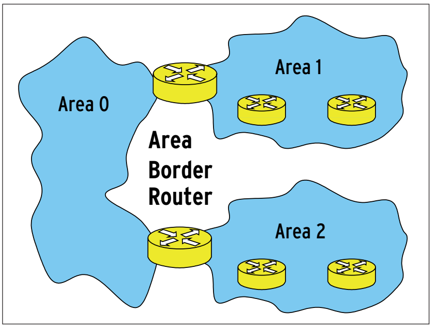


**Figure 5: All of the areas in an autonomous system are connected to the backbone area (area 0)**

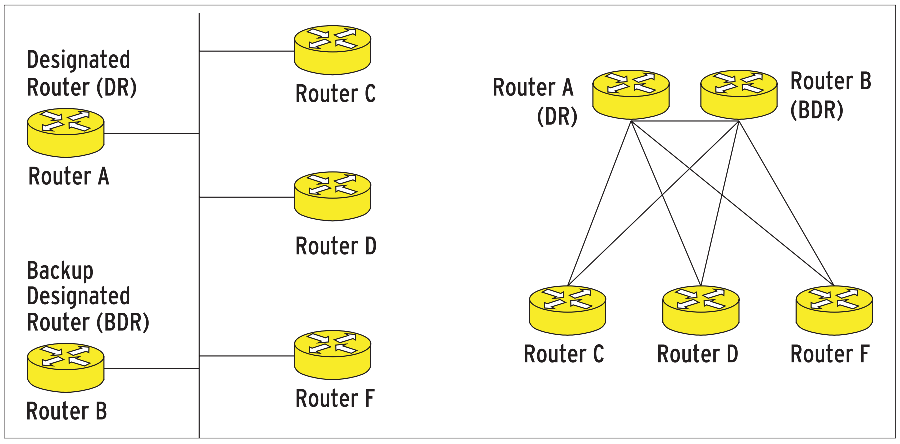


**Figure 6: On a multi-access network (left) every router is connected to every other router. OSPF sees only the connections to the DR and BDR, and assumes a star topology (right).**

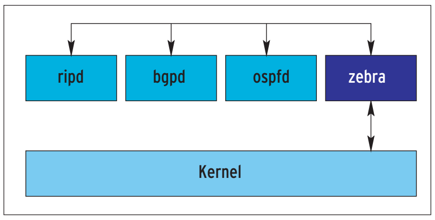


**Figure 7: The "zebra" daemon updates the kernel based routing tables and propagates information**

the type of router. In the case of point-to-point connections (P2P) both neighbors become adjacent. This is different in multi-access networks: in this case the admin specifies the designated router (DR) and a back at designated router (BDR), where adjacencies exist for every other router in the network.

It obviously makes sense to differentiate between P2P and multi-access as every router in a multi-access network (such as an Ethernet) is connected to every other router. If the protocol were to map these relationships as a grid, the complexity of the network will explode: given n routers there would be n*(n-1)/2 adjacencies.

In multi-access networks OSPF acts as if all the routers were only connected to the DR and a single BDR. This provides for a star shaped router topology as shown on the right of Figure 6. Each router set up an adjacency to be DR and the BDR, allowing the BDR to take over from the DR if it fails.

## Informative exchanges between neighbors

OSPF routers exchange LSA messages (link state advertisements) with their neighbors. LSAs are packets that contain information on interfaces, connections, and their status. When a router receives an LSA it enters the LSA information in its link state database and sent a copy of the LSA to its adjacent routers.

Thus, the link state database is like a map of the internet that contains all the connections, their status and bandwidth. Each router in the given area possesses the same map and applies Dijkstra's Shortest Path First algorithm to the map to discover the shortest paths. If the cheapest path fails, and an alternative part is available, the router will automatically use the second best route. If two alternatives of equal costs are available, that is, if the router has knowledge of two equivalent routes, it will share the load equally across both. Refer to the "SPF Algorithm" inset for a description of the algorithm.

