Realtime publishers

The Essentials Series: The Business Benefits of Rapid Application Failover

The Power of Rapid Application Failover

sponsored by

ARCserve® More than Backup

by Greg Shields

Copyright Statement

The Power of Rapid Application Failover

The previous article in this series outlined in distressing detail the peril associated with a data-centric focus on data protection. That article explained how IT organizations must apply as much priority to preserving application availability as they currently do on the data itself. In effect, organizations who value the continued functionality of their data center operations must look to modern solutions for application protection as much as data protection.

Rapid application failover comprises a solution set that has grown into maturity over a long history. You'll find that there are actually multiple techniques and technologies that merge to enable the rapid failover of applications. This combination of techniques is laid in place to protect the IT environment against several outage scenarios. At the same time, these tools also ensure that your applications' needed data is present in the backup facility should a failover occur. This article continues the discussion through an analysis of the mechanisms in which these technologies function as well as best practices for their implementation.

The Technologies Behind Rapid Application Failover

The best way to appreciate the power of rapid application failover is to understand its technology underpinnings. With a good understanding of how this high-availability infrastructure works, you'll find multiple scenarios where this technology fits in protecting your environment's availability.

Let's first break down the solution into the different problem sets it overcomes. To protect against any service interruption scenario, there are a number of scenarios to consider in your planning.

Application Failover

If a business experiences a server crash, configured applications on that server must fail over to a backup location. This failover includes the processing of the application itself as well as the internal state information within that application. Accomplishing this failover involves the real-time monitoring of file systems and internal registry values. In the case of a disaster, this same process must occur across all the servers and applications in the lost site.

From a technology perspective, this application failover is commonly accomplished through the use of a file system filter driver that is installed to the same server where an application is hosted. Because this filter driver works at a very low level, interacting with the file system itself, it can very discretely gather changes to the system and replicate those changes over the network to a standby server (see Figure 1).

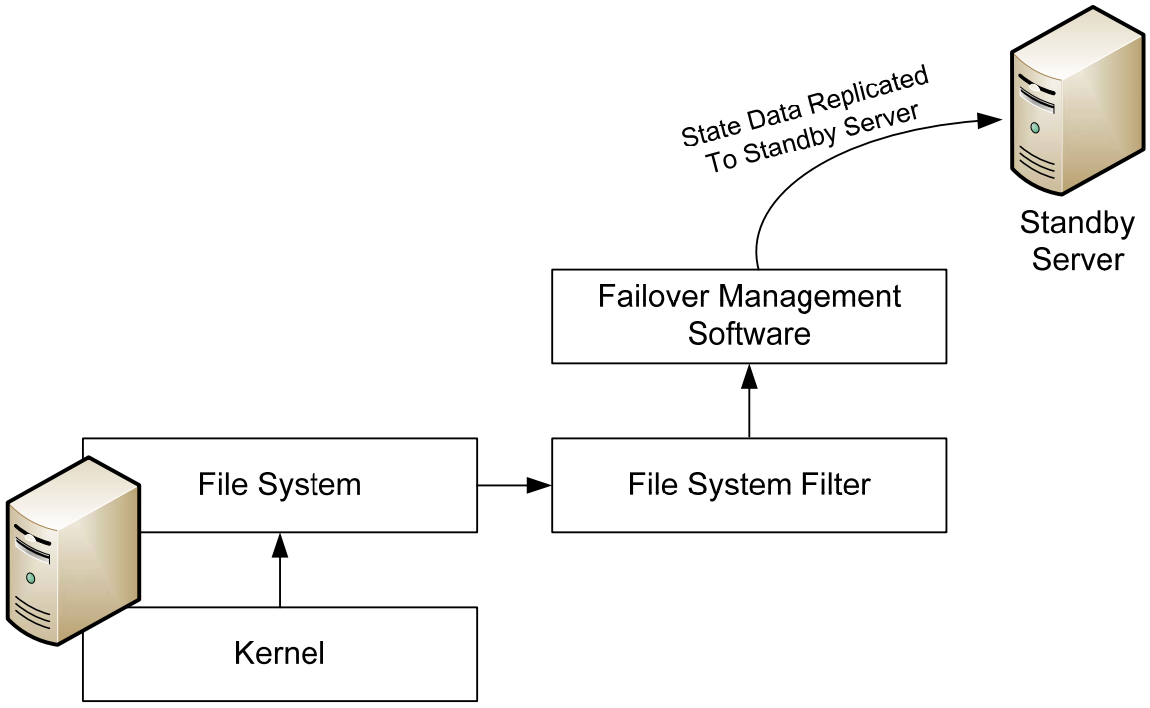

Figure 1: Discrete application changes are captured at a low level and replicated to a backup server.

