

# PERFORMANCE OF ZEBRA ROUTING SOFTWARE

ROD FATOOHI

RUPINDER SINGH

Computer Engineering  
San Jose State University  
San Jose, California 95192, USA

## ABSTRACT

Zebra is publicly available free routing software that is distributed under [GNU General Public License](#). It supports the following IP routing protocols: BGP-4, RIPv1, RIPv2 and OSPFv2. Zebra is unique in its design because it has a process for each protocol that runs on a multithreaded UNIX kernel. Zebra software offers true modularity since each protocol module can be upgraded or configured independently of the others. Zebra support exists for Linux, FreeBSD, NetBSD, OpenBSD and SUN Solaris. This document covers the results of a study conducted for interoperability and functional testing of Zebra protocol modules with Cisco routers. Both systems are compared for their usability, functionality, and supported features.

**KEY WORDS:** Internet Routing Protocols, Internet Architectures, Performance Evaluation, and Interoperability.

## 1. INTRODUCTION

With the rapid growth of the Internet and the increased demands on accessing it, many organizations are struggling with providing the needed hardware and software to fulfill these demands. Many small organizations cannot afford to acquire dedicated internetworking devices (such as routers and switches) to connect to the Internet. One possible, intermediate, and even permanent, solution is to convert under-utilized workstations and personal computers (PCs) into routers by installing freely available routing software and adding Network Interface Cards (NICs) to these machines, if needed.

Zebra is a routing software package that provides TCP/IP based routing services with routing protocols support such as RIP, OSPF and BGP [9]. It supports both IPv4 and IPv6 routing protocols. It is intended to be used as a route server and a route reflector as well as a normal router [2]. The Zebra project began in 1996. Today, it is near completion and version 1.0 will be released soon. It is

distributed, free-of-charge, under the GNU General Public License. Zebra version 0.88 is used in this document.

One of the attractive features of Zebra is that it does not require dedicated hardware, such as a router, to run on. Zebra runs on several platforms, mainly PC-based platforms, such as Linux, FreeBSD, NetBSD and OpenBSD as well as SUN Solaris. A multihome computer (a host with multiple interfaces) can easily be configured as a router that runs multiple routing protocols with Zebra. Another routing software with similar capabilities is *Gated* [4]. Although both Zebra and Gated run similar platforms, they differ in supported features, configuration interface, flexibility, and source-code release policy (only older versions of Gated are now in public domain).

Zebra's system architecture is quite different from other routing systems in a sense that the routing software in these systems consists of a single process program that provides all the routing functionalities. On the other hand, Zebra uses a multi-process approach where its routing software consists of a collection of routing daemons working together to build a routing table. In Zebra, there is a routing daemon to handle each routing protocol, such as *ripd* for RIP, *ospfd* for OSPFv2, and *bgpd* for BGPv4. In addition, a master daemon, called *zebra*, is used to redistribute routes among different routing daemons and update the kernel routing table.

Zebra's modular architecture has many advantages as well as limitations. Only the routing daemons associated with the routing protocols in use need to run at any given time. New routing protocols can easily be added without affecting the whole system. In addition, these daemons do not have to run on a single machine. Here the master daemon can interact with the routing daemons through networking protocols. Moreover, multiple copies of the same routing daemon can run at a given time. On the other hand, multiple daemons demand multiple configuration interfaces. However, Zebra provides an integrated user interface shell that connects to each daemon. Another factor is performance. Multiple daemons put more load on any system than a single one. We are currently looking at this issue.

This document presents the results of a study conducted to evaluate Zebra in comparison with Cisco routers. The user interface and the main features of both systems are given in sections 2 and 3. Section 4 describes the architecture, the management and debugging features of both systems. Section 5 introduces our interoperability test bed while Section 6 covers the results of several protocol tests. Details of the routing protocols are given in [1-3, 5] while OSPF is also covered in [8] and BGP is covered in [6]. One of the problems we faced in conducting this study was that many of Zebra features are not well documented, and therefore, the only way to find out if they are supported is to conduct well-designed experiments.

For purposes of this document UNIX is used to mean any derivative of UNIX and Linux operating system. The word Zebra (with Z in uppercase) is used to denote the routing system and zebra (with z in lowercase) is used to denote the zebra process running on a UNIX kernel. In the latter, zebra is the routing manager and provides kernel routing table updates, interface configuration, and redistribution of routes between different routing protocols.

## 2. USER INTERFACE

### 2.1 INTERFACE

The first item of concern when using any system is its interface. In a Cisco router, only one system image runs and the system is accessible via telnet or thru the asynchronous console on the router itself. Zebra is implemented as multiple processes that run on top of a UNIX kernel. Zebra does not provide console access. Each routing process and the zebra process can be accessed by remote login to a specific port on the UNIX system running Zebra, where each routing process has a default port assignment such as 2601 for zebra, 2602 for *ripd*, 2604 for *ospfd* and 2605 for *bgpd*.

The user interface and command syntax for Zebra are very similar to Cisco IOS (Internetworking Operating System). Anyone familiar with Cisco IOS should experience a very short learning curve when using Zebra and vice-versa. Since all the routing protocols in Zebra are configured in their respective processes, the static routes must be configured in the zebra process.

### 2.2 USER MODES

Zebra and Cisco IOS both have two user modes – normal and enable. Normal mode is not privileged and the user can only issue a subset of show commands to view system status and statistics. Enable mode is fully privileged and the user can issue all commands including the one to configure the system. Both systems use the “>” prompt to

denote normal mode and “#” prompt to denote the enable mode.

## 2.3 CONFIGURATION MANAGEMENT

Due to the differences in architecture, Zebra varies significantly from Cisco in maintaining the system configuration. Cisco uses a single file per router for system configuration and it is stored in the NVRAM on the system. The main advantage of a single configuration file is that all system configurations are in one place and this file can be saved and retrieved via multiple methods such as TFTP, RCP, or FTP. Zebra uses one file per process to save the configuration and uses the UNIX file system. Configuration files are only readable by *root* and are all saved in one directory (*/usr/local/etc*).

## 3. FEATURE COMPARISON

Zebra version 1.0 is targeted for common unicast routing protocols such as RIP, OSPF, and BGP. Multicast routing protocols such as PIM (Sparse and Dense modes) and BGMP will be supported in Zebra 2.0. This document compares Zebra version 0.88 with Cisco IOS version 12.1.

### 3.1 RFC SUPPORT

Table 1 lists the RFC support for Zebra and Cisco IOS using the tested versions. Some of the missing features are currently under development.

**Table 1. RFC Support.**

| RFC # | Name  | Ze<br>bra | Cisco<br>IOS |
|-------|---|-----------|--------------|
| 1058  | Routing Information Protocol  | X         | X            |
| 1587  | The OSPF NSSA Option  |           | X            |
| 1771  | Border Gateway Protocol 4   | X         | X            |
| 1793  | OSPF over demand circuit  |           | X            |
| 1812  | Router Requirements   | X         | X            |
| 1997  | BGP Communities Attribute   | X         | X            |
| 2080  | RIPng for IPv6  | X         |              |
| 2283  | Multi-protocol Extensions for BGP4                                  | X         | X            |
| 2328  | OSPF version 2  | X         | X            |
| 2370  | OSPF Opaque LSA Option  |           | X            |
| 2439  | BGP Route Flap Damping  |           | X            |
| 2453  | RIP version 2   | X         | X            |
| 2545  | Use of BGP4 Multi-protocol Extensions for IPv6 Inter-Domain Routing | X         |              |
| 2547  | BGP/MPLS VPNs   |           | X            |
| 2740  | OSPF for IPv6   | X         |              |
| 2796  | BGP Route Reflection  | X         | X            |

## 3.2 SUPPORTED PLATFORMS

Zebra support exists for Linux (versions 2.0.x and 2.2.x), FreeBSD (versions 2.2.8, 3.1, and 4.x), NetBSD (version 1.4), OpenBSD (version 2.4) and Solaris (versions 2.6, 2.7). Support is planned for GNU Hurd (version 0.3). Cisco IOS runs on multiple dedicated routing platforms manufactured by Cisco Systems Inc.

## 3.3 ROUTING TABLES

Zebra can select the primary kernel routing table provided the underlying OS supports multiple routing tables, for example Linux 2.2.x. Cisco IOS has a concept of multiple routing tables in MPLS-VPN context but in the normal case only there is one global routing table running on the router.

## 3.4 OSPF

### 3.4.1 OSPF PROCESS

Major limitation in the current Zebra OSPF code (version 0.88) is the support for only one OSPF process per router. Cisco IOS can support multiple OSPF processes per router, where the limit is the number of interfaces running IP. Each process needs a unique IP address to serve as a router-id.

### 3.4.2 OSPF ROUTER-ID

Zebra OSPF and Cisco IOS let the user configure the router-id. By default OSPF would pick the highest numbered IP address to be the router-id or if a loopback interface is configured, the highest loopback address would be picked as the router-id. This is a very useful feature because OSPF process is not tied to any interface and would stay up regardless of interface failures. It is configured under the OSPF process.

### 3.4.3 RFC 1583 SUPPORT

Both Zebra and Cisco IOS conform to the latest OSPF v2 (RFC 2328) and are also backward compatible with earlier RFC 1583 implementations (RFC 1583 contains the OSPF specification). RFC1583 is disabled by default in Zebra and enabled by default in Cisco IOS.

### 3.4.4 ROUTE SUMMARIZATION

Internal route summarization is supported in both Zebra and Cisco IOS via the *area .. range* command. External route summarization is only supported on Cisco at the ASBRs when external routes are redistributed into OSPF. This is done via *summary-address* command.

## 3.5 BGP

### 3.5.1 BGP PROCESS

Cisco IOS supports a single BGP process per router. Zebra has a novel feature of supporting multiple BGP

instances on the same router. Zebra also has a concept of a BGP view. A BGP view is similar to a BGP process but the results of route selection do not get injected into the kernel routing table. Views are used to only exchange BGP routing information. This concept is unique to Zebra.

### 3.5.2 BGP ROUTE DAMPENING

We are currently testing BGP route dampening in Zebra. This is an important feature that is supported in Cisco IOS and needed in most production environments.

### 3.5.3 IGP SYNCHRONIZATION

Zebra does not require synchronization with the IGP and will announce the specified route even if it does not exist in the router IGP. By default, Cisco IOS requires that the route exist in the IGP before it is propagated to other BGP peers. If the router does not have a destination route in its IGP, it may receive traffic that it's unable to deliver. This behavior can be overridden in IOS by the command *no synchronization*.

### 3.5.4 BGP CAPABILITIES NEGOTIATION

Both Zebra and Cisco IOS support the BGP4+ capabilities negotiation feature.

### 3.5.5 BGP SOFT RECONFIG

Older BGP implementations required that the TCP connection between BGP peers be reset before any policy changes could take effect. This practice was very disruptive for the network stability. Fortunately, now both Zebra and Cisco IOS support *soft reconfiguration* which allows the new policies to take effect without resetting the peers.

### 3.5.6 EBGp MULTIHOP

EBGP requires that the BGP peers must be physically connected to each other. However, situations may arise when physical adjacency is not possible but IP connectivity exists. For example, two BGP routers are connected by non-BGP speaking routers. Both Zebra and Cisco IOS let the user overcome this restriction by letting the user configure *multihop* EBGP.

## 4. USABILITY

### 4.1 ARCHITECTURE

Zebra is implemented as multiple daemons working together to provide routing functionality on a UNIX kernel. Zebra is more modular and provides easier and independent upgrades than IOS. Also, each protocol can be independently restarted without affecting other protocols. Cisco IOS uses a single image file for system code. User has to reboot the system in order to upgrade any one component in the code.

## 4.2 MANAGEMENT

SNMP work is under progress in Zebra. Zebra itself is not an SNMP agent but provides routing protocol MIBs in conjunction with an external agent that supports the SMUX protocol, specified in RFC1227. It is recommended that the *ucd-snmp* software (from University of California at Davis) be used for SNMP agent.

Cisco IOS provides full SNMP agent functionality and supports routing protocol MIBs. In addition, it also supports the UNIX *rcp* and *rsh* capability to remotely configure and manage a Cisco router from a UNIX host. This can come in very handy for bulk configuration changes and routine administrative tasks.

## 4.3 LOGGING

Both Zebra and Cisco have extensive logging capabilities to log system events and errors. Since Zebra runs on a UNIX like system, it has access to the underlying file system and can directly log to a file. This option is exclusive to Zebra and can be configured.

## 4.4 DEBUGGING

In Zebra a user can use *tail -f* UNIX command on the log file. It is easier to change interface parameters and debug interfaces on Cisco routers since the driver is integral to the code.

## 4.5 COMMANDS

Both Zebra and Cisco IOS provide the very useful “?” suffix after any command to display any options or the next required parameter. Zebra exclusively features the *list* command. This command is very useful since it lists all the possible commands in a given context. For example, in the enable mode, it lists all the commands that a user can issue. In the *config* mode, it lists the *config* mode commands. The *list* command helps the user to rapidly get familiar with Zebra and its capabilities.