## SPF Algorithm

The goal of the SPF algorithm (Shortest Path First) is to calculate the shortest path, or the path with the lowest costs between routers (nodes). The SPF algorithm developed by Dijkstra uses a topology database, a map of the internet that each router possesses. In the case of OSPF routing (Open Shortest Path First), the map is generated by exchanging LSA (Link State Advertisement) messages.

### The Network as a Graph with Nodes and Edges

The topology database represents the network as a graph, where routers become nodes and the paths between routers become edges. Costs are assigned to each edge. In OSPF the costs default to 100.000.000 divided by the bandwidth of the link in Bits/s. The lower a link's bandwidth is, the higher its cost. The only restriction is that costs cannot be less than 1. The total cost is calculated as the sum of the costs for the path between the originating node and the target node. Admins can define the bandwidth in Zebra using "ospf cost auto-cost reference-bandwidth value". The default value is 100,000,000.
When modifying the reference bandwidth, ensure that the same value applies to all routers on the network. The author recommends changing the default value when using Gbit network adapters to differentiate between network links running at 100 Mbit/s and 1 Gbit/s. The costs for these links would otherwise be 1 in both cases.

### Appropriate Connection Costs

Figure 4 shows a small sample network showing the costs for the individual connections. The original state is shown on the left. The minimum cost for the path between router A and router C is 3. Table 1 contains a topology database for the network in Figure 4. It shows the same information as a table. Routers use this information to calculate the cheapest path.
The router running the SPF algorithm is referred to as the root. SPF calculates the costs to all other nodes (routers) starting at this point. Our example shows the SPF algorithm running on router A and examines the individual steps.
In step 1 all neighbors of node A are added to the candidate database; these are nodes B, C, and E in our example (Table 2, Column 1, Step 1). The candidate database contains the edges that could be added to the routing table. Whether this actually happens or not, depends on their cost: only edges with minimum cost routes are added to the routing table, that is, C in our example. And, in fact, C actually appears in our routing table.

### Step-by-Step to new Candidates

C's neighbors, B and D, are evaluated as possible candidates for the candidate database in the next step. Again SPF calculates the costs and selects the edge with the lowest costs for the routing table in the next step. This loop continues until all other nodes are reachable from node A, or the candidate database runs out of candidates.
The results of the SPF algorithm can be seen in step 4. Figure 4 shows the results graphically on the right; only the lowest cost routes from router A to the other routers have been added to the routing table.

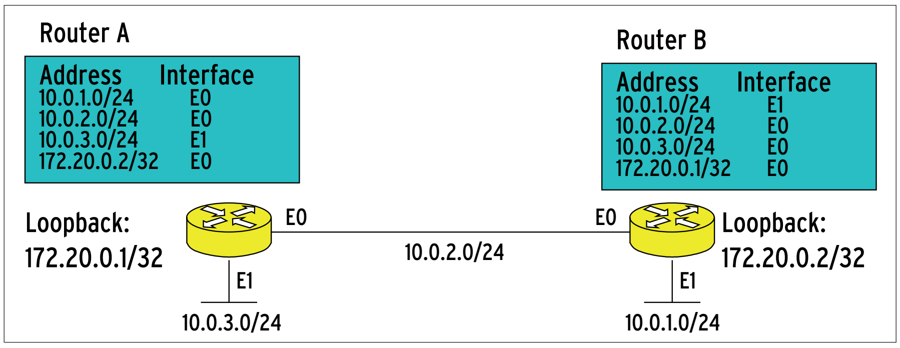


**Figure 8: In this sample network, three networks are connected by 2 routers. The 10.0.2.0/24 link is connected directly to interface 0 on both routers**

One disadvantage of OSPF is the fact that running the SPF algorithm requires considerable processing time in comparison to other routing protocols. Additionally, as the mechanism stores redundant information in various databases, is also requires fairly large amounts of memory. However, this is not a real problem given today's PCs.

Five free software packages for Linux implement the major dynamic routing protocols. Routed is the oldest and best known routing daemon for Unix. Routed is supplied with more or less every Unix version and supports RIP v1 and v2.

