

# A Redundant Architecture for Routing Protocols



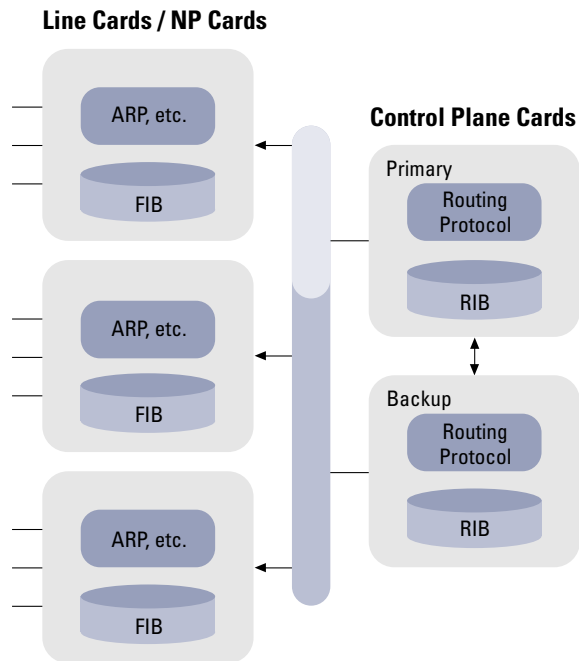
**ip** infusion™  
white paper

## Introduction

As service providers (SPs) continue to compete for customers, one factor that will become increasingly important is that of *High Availability (HA)*. This is largely due to the migration of applications such as voice and video transfer, from traditional telecommunication networks to IP based networks. In order to help *guarantee* reliable service to their customers, SPs will ensure that the equipment they purchase has built-in HA support.

Unfortunately, IP networks—and IP routers—were designed with best-effort mindset and are thus not suited for systems that must be highly reliable. To satisfy High Availability, IP routers must be robust to events such as hardware and software failures (or updates) and be available 99.999% of the time.

The most common approach to satisfying HA is to make a system *Fault Tolerant (FT)*—or to ensure there is no single point of failure. This usually requires provisioning backup copies of all hardware and software so that there is a backup component in the event of failure.



**Figure 1** Architecture Diagram

In the case of an IP router, *all* components must be backed up, including physical connections, line cards, control plane cards, power supplies, etc. This document examines the hardware architecture of a chassis designed for High Availability and identifies the requirements of adding HA support to routing protocols in this environment. This document then compares and contrasts the three most popular designs for adding this support.

### **Control Plane (CP) Redundancy**

Today, several companies are aggressively developing high performance, yet highly reliable IP routers. This is usually achieved using a combination of control plane cards and line cards (or Network Processor (NP) cards), connected with a reliable bus and enhanced operating system.

In order to operate in this environment, routing protocol software and other control plane applications must be modified to understand the new architecture and interact properly with it. Figure 1 illustrates this high level architecture.

In the event that a primary line card or control plane card fails, control is easily switched to one of the backups.

### **Architecture Characteristics**

In order to clarify the requirements for providing routing protocol redundancy, it is necessary to identify all architecture characteristics that are relevant to the control plane blades and applications running on them.

1. Multiple Control Plane Blades—A separate instance of each control plane application will run on each redundant control plane blade. Each blade will contain its own CPU, memory, etc.
2. Separation of Forwarding Plane—The Forwarding Plane (FP) is considered to be separate from the control plane. In several systems, it will be physically separate, however, this is not assumed. In either case, the FP is considered to be logically independent.
3. CP Connection—The control plane blades will be connected using a backplane, bus, or similar medium. Communication between processes running on different CP blades will be provided by the operating system.
4. Separate Management Entity—The system management should be controlled by a separate entity that is capable of communicating with all CP blades and line cards. It should configure redundant instances of control plane applications identically.
5. Other System Redundancies—As per the definition of fault tolerance, it is assumed that redundancy support exists for all other parts of the system.

## CP Redundancy Requirements

In order to further define routing protocol redundancy, it is imperative that we identify the requirements for redundant applications running on multiple CP blades. (Below, the term *sync-data* is defined to be any internal data that is synchronized between the *Primary* and *Backup* instances.)