This architecture is important because application failover technologies do not necessarily replicate the configuration and contents of an entire server from one location to another. Their focus is on the application itself. This application-centric focus is a critical differentiator because it enables an extremely fast failover to occur. Alternative solutions that replicate an entire server and its configuration can require additional time and effort to bring online after a service interruption.

Real-Time Data Replication

Occurring at the same time as the application's state replication is the need to replicate that application's data. For example, consider a server application such as email services with Microsoft Exchange. In order for the services provided by Microsoft Exchange to be available after an interruption, the configuration of the Exchange server itself as well as its mailbox contents must be present on the standby server. Successfully completing a rapid application failover requires that each change to the data on that Exchange server must also be replicated to the standby server.

There are obviously a number of factors that impact how well this process works. The failover application must be able to capture changes at an extremely low level, similar to what is shown in Figure 1. Appropriate network connectivity must be maintained between primary and standby data center sites. The application server must also be equipped with a small amount of spare processing capacity to support the offsite copying of deltas during normal operations.

Ensuring Users Maintain Access

Obviously, when any problem occurs, users must maintain access to the application for a failover technology to be valuable. Another component of a rapid failover solution is involved with the convergence of the network after a failover event occurs. This concept of convergence relates to the process whereby the network—and, thus, the servers and workstations—is made aware that a server's applications have been relocated to a new network location. Essentially, during a failover event, the network itself must recognize that the application has rehomed onto a new server.

Clients must also be capable of shifting to the new location. This shift must include both the network connectivity to the standby site and the redirected name resolution to that site. Many of these functions are incorporated through mechanisms outside the failover service itself but should be considered when looking at any failover solution set.

Failover/Failback Management

Effectively managing the actual failover is yet another component of a best-in-class solution. The process of determining an application's availability and making the decision to invoke a failover requires a number of rules and monitors. First, for an automated failover to occur, the failover service must have the built-in awareness to know when an application has actually ceased to function. This logic involves more than just recognizing that the server no longer responds to a network "ping" request.

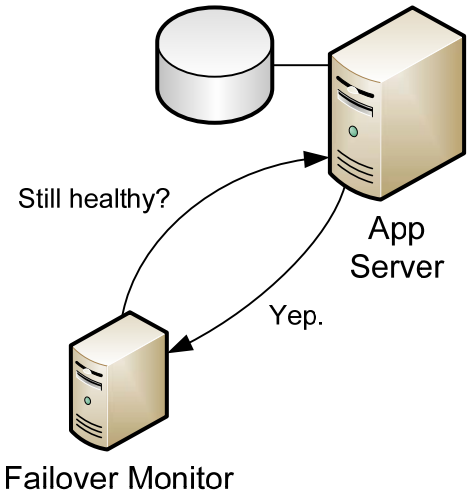

Figure 2: The job of a failover infrastructure's monitoring integrations is to watch for problem conditions and invoke a failover when necessary.

Effective failover solutions include built-in integrations into the applications they service. These integrations validate the ongoing health of the application with the goal of recognizing when that application's services are no longer available (see Figure 2). Yet the logic associated with determining a failover is necessary doesn't automatically end there. Policies and/or rules associated with failover conditions and behaviors should be built-in to the solution's management toolset. This inclusion enables administrators to identify when a failover is to occur or to manually invoke a failure in the case of a planned outage.

This rich management flexibility is critical because any failover event represents a non-trivial situation. Invoking a failover should only be accomplished when there is an expectation of extended downtime at the primary site. Such is the case because any failover of an application requires the precise orchestration of multiple events; a situation that can be challenging to reverse without the right toolsets and planning in place.

Remember that once a failover occurs, your users will continue to interact with that application as if it were in your primary data center. In its failover configuration, the state of and data within that application will incrementally change over time such that failing back the application requires a process of reverse replication. Any effective failover solution must also include the necessary toolsets to enable a successful reverse replication and failback when you are ready to return to normal operations.

Failover Technologies Protect Your Infrastructure

You can easily see how the positioning of a rapid application failover infrastructure in your environment will go far toward protecting your critical applications. With the right technologies in place, your applications and their data can remain available for users during periods of interruption and corruption or even in the event of data center disaster. The combination of the right solutions with a proper architecture will ensure a successful implementation.

Although this article has focused on the techniques associated with the failover product, one facet of this solution set that requires additional discussion relates to the failback process. The process of failback is complex and requires special consideration to be successful. The third and final article in this series will discuss the necessary steps and provide guidance on best practices for its proper configuration.