# A Blueprint for Building Web Sites Using the Microsoft Windows Platform

*Draft, Version .9*

Microsoft Corporation

January 2000

**Summary:** This article shows architects and decision makers how to build complex Web sites using Microsoft technologies. (44 printed pages)

**Contents**



## Executive Summary

Businesses are rapidly moving to a standard Web-based computing model characterized by loosely connected tiers of replicated, task-focused systems. A large percentage of business Web sites-collections of servers, applications, and data that offer online services-are built using the Microsoft platform today as the basis for this computing model.

This article focuses on the use of Microsoft technologies to build the infrastructure for a scalable, available, secure, and manageable site in the most cost-effective and time-efficient way possible. It stresses keeping operations and application design of a Web site simple and flexible and emphasizes how a dot-com can successfully deploy and operate a site with the necessary effective scalability, availability, security, and manageability. There is less emphasis on the currently well-documented tools and development methodologies for building a Web application component. It also examines, from the macro level, the advantages of a Microsoft solution (using both Microsoft Windows NT® version 4.0 and/or Windows 2000 ), and drills down to define how to set up each tier of the site architecture using Microsoft products. Finally, it looks at Web site management using Microsoft tools and technologies.

Although intended as an overview, this article examines an example Web site that uses a successfully deployed architecture, which can serve as a model for sites built using the Windows platform. This document does not address (other than when relevant to scalability, availability, security, and manageability) topics such as application design, development tools, or database design; however, it does provide pointers to appropriate documents covering these areas.

The "Architecture Overview" section introduces a number of architectural concepts that are important for large Web sites. In the section "An Example Site," we describe a representative site and explain the infrastructure and various tiers used. The remaining sections discuss the four key attributes of a site-"Scalability," "Availability," "Security," and "Management and Operations"-and use the example site to illustrate these issues. References to relevant documents appear throughout the document.

# Architecture Overview

## Introduction

Large business sites are models of dynamic change: They usually start small and grow exponentially with demand. They grow both in the number of unique users supported, which can grow extremely quickly, and in the complexity and integration of user services offered. The business plans for many site startups are vetted by their investors for a believable 10-100x scalability projection. Successful business sites manage this growth and change by incrementally growing the number of servers that provide logical services to their clients either by the servers offering multiple instances of themselves (clones) or by partitioning the workload among themselves and by creating services that integrate with existing computer systems. This growth is built on a solid architecture foundation that supