

# High Availability for Windows Services

*by Dustin Puryear*

Windows drives countless corporate networks around the globe. And because of its essential role in e-infrastructure, Windows, and Windows server applications in particular, assume an ever-increasing responsibility for driving the revenues—that is, the lifeblood—of the corporate world. It is no stretch to say that Windows high availability must be a fundamental element in your short- and long-term strategic IT planning.

This white paper discusses Windows high availability, with a focus on business drivers and benefits. Indeed, cost and real-world benefit, which are central to the white paper, are considered the two most important elements in choosing the right high availability solution. The paper first discusses current market solutions and technologies. Real-world challenges, including cost-benefit analyses, are discussed in detail. The white paper also discusses the major technical elements required in choosing a high availability solution, including the robustness of the technology, time-to-failover, and implementation difficulties. These issues are discussed with two core applications as examples, Microsoft SQL Server and Exchange Server. In conclusion, the white paper discusses Neverfail's high availability solutions, with an emphasis on how they can affect the reliability and resilience of your small, medium, and corporate networks.

## → Contents

<a href="#"><u>Business Drivers for High Availability</u></a> .....	1
<a href="#"><u>A Brief Introduction to High Availability</u></a> .....	1
<a href="#"><u>Microsoft Windows and Resilience</u></a> .....	2
<a href="#"><u>System</u></a> .....	3
<a href="#"><u>Network</u></a> .....	3
<a href="#"><u>Fault-Tolerant Systems</u></a> .....	3
<a href="#"><u>Clustering</u></a> .....	3
<a href="#"><u>Failover/Server Redundancy</u></a> .....	3
<a href="#"><u>Mission-Critical Windows Applications:</u></a>	
<a href="#"><u>SQL Server and Exchange Server</u></a> .....	4
<a href="#"><u>Log Shipping</u></a> .....	4
<a href="#"><u>Replication</u></a> .....	4
<a href="#"><u>Microsoft Cluster Service</u></a> .....	5
<a href="#"><u>The Neverfail Solution</u></a> .....	5

# High Availability for Windows Services

by *Dustin Puryear*

In today's world, Windows plays a crucial role in business e-infrastructure. For everything from the stock market to hospitals to military organizations, Windows servers run mission-critical applications. Because of the critical nature of these servers and the applications that they provide, it is vital that companies maintain a robust, reliable, and fully redundant Windows server infrastructure. This white paper addresses the concerns that most CTOs, CIOs, and managers face today: ensuring that Windows servers and applications remain available at all times.

## Business Drivers for High Availability

High availability is all about keeping mission-critical systems up and running at all times. Down systems are expensive, sometimes devastatingly so. Network World provides estimates of the cost of downtime (<http://www.nwfusion.com/careers/2004/0105man.html>) for operations such as Supply Chain Management at \$11,000 per minute for larger organizations. Depending on how heavily you rely on your network and Windows infrastructure, this cost may be even greater. You can break down the cost of downtime, and the business drivers for high availability, into three broad categories: revenue loss and growth, service level agreements, and business continuity.

### Revenue Loss and Growth

Downtime can have an immediate, direct, and harmful impact on revenue. As businesses become more reliant on automation, the cost of losing a key service (e.g., messaging or databases) increases in severity. The easiest way to determine revenue loss is to determine which revenue-generating activities rely on your Windows servers (e.g., via an "impact matrix"). Based on the amount of revenue per minute that your Windows servers support, you can then determine the immediate revenue loss if a Windows server becomes unavailable. (This does not consider the long-term cost of the loss of customer confidence.)

### Service Level Agreements

A Service Level Agreement (SLA) is a written agreement

between two parties (e.g., an IT department and the e-commerce division, or between one company and another) that offers a minimal level of service. Quite often, an SLA is used to ensure that core services, such as email and databases, are properly managed and available to clients. Typically, SLA's include a penalty clause that costs the provider money in the event of a failure. The more often a system becomes unavailable, the higher the penalty. SLA's can especially impact infrastructure providers (e.g., telephone and network companies) as well as the financial industry.

### Business Continuity

Business Continuity comprises the strategies and processes that a company takes to ensure that when a catastrophic event occurs (e.g., a hurricane destroys a corporation's headquarters), the business survives. Business Continuity places a heavy dependence on high availability, particularly over a geographically separated area. This allows a business to continue operating even if the primary place of business is destroyed. Based on the needs of Business Continuity alone, high availability solutions are usually cost-justified.

## A Brief Introduction to High Availability

Now that you understand the business case for high availability, let's briefly discuss the concept of "High Availability." High Availability (HA) has come to mean different things to different people, but the generally accepted definition for High Availability is the reduction or elimination of downtime for a service (e.g., a Web site).

HA is ultimately about ensuring that clients have access to the resources they need at all times. These clients may be other software components, servers across the Internet, or consumers purchasing books on Amazon.com. All of the clients depend on the availability of the service that your servers offer. When discussing HA, two terms are important: Reliability and Resilience.

### Reliability

For this white paper, reliability is defined as the identification of foreseeable and predictable problems within a server environment and the ability for proactive measures to be taken.

Achieving reliability requires complete system and configuration analysis and application of best practice configurations to achieve and maintain a sound environment.

In other words, reliability concerns itself with problems that you can avoid before actual trouble arises. For example, badly configured hardware will generate various problems, ranging from inadequate performance to corrupt data. Fortunately, with the use of best practices and expert knowledge, you can avoid these issues by properly configuring and verifying your hardware configuration. This also applies to software. If your server software (e.g., Microsoft SQL Server) is not properly configured, then problems can and will develop over time, even though any issues could have been avoided by ensuring that the software was properly configured and managed.

## Resilience

Resilience can be defined as the ability to recover from any unpredictable and unforeseeable failure. More specifically, resiliency affects the ability for your server to offer consistent and reliable service (i.e., availability) when needed.

You can measure availability by factoring in the amount of uptime vs. downtime a system has. Quite often, availability is discussed as a percentage of time that a system is up, more commonly known as the “nines.” For example, a system that is available for 99.9% of the year has 8.76 hours per year of downtime in which the system cannot service requests. The reason for downtime may include maintenance, failed components, loss of network or power, or even data corruption.

### Note:

Computing the “nines” is simple. With the knowledge that there are 8,760 hours in a year, the formula to compute your uptime is:

$$\text{Uptime \%} = (8760 - \text{hours down per year}) / 8760$$

So if up to one hour of downtime is allowed each month you have:

$$\text{Uptime \%} = (8760 - 12) / 8760$$

$$\text{Uptime \%} = 99.8\%$$

That is, one hour of downtime a month results in two “nines.” As you can see, achieving a high level of nines can be difficult.

Because resiliency addresses unforeseen problems (e.g., disk failure, loss of network in the LAN and WAN, application faults), it can be more difficult to ensure. Common solutions for ensuring availability of your services include replication, clustering, redundant hardware, and strong application monitoring.

## Microsoft Windows and Resilience

As one of the dominant operating systems in the business world (rivaling UNIX and other traditional “data center” operating systems), Windows is required to provide the resilience of other industrial-grade operating systems. When discussing Windows resilience, you can break down needs into five major categories:

- Application
- Data
- System
- Network
- Site

As shown in Figure 1, each of these categories plays a role in the larger Windows resilience world. All of the pieces must be in place for a resilient environment to exist.

Following is a discussion of these five categories, along with methodologies and technologies used to protect these pieces of the larger resilience puzzle.

### Application

Applications contain the actual application code that interacts with users and the data being manipulated on the server. Typically, an application such as Exchange becomes unavailable when the application stops running (e.g., it crashes) or the server hosting the application fails. Additionally, applications, even while running, may become unavailable if the network becomes unavailable. Protecting the application requires that both the application and the server hosting the application remain fully functional and available. Most often, this requires true clustering or a failover server (discussed later in the section “Technologies and Methodologies for Resilience”).

### Data

One of the most difficult tasks when providing resilience is ensuring that the data driving the application is available and consistent. Being “available and consistent” can mean different things, based on the application under discussion. For SQL Server, the data must be readable and writable by

the database server, and the data must be consistent with the expectations of the application. This can be difficult to ensure in many situations because the loss of an application such as a database management system may occur at unexpected times. If such a loss occurs during a backup, you have no guarantee that the backup is valid, meaning that the data is no longer consistent.

## System

Resilient solutions that provide for application and data availability can often fail if there is no system redundancy. A system may fail for a number of reasons (e.g., power loss), so ensuring that the application and data is available on a single server is quite often inadequate to provide for a fully available environment. This is the failure of high-availability systems that rely strictly on fault-tolerant systems, as discussed shortly.

## Network

Alas, an often-overlooked component of resilience is the network used to connect servers and clients. A resilient solution must also be capable of providing fail-over for all of the parts and pieces of the network, including switches, routers, and Wide Area Networks (WANs). Often overlooked, however, is the need for a resilient solution to ensure that if a system fails entirely, that a “failover” (which is discussed shortly) to a redundant system is seamless and does not require many, or preferably any, network changes.

## Disaster Recovery

Finally, the ultimate responsibility of resilient systems is to provide for availability when an entire site fails (e.g., when a building or campus housing servers is removed from the equation). Providing for resilience at this level requires the proper use of a WAN to connect clients to a failover system at another site. This problem is often the most difficult to solve because of the bandwidth and latency restrictions inherent to WANs, which places a high demand on the reliability and efficiency of the resilient solution in use.

### **Technologies and Methodologies for Resilience**

Having an understanding of what needs to be protected lets you properly

consider possible solutions. In Windows, three main types of solutions are available: fault-tolerant systems, clustering, and failover. Each of these solutions is discussed below, with a comparison of each in Table 1.

## Fault-Tolerant Systems

Fault-tolerant systems are typically high-end, single-server solutions that offer redundancy for all internal components, such as drives, power supplies, and even CPUs. Unfortunately, fault-tolerant systems tend to be very expensive, often stay far behind the performance curve, and still do not offer the full redundancy that a dual- or multi-server solution does. Fault-tolerant systems tend to solve some of the availability issues discussed earlier, but fail to fully address reliability or the full range of requirements of fully resilient services, such as server redundancy.

## Clustering

Clustering relies on two or more servers that share the load of an application. With most clusters, all of the servers are in use and the load is normally balanced between all of the servers.

A caveat of clustering solutions is that the application must be programmed in such a way that all instances can concurrently access, or block access to, shared data. For example, if an application is running on two servers and both instances try to update a single file on a file server then a collision occurs—the application software must be written to support this kind of operation. This rules out the use of clustering in many situations, and when you can use clustering, quite often it requires specialized software or modification to the original application. This form of clustering is often known as Active/Active clustering.

	Fault-Tolerant	Clustering	Failover
Reliability Focus	No	No	No
Availability Focus	Yes	Yes	Yes
Application Availability	No	Yes	Yes
Data Availability	Yes	Yes	Yes
System Availability	No	Yes	Yes
Supports WAN-level Redundancy	No	Yes	Yes
May Require Application Changes	No	Yes	No

**TABLE 1**  
Comparison of High Availability Solutions

## Failover/Server Redundancy

A failover solution, like a single fault-tolerant system, provides physical redundancy of components. Unlike a single fault-tolerant system, a failover solution can survive the complete failure of a server because at least two systems are available. Failover solutions also do not need to contend with concurrent access to shared resources because only one system is active at any time. While costs are not spread across as many active systems, as with Active/Active clustering, fault-tolerant solutions tend to be simpler to design and implement and require less specialized application software. Failover is a form of clustering known as Active/Passive clustering.

### A Common Problem: Shared Data

Availability solutions tend to group around two methods of sharing data between nodes: Shared All or Shared Nothing. Both solutions have costs and benefits. With Shared All, you can more easily offer an Active/Active cluster solution. However, Shared All solutions tend to be more expensive and require more modifications to the target application. Shared Nothing solutions are easier to implement, but come with the associated cost of only one server being able to access data at a time. Shared Nothing solutions also tend to be more reliable because data access is much less complex than in a Shared All scenario.

#### Note:

A Shared All solution means that every server may access any of the data at any time, whereas a Shared Nothing solution only allows one server to work with the data at a time.

## Business Continuity and Resilience

What is often overlooked when considering resilience for mission-critical systems is the need to geographically disperse clustered systems across the enterprise. It is now truer than ever that enterprise infrastructure, such as business and division headquarters, may be lost. Quite often the buildings that are lost provide the support for the entire IT infrastructure that drives a company. Even using an off-site data center still leaves all of an IT division's eggs in one basket. Because of the increased risks of having all of your mission-critical services in one location, it is important to use geographic diversity as part of your high availability and Disaster Recovery (DR) planning.