## 5. INTEROPERABILITY TEST BED

### 5.1 TOPOLOGY

The network of our test bed, as shown in Figure 1, comprises two Linux servers running Zebra software, seven Cisco routers, and one Cisco router acting as a frame relay switch. The layer 2 protocols include Ethernet, Token Ring, HDLC, and Frame Relay. Layer 3 protocol used is IP. The routing protocols used for interoperability testing are RIP, OSPF and BGP. Table 2

lists all the hardware and software components of the test bed.

The Frame Relay routers are fully meshed with each router having PVCs terminate on the remaining three. This creates a software configurable way to select which route the data must take. For example, in the normal case Router\_D can directly reach Router\_F since it's one hop away. If on Router\_D, the PVC connecting to Router\_F is taken out of service, then data will find an alternate path through Router\_E or Router\_G.

## 5.2 ZEBRA AND LINUX

Zebra system runs as multiple processes on top of a Linux/UNIX kernel. The zebra process is the interface to the OS kernel. So the routes that the Linux kernel knows are derived from zebra, which in turn gets them from the protocol specific modules or processes.

## 5.3 CAVEATS

We ran into the following issues while bringing up the network and should be watched out. First, IP forward must be turned on the Linux box running Zebra software. Second, Zebra OSPF failed to make an adjacency on a Token Ring interface (Zebra\_2 <-> Router\_C). Zebra received a HELLO packet from Router\_C but did not send one to Router\_C so it was stuck in the *Init* state, with one-way connection. The fix is on the Cisco Token Ring router to use the command “ip multicast use-functional”. And third, the MTU size has to match on the Token Ring segment on OSPF adjacency, which won't come up and will be stuck in the EXCHANGE state. Here the fix is to change the MTU to match on Linux and the adjacency will come up.

## 6. INTEROPERABILITY

This section covers the various network scenarios that we tested in our test bed. All the Zebra routers are running Zebra 0.88 software and the Cisco routers are running 12.1(4) release of IOS.

### 6.1 RIP

In this scenario, we run RIPv2 on all routers. By default Zebra supports RIPv2 and Cisco supports RIPv1. Correct protocol version must be specified on both platforms for proper operation. Zebra\_2 joins the two Cisco “clouds” (the first cloud has Routers A, B and C while the second cloud has Routers D, E, F, and G). All masks are 255.255.255.0 or /24. Zebra\_2 is in the middle and sees all the routes in the network. This experiment was successful and full connectivity was achieved.

## 6.2 OSPF-RIP

This scenario does OSPF-RIP redistribution on Zebra\_2. The routing protocols run on the routers are given in Table 3.

**Table 3. OSPF-RIP Scenario.**

| OSPF     | RIPv2    |
|----------|----------|
| Zebra_2  | Zebra_2  |
| Router_A | Router_D |
| Router_B | Router_E |
| Router_C | Router_F |
| Zebra_1  | Router_G |

All OSPF routers are in the backbone, area 0. The routing tables and configurations for Router C, Zebra\_2, and Router G show all the routes as expected.

## 6.3 BGP

This scenario connects two AS clouds with BGPv4 running on Zebra\_1 and Zebra\_2. The IGP for AS1 is OSPF and the IGP for AS2 is RIPv2. BGP routes are redistributed into the IGPs, given in Table 4, for complete connectivity in the network.

**Table 4. BGP Scenario.**

| BGP     | AS 1     | AS 2     |
|---------|----------|----------|
|         | OSPF     | RIPv2    |
| Zebra_1 | Zebra_1  | Zebra_2  |
| Zebra_2 | Router_A | Router_D |
|         | Router_B | Router_E |
|         | Router_C | Router_F |
|         |          | Router_G |

The routing tables and configurations for Zebra\_1 and Zebra\_2 show all the routes as expected.

## 7. CONCLUSION

Zebra is easy to use and provides many advanced routing features. By splitting the routing protocols into separate processes, it provides modularity and stability to the whole system. Zebra is open source and is extensible.

Zebra is freely available and runs on publicly available operating systems. This extends the power of Zebra to users who typically can't afford expensive dedicated routers. We see Zebra being utilized extensively in the academic and research communities while serving in production service provider networks.

We plan to continue this study by comparing Zebra with *Gated* and examining other parameters of PC-based routers such as performance measurements, reliability, scalability, and stress analysis.