1. Applications must establish communication with other instances of the same application running on different CP blades.
2. When functioning as *Primary*, applications must send update messages to the *Backup* instances whenever sync-data changes occur.
3. When functioning as *Primary*, applications may receive synchronization requests from *Backups* and must send a copy of (or subset of) the sync-data to the *Backup(s)*.
4. When functioning as *Primary* and a shutdown event is received (for CP blade shutdown, software upgrade, etc), a notification must be sent to the *Backup* instance(s).
5. When functioning as *Backup*, the application must respond to 'Primary down events' and transition to *Primary* (or failover).
6. When functioning as *Backup*, applications must request synchronization from the *Primary* instance when first brought online. They may also periodically request synchronization during normal operation.

## Design Considerations

The requirements just presented attempt to make routing protocols fault tolerant, such that if one instance fails, another can takeover immediately, resulting in minimal disruption to the forwarding plane and the attached networks.

There are several strategies that have been considered for adding this support to routing protocols and other control plane applications. This paper compares the three most popular strategies and highlights their relative advantages and disadvantages.

### Packet Duplication

This strategy involves no communication between the routing protocol instances. Note that, as a result, this strategy does not conform to the requirements presented above, however, it aims at achieving the same goals.

With this strategy, the lower levels of the system are responsible for duplicating ingress packets and passing a copy to each of the CP blades. For egress packets, the OS or IP stack simply discards packets originating from a backup blade. This model uses systems theory such that each instance starts with the same initial state; processes the same events or packets in the same order; and should therefore maintain consistent state at all times.

*Advantages:* The advantages to this approach are that very few modifications are required for the routing protocols and other CP applications.

*Disadvantages:* The main disadvantage to this approach is that it places significant stress on the lower levels of the system (i.e., the forwarding plane) in order to duplicate packets and pass around multiple copies. The performance advantages of the architecture are diminished, which causes most system designers to reject this option out-of-hand.

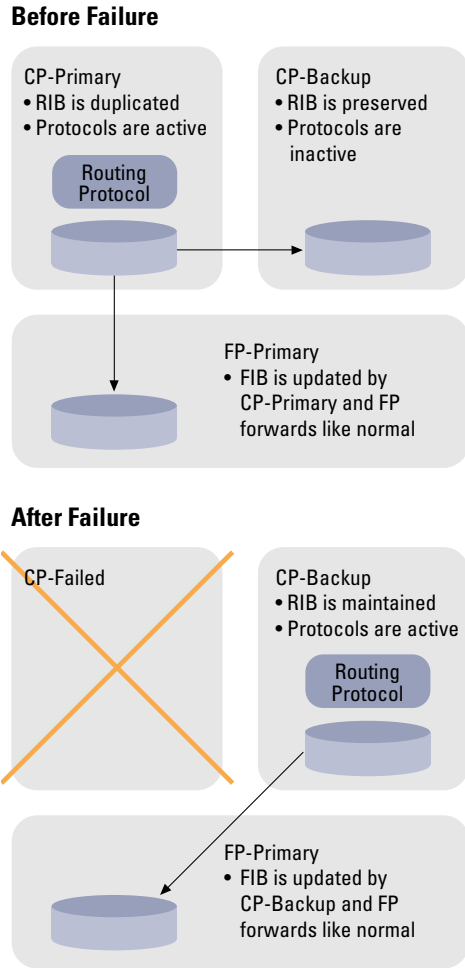
A second disadvantage of this strategy is that it does not allow for *Backup* instances to be started after the *Primary* is already configured and running. This is a serious restriction and unacceptable to most designers.

### Complete Protocol Synchronization

This strategy involves a complete synchronization between the instances of each routing protocol. Whenever the *sync-data* of any protocol changes, a message is sent from the *Primary* to the *Backup(s)*. In theory, if a primary CP blade fails, then a backup blade will takeover, and each protocol should assume the responsibilities of the failed instance with no impact to the network, neighboring routers, etc.

*Advantages:* In the event of failure, this approach would result in the least impact to the network and least amount of recovery time required for the backup instances.

*Disadvantages:* The most serious disadvantage of this strategy is the performance impact on the routing protocol applications. In order to work correctly, all synchronization updates



**Figure 2 Failover Scenario**

must be atomic, meaning that the Primary must know that each update was received and processed by each Backup exactly once. If packets are lost, the protocol instances could become out-of-sync and unpredictable events could occur during failover. Ensuring this atomicity will require complicated database-style techniques that will slow performance.