## Mission-Critical Windows Applications: SQL Server and Exchange Server

Two of the most common Windows applications that power

enterprises are SQL Server and Exchange. The caveats, costs, and benefits of resilient solutions for SQL Server and Exchange Server are examined below.

### Microsoft SQL Server Resilience

Microsoft SQL Server is an obvious target for high availability because it plays such a crucial role in driving enterprise applications. As a database management system, SQL Server is used to power a number of commercially available packages, as well as software developed in-house.

Traditionally, there have been three main methods of clustering with SQL Server 2000: log shipping, replication, and Microsoft Cluster Service.

## Log Shipping

With log shipping, SQL Server performs a backup of the transaction log and transfers that backup to a secondary SQL Server. The secondary SQL Server in turn restores the backup to its local database. In this way the secondary SQL Server has a relatively recent copy of the enterprise's databases.

Benefits of log shipping include:

- Relatively easy implementation
- Reliability
- Protection against the failure of servers
- Protection against the failure of storage devices

In addition, log shipping works well for geographically distributed servers given sufficiently large WAN bandwidth, which aids in Disaster Recovery (DR).

However, log shipping has quite a few problems, including:

- Lack of automatic failure detection
- Lack of automatic failover
- Relatively extensive downtime
- Requirement to update name/IP address of server manually
- Painful process to reset Primary server back to a production state

Depending on how busy your SQL Server is, there may also be lost data caused by the delay between when a change is made, a backup created, and the restore performed on the standby server. These problems will, in most cases, rule out the use of log shipping for a mission-critical database service.

## Replication

Both SQL Server 7 and SQL Server 2000 offer replication,

although SQL Server 2000 offers easier to manage and more reliable replication. Replication is a viable method of transporting data from a primary to a secondary SQL Server. Replication however has many of the same problems as log shipping, especially including:

- No automatic failure detection
- A requirement to manually failover a database server
- Painful process to reset Primary server back to a production state

This also means that downtime can be relatively long. In addition, transaction consistency is not guaranteed with SQL Server replication, meaning that there is a potential for data loss.

## Microsoft Cluster Service

Two of the major failings of both log shipping and replication are the lack of automatic failure detection or failover. The Microsoft Cluster Service (MSCS) solves these two problems. MSCS relies on shared storage between two nodes. As long as the primary node is active and running, MSCS uses it to service client requests. When the primary node fails (e.g., an application or system failure), then the secondary node takes over. MSCS is a failover solution, meaning that only one node can be active at a time (Active/Passive). With MSCS, failover is as quick as one minute.

Unfortunately, MSCS has a major drawback. MSCS requires the use of shared storage to house MSCS information and the information used by the application (SQL Server in this case). When the primary fails, the secondary begins using this shared storage to access the SQL Server databases. Because of this reliance on shared storage, MSCS clusters have a very limited geographical range, typically in meters. For DR scenarios, MSCS is insufficient. Additionally, the shared data storage required by MSCS introduces a Single Point of Failure.

In summary, the benefits of MSCS include:

- Automatic failure detection
- Automatic failover

Problems with MSCS include:

- Shared storage introduces Single Point of Failure (SPoF)
- Shared storage requires close physical proximity of nodes
- Inability to be used over a WAN for Disaster Recovery (DR)
- Can be expensive and complex to configure and manage

## Microsoft Exchange Resilience

For organizations wishing to provide resilient Microsoft Exchange services a common solution has been the use of MSCS. Microsoft Exchange 2000 and 2003 offer “clustering” as a service out-of-the-box, but the clustering offered is targeted for performance rather than resilience. This means that the loss of an Exchange server in a traditional Exchange cluster will result in the loss of service.

As with SQL Server, when using MSCS with Exchange, you must configure a primary and secondary Exchange server. Each server shares data storage, which is then used to exchange MSCS information and to store the Exchange databases. In the event of a failure of the primary Exchange server, the secondary comes online and assumes control.