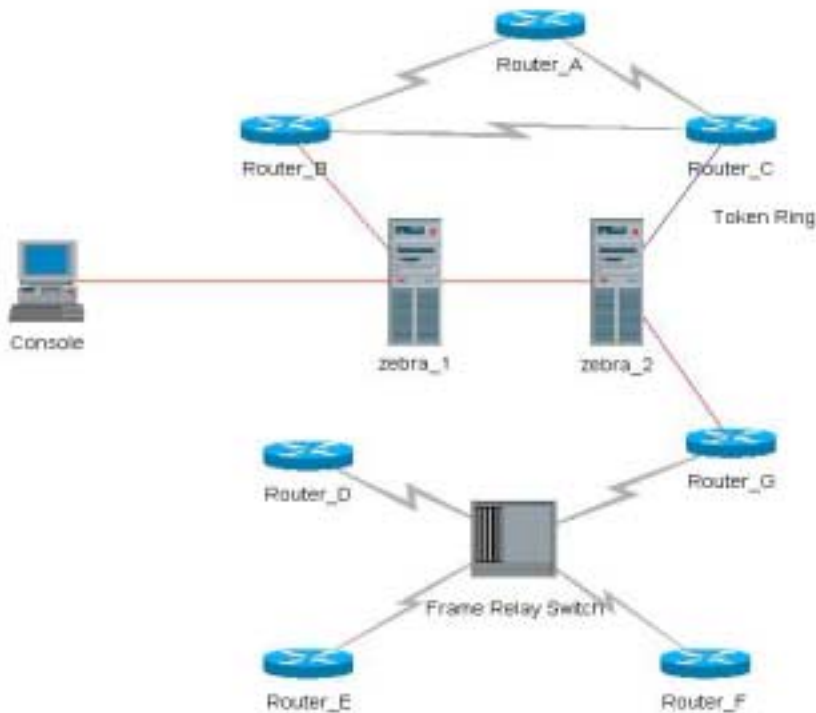
## ACKNOWLEDGEMENTS

We would to thank IPInfusion, Inc, especially Kunihiro Ishiguro, Chief Technology Officer, and Matt Crosby, Director of Business Development, for providing assistance.

## REFERENCES

- [1] Comer, D., *Internetworking With TCP/IP, Volume 1: Principles, Protocols, and Architectures*, 4<sup>th</sup> Ed., Prentice hall, 2000.
- [2] Halabi, B., *Internet Routing Architectures*, 2<sup>nd</sup> Ed., Cisco Press, 2000.
- [3] Huitema, C., *Routing in the Internet*. Prentice Hall, 2000.
- [4] Merit Gated Consortium Site, <http://www.gated.org>.
- [5] Perlman, R., *Interconnections: Bridges and Routers*. Addison-Wesley, 2000.
- [6] Rekhter, Y., and Li, T., *A Border Gateway Protocol 4 (BGP-4)*, RFC 1771, 1995.
- [7] Stevens, W. R., *TCP/IP Illustrated Volume 1*, Addison-Wesley, 1994.
- [8] Thomas II, Thomas M., *OSPF Network Design Solutions*, Cisco Press, 1998.
- [9] Zebra Developers Site, <http://www.zebra.org>.

**Figure 1. Network Topology for the test bed.**



**Table 2. Hardware and software components of the test bed.**

| Name           | Hardware       | OS Version         | SW Version           | Interfaces         |
|----------------|----------------|--------------------|----------------------|--------------------|
| Router_A       | Cisco 2516     | IOS 12.1(4)        | c2500-js-1.121-4     | 1Eth, 2 Serial     |
| Router_B       | Cisco 2500     | IOS 12.1(4)        | c2500-js-1.121-4     | 1Eth, 2 Serial     |
| Router_C       | Cisco 2500     | IOS 12.1(4)        | c2500-js-1.121-4     | 1 TokenR, 2 Serial |
| Router_D       | Cisco 2500     | IOS 12.1(4.2)T     | c2500-js-1.121-4.2.T | 1Eth, 2 Serial     |
| Router_E       | Cisco 2500     | IOS 12.1(4)        | c2500-js-1.121-4     | 2 TokenR, 2 Serial |
| Router_F       | Cisco 2500     | IOS 12.1(4)        | c2500-js-1.121-4     | 1Eth, 2 Serial     |
| Router_G       | Cisco 2500     | IOS 12.1(4)        | c2500-js-1.121-4     | 1Eth, 2 Serial     |
| Zebra_1        | PC 400 Mhz PII | Linux (Redhat 6.1) | Zebra 0.88           | 3 Eth              |
| Zebra_2        | PC 366 Mhz PII | Linux (Redhat 6.1) | Zebra 0.88           | 2 Eth, 1 TokenRing |
| Frame Relay SW | Cisco 2522     | IOS 12.1(4.2)T     | c2500-i-1.121-4.2.T  | 1 Eth, 8 Serial    |