A second disadvantage to this approach is that it does not work for protocols like BGP that use TCP connections to communicate with peers. During primary failure, the connections will be broken, causing the peers to reconfigure. The only way to prevent this is by using a specialized TCP/IP stack that is able to preserve TCP connections during failover. Unfortunately, no commercially available TCP/IP stack currently implements this capability.

A final consideration is that this strategy may be overkill. The total amount of data to synchronize could be enormous, resulting in increased stress on system resources (bus, backplane, etc) that may not be desirable to system designers.

### **RIB Synchronization with Graceful Restart**

In most cases, the disadvantages of the first two strategies are unacceptable and warrant a third option.

This strategy focuses on leveraging the benefits of the protocol synchronization strategy, while reducing the disadvantages to a reasonable level.

The goals of the RIB Synchronization with Graceful Restart strategy are:

- To preserve the forwarding plane during Primary failure so that the router is able to continue forwarding packets.
- To minimize impact to the network.
- To minimize the performance impact on the routing protocols.
- To discover a strategy that will work for protocols like BGP.
- To impose no restrictions on the lower levels of the system.

This strategy involves synchronizing the Routing Information Bases (RIBs) between the *Primary* and *Backup* CP blades. This preserves the most important information from all routing protocols (ie, the routes), and the system is able to continue forwarding during *Primary* failure.

In addition, the RIB is usually controlled by a Routing Table Manager (RTM), which runs outside of the routing protocol tasks. All changes should be localized to this RTM and should result in minimal modification to the protocols themselves. As a result, there should be no performance impact on the routing protocols. Complicated synchronization techniques guaranteeing reliable delivery of messages can usually be avoided using other methods, so the performance impact on the RTM should be minimal.

A potential disadvantage of this strategy is that the routing protocols must re-establish connections and rebuild their internal databases after a failure

occurs. To minimize the delay, this strategy leverages various protocol Graceful / Hitless Restart extensions. During *Primary* failure, the neighbors will timeout before reconfiguring, allowing the *Backup* to come online and re-establish the connections. The new protocol instance is able to rebuild its database more quickly, and the failure does not impact the network.

Figure 2 (see previous page) illustrates this strategy before and after a Primary CP blade failure.

*Advantages:* The advantages of this approach are that it solves all goals established previously. It allows the router to continue forwarding during failure, minimizes impact to the network, and does not degrade protocol performance. Furthermore, it is the best possible solution for protocols like BGP.

*Disadvantages:* The disadvantages of this approach are that it does not preserve as much information as the *Protocol Synchronization* strategy defined previously, and may result in slightly longer downtime for non-TCP protocols. Also, the benefits of Graceful / Hitless Restart can only be realized if *all* routers in the network implement the feature.

## Conclusion

This paper discusses the architecture of several fault tolerant routers being developed today, and presents three alternatives for adding redundancy support to routing protocol applications.

The first approach—*Packet Duplication*—is usually rejected immediately by system designers due to the impact on the lower levels of the system.

The second approach—*Protocol Synchronization*—has been investigated by several companies and may prove useful for protocols like OSPF that do not use TCP. However, this usually comes at the expense of protocol performance and increased stress on the system bus. Furthermore, this strategy is useless for protocols like BGP that use TCP.

The final strategy—*RIB Synchronization with Graceful Restart*—leverages many of the benefits of *Protocol Synchronization*, while minimizing its disadvantages. The router is able to continue forwarding during a failure, protocol performance does not suffer, system resources are not impacted, and it is the best solution for protocols like BGP. Furthermore, because it leverages industry standard protocol extensions, it is the one most likely to be accepted by the routing community.

RIB Synchronization with Graceful Restart is considered to be the most feasible solution for adding redundancy support to routing protocols.



IP Infusion Inc.  
111 W. St. John Street  
Suite 910  
San Jose, CA 95113  
tel: 408-794-1500  
fax: 408-278-0521  
marketing@ipinfusion.com  
www.ipinfusion.com

© Copyright 2002 IP Infusion Inc. All Rights Reserved.  
IP Infusion, ipinfusion and ZebOS are trademarks of  
IP Infusion Inc. All other brands or product names are  
trademarks or registered trademarks of their respective  
holders. All specifications within this document are  
subject to change without notice.

Part No. 01930001