The use of MSCS for Exchange availability, as with SQL Server, has some benefits:

- It supports automatic failure detection
- It supports automatic failover

Unfortunately, using MSCS with Exchange has the same drawbacks as MSCS with SQL Server, including:

- You are restricted by the physical limitation of the shared storage
- The cluster will fail if the shared storage fails
- Can be expensive and complex to configure and manage

## The Neverfail Solution

The Neverfail Group provides an “out of the box” Windows high availability solution designed to provide high availability

### Note:

Notably, Neverfail provides a proprietary performance monitoring, configuration verification, and what-if analysis application, named SCOPE, to validate and examine the reliability of target systems and applications. SCOPE is a key differentiator for Neverfail because the application allows systems administrator to easily determine if systems are correctly configured and performing adequately while they also use the broader Neverfail HA solution to provide redundancy for their mission-critical services, meaning that Neverfail provides a solution for both the reliability and resilience of your systems.

to any mission-critical application. Unlike most high availability solutions, the Neverfail Heartbeat software solution is positioned for any size organization, whether a small business or a global enterprise, and presents a simple and effective solution, whether over the LAN or the WAN, to protect against the cost and damage caused by unplanned application downtime.

Neverfail Heartbeat is the first software suite to take a holistic view and consider the real implications of downtime – the impact on the business, customers, and suppliers when end-users are unable to use applications from their desk. While backups, replication, and clustering address certain aspects and certain failures, the Neverfail approach looks at the whole application stack: the data, the system (server and operating system), the network, the site, and the application itself. Failures in any one component of the stack can lead to end-user experienced downtime and it is this single point of failure that Neverfail Heartbeat addresses.

Neverfail Heartbeat is the underlying technology engine that provides functionality and application high availability. Through an approach based upon using interconnected, redundant servers, the solution protects these applications against a range of commonly occurring faults, including application failures, operation system failures and site failures.

Neverfail reduces downtime by

- **Monitoring** the whole application environment to ensure all components are working.
- **Recovering** from errors in individual components without the need to failover to a secondary system.
- **Replicating** the whole application environment, including data, application, system and even registry information to ensure that there is no single point of failure in the solution.
- **Switchover** in the event of major problem, from a primary server to a secondary in a controlled fashion to ensure no data loss and minimum downtime.
- **Failover** in the event of a catastrophic failure, rapidly bringing the application up on a secondary server and getting end-users up and running again.
- **Switchback** after a Switchover or Failover, recovering back to the original production state requires no manual effort or system outage.
- **Neverfail** is based on an Active/Passive failover model discussed in the section “Technologies and Methodologies for Resilience.” As discussed earlier in the section “Microsoft

Windows and Resilience,” any high availability solution must address five key areas: application, data, system, network and site. As discussed below, Neverfail does just that.

### **Application-Level Protection**

In the previous discussion of Windows resilience, you learned about the lack of failure detection when using solutions such as SQL Server log shipping or Exchange clustering. While those technologies do provide a method for reproducing data, they do not provide actual failover capability. Neverfail failure detection is performed by application modules (e.g., Neverfail for SQL Server). When a failure condition is detected in the application, Neverfail will take any number of configurable, corrective actions. For instance, it could first attempt to restart the application (known in Neverfail as Recovering). If this process fails, Neverfail can then proceed to Switchover the service to the backup server. With Switchover, Neverfail performs an orderly “hand-off” of the application from the primary to the secondary server.

#### *Note:*

**A Neverfail switchover can also be initiated manually. This is useful when performing maintenance on a mission-critical service, or when performing a switchback (i.e., a switchover from the secondary back to the primary server).**

### **Data-Level Protection**

Data-level protection is one of the principle areas where most high availability systems fail. For example, MSCS relies on a shared storage disk—if this storage fails then so does the cluster. With Neverfail, this is never a problem because Neverfail provides data level protection by replicating all of the application data from the primary server to the secondary server. This data may include files, folders, and Registry settings. Neverfail does this by intercepting calls to the Windows logical I/O subsystem and replicating those calls to the secondary server. The actual changes are sent to the secondary server asynchronously in real-time, meaning the secondary server always has the identical set of data, as does the primary server.

One of the most important features of Neverfail is that because it replicates the data based on the logical I/O subsystem, you do not need to worry about configuring any application-level replication on your own. This eases installation and configuration, and also reduces the number of parameters that you need to monitor for your high availability cluster.