## Well-known Routing Packages for Linux

Gated is also well-known. This software supports all the major routing protocols, such as RIP v1 and v2, OSPF v2 and v3, as well as ISIS and BGP. Traffic Engineering APIs are available for the program, but unfortunately the software is no longer free, as the freeware version of gated has not been maintained or developed for about two years now. We will not be looking into gated in this article (however, the "Gated and Zebra" inset provides some additional detail. This also applies to the Bird [4] and MRT [5] projects, both of which are not under active development at the time of writing.

Zebra is fairly recent, and extremely promising GNU project. This modular routing software provides a daemon for each protocol it supports. The central "Zebra" server updates the kernel based routing table and propagates routing information to the individual protocol daemons. The exchange of information between routing protocols is referred to as redistribution and provides a method for propagating information on static kernel based routes via OSPF. Figure 7 shows the way Zebra co-operates with the individual daemons and the kernel. Zebra provides protocol support for RIP v1 and v2, RIPng, OSPF v2 and v3, as well as BGP v4 and 4+. OSPF v3 and BGP+ are IP v6 versions.

The Zebra project does not actually support the ISIS protocol. However, a project based at [3] is currently working on an ISIS module for the Zebra platform. Cisco [2] also provides additional information on ISIS.

## OSPF in Production Internets

The following OSPF configuration is based on the sample network in Figure 8. Routers A and B are both running an OSPF process. Both routers reside in the same area and are connected via Ethernet interface 0.

After installing Zebra (see the "Zebra Installation" inset) there are two ways of configuring the program: you can either

### Table 3: Important commands

| Command | Mode | Description |
|---|---|---|
| **Zebra** | | |
| show ip route | Enable Mode | Output the routing table |
| hostname Hostname | configuration mode | Change hostname |
| password Password | configuration mode | Change password |
| **OSPF** | | |
| show ip ospf neighbor | Enable Mode | Display status of OSPF neighbors |
| show ip ospf database | Enable Mode | Output the OSPF database |
| router ospf | configuration mode | Launch OSPF configuration mode |
| ospf router-id IP address | OSPF configuration | Set sender's address for OSPF packets |
| network address area | OSPF configuration | Defines the interfaces that OSPF OSPF |
| area number | | runs on and sets the area ID |
| passive-interface name | OSPF configuration | Do not send OSPF HELLO packets on this interface |
| **Both** | | |
| show running-config | Enable Mode | Displays the current configuration |
| copy running-config | Enable Mode | Writes the current configuration from |
| startup-config | | memory to disk |
| terminal monitor | Enable Mode | Redirects logging to the console |
| no terminal monitor | Enable Mode | Disable console logging |
| description Comment | Single configuration | Adds comments to the configuration file |

### Gated and Zebra

It has been two years since development and support were suspended for any non-commercial versions of Gated by Next Hop. The last free version 3.6.x and innumerable patches are seen online from time to time. The commercial version of gated costs somewhere in the region of 1000 US dollars and contains a single router license for RIP, OSPF, and BGP. Prices depend on the licensing model and the number of licenses.

#### Customized Intervals

Of course, you can run a network with routing software provided by a variety of manufacturers. Network operators willing to migrate and interested in testing Zebra in their existing gated router network, should be aware of the intervals at which routers exchange HELLO packets. Zebra works like a Cisco router in this respect; different intervals are needed for gated routers. Add the following lines to your OSPF configuration to remove this obstacle:

```
interface eth0
 ip ospf hello-interval 60
 ip ospf dead-interval 180
 ip ospf retransmit-interval 30
```

edit the configuration file, "zebra.conf", directly or use telnet to access the daemon interactively. Zebra provides a plain, character based interface that provides context based help when the [?] key is pressed. Command completion with [Tab] is also supported.

You can use a copy of the "zebra.conf.sample" the first time you launch the Zebra daemon (by typing "zebra -d"). The daemon will not launch without the "zebra.conf" configuration file. You can now start configuring Zebra interactively. The Zebra daemon binds to port 2601; thus, "telnet localhost 2601" provides access to the daemon. Admins with experience of IOS (Internet Operating System) on Cisco routers will soon feel at home. The Zebra daemon prompts you for a password at this point. The default password is "zebra" and the admin does not need to supply a user name.

Just like Cisco IOS, Zebra supports three modes after logging in. The first variant is a kind of user mode (also known as Normal Mode); this is the default mode directly after logging in. Normal Mode allows only a few simple commands, and the user has extremely restricted privileges. In contrast to this, Enable Mode is similar to root access for a Linux machine. At this level the admin is permitted to launch arbitrary commands. The third mode is known as Config Mode and used for configuring the router.

## Privileged

After typing the password to logon, the admin user is first placed in User Mode. To access Enable Mode, enter the "enable" command and the appropriate password. Again, this defaults to "zebra".

In Enable Mode, the user can view the current configuration by entering the "show running-config" command, for example.

The "configure terminal" command quits the Enable Mode and places the admin user in to the Configuration Mode. At any time if you want to quit a mode, type "exit", "quit" or press [Ctrl] + [D].

The first thing a responsible admin should do, is change the default password for the new router. To do so, simply enter "password my_password" while in the configuration mode. "enable password second_password" will set the enable password, and "hostname my_name" will obviously change the host name.

The next task involves the IP addresses for the network interfaces. The "interface eth0" command selects the interface "eth0"; we can then enter "ip address 10.0.2.1/24" to assign the required IP address of 10.0.2.1/24 to the interface. Finally, let us assign a dummy interface to the router.

The virtual interface offers the useful advantage of being available to all processes at all times, instead of being dependent on a physical status (such as up/down).

The command required to create this is "interface dummy0", followed by the appropriate IP address (this would be "ip address 172.20.0.1/32" for the router shown in Figure 8).

To store the new configuration, the admin user will still need to enter the "copy running-config startup-config" command while still working in Enable Mode.

After configuring the interfaces on routers A and B, interface 0 on the machine at the other end of the connection should have successfully reacted to a "ping IP address" and so indicating that the routing is correctly set.

If the admin user is working on router A, E0 should be pingable from router B, and vice versa. If not, you should start

### Zebra Installation

Follow the usual steps to compile Zebra "configure && make && make install". An installation guide is available from [1], however, the documentation is not entirely up to date. The info files provided with the source code have more recent information. Zebra performed extremely well on a 2.4.19 kernel, and proved to be extremely stable. Thus, there is no need to upgrade to a 2.2.x kernel version, as the documentation suggests.

The Linux kernel uses the cheapest route by default, opting for the first route, if multiple equivalent routes are available. Load sharing, that is the distribution of IP traffic across multiple equivalent routes, is available if you customize your kernel. To do so, enable the "IP: advanced router" and "IP: equal cost multipath" networking options before compiling.

The configure step specifies the number of equal cost routes that Zebra will add to the kernel based routing table: "./configure --enable-multipath=number". A value of "0" causes Zebra to add any potential routes.

### Listing 1: Zebra Configuration for Router A

```
!
! Zebra configuration saved from vty
! 2003/01/02 17:44:30
!
hostname Router A
password zebra
enable password zebra
log file /usr/local/etc/log.zebra.log
no banner motd
!
interface dummy0
 description Loopback-Interface
 ip address 172.20.0.1/32
!
interface eth0
 description Ethernet Interface to Router B
 ip address 10.0.2.1/24
!
interface eth1
 ip address 10.0.3.1/24
!
! The following access list restricts Zebra
! configuration to the local host.
!
access-list localhost permit 127.0.0.1/32
access-list localhost deny any
!
! The "access-class" command binds the access list
! "localhost" to the "vty" interface . "vty" is the
! interface that admin uses to log on when configuring
! Zebra interactively via telnet.
!
line vty
access-class localhost
exec-timeout 0 0
!
```

by checking your cabling and Zebra configuration.

## Routing with the OSPF Daemon

After copying "ospfd.conf.sample" to "ospfd.conf", the OSPF daemon is launched by typing the "ospfd -d" command. The demon binds to port 2604 by default. Typing "telnet localhost 2604" should set up the connection to the OSPF process. This character based interface works just like the one provided with the Zebra daemon, and also supports three modes. The following commands need to be entered in configuration mode.

The "router ospf" command enables the daemon and selected for additional configuration tasks. Having done this, the admin user will need to specify the router ID, that this, the source IP address the OSPF daemon uses when transferring information. Our example uses the virtual interface "dummy0": "ospf router-id 172.20.0.1". The "network address_area area number" command tells the OSPF process which one of their networks are reachable via this router and also specifies the area in which the router resides.

Our example uses a single area. As it represents the backbone network of a small business, this will be area 0. Interfaces that do not support OSPF should be marked as passive using the "passive-interface name" syntax to prevent them placing HELLO packets on the network.

The configuration is stored by running the "copy running-config startup-config" command in Enable Mode. Unfortunately, Zebra overwrites any comments the admin user may have added to the configuration file previously. The "show running-config" command displays the current configuration. To save yourself headaches when returning to the configuration weeks later, you should consider making liberal use of the "description" command, which adds comments and other important details to the configuration. The command is available in Configuration Mode after selecting a configuration target such as an Ethernet interface or OSPF router.

Finally, after configuring both OSPF daemons, the daemons will exchange HELLO packets and build up an adjacency. After a short convergence time all their interfaces are reachable via every router.

### Listing 2: OSPF Configuration for Router A

```
!
! Zebra configuration saved from vty
! 2003/01/02 17:53:58
!
hostname Router A
password zebra
enable password zebra
!
!
!
interface lo
!
interface eth0
!
interface eth1
!
router ospf
 ospf router-id 172.20.0.1
 passive-interface eth0
 network 10.0.3.0/24 area 0
 network 10.0.2.0/24 area 0
 network 172.20.0.1/32 area 0
!
! The following access list restricts OSPF
! configuration to the local host.
!
access-list localhost permit 127.0.0.1/32
access-list localhost deny any
!
!
line vty
access-class localhost
exec-timeout 0 0
!
!
```

## Authentic Neighbors

Authentication of neighboring routers is provided as an option. Before exchanging routing information, neighbors first prompt each other for the appropriate passwords. If the password is not returned, the adjacency fails.

In production networks each routing process is typically assigned an authentication password. This prevents any routers accidentally launched on the network from specifying incorrect paths and thus causing stability and security issues.

The configuration files "zebra.conf" (Listing 1) and "ospfd.conf" (Listing 2) contain a complete configuration for router A as appropriate to our example. Configuring router B simply entails changing the IP addresses.

## Evaluation

Zebra and a standard PC will allow you to implement a router with a throughput of 500 Mbit/s at an absolutely unbeatable price. Unfortunately, WAN adapters with speeds in excess of 2 Mbit/s are not commonly available and fairly expensive to boot. This prevents many WAN router operators from testing Linux based solutions. But it is well worth testing Zebra on a local network. Even an older machine is perfectly suited for 10 Mbit/s or 100 Mbit/s networks.

Potential redundancy is one reason for using dynamic routing protocols on enterprise networks, and the automatic configuration option, yet another. A machine could be assigned its IP address via DHCP, use this address to launch a routing process, and use the process to learn the routes to other networks from a router. If one router in a multiple router LAN were to fail, the process would immediately opt for the next best path in the routing table; thus the machine would be reachable at all times.

Zebra is easy to extend, thanks to its modular design. The admin user can replace individual routing daemons, or add new daemons – such as those provided by the ISIS project [3] for example. Due to its design, the software is interesting for designers as a platform for new module. Readers interested in a short, commented guide to the Zebra code should refer to [6]. ■

### INFO

[1] Zebra -Homepage: *http://www.zebra.org*

[2] Cisco: *http://www.cisco.com*

[3] ISIS daemon for Zebra: *http://isisd.sourceforge.net*

[4] Bird: *http://bird.network.cz*

[5] MRTD: *http://www.mrtd.net*

[6] Zebra for Dummies (Zebra Hacking Howto): *http://zebra.dishone.st/zhh.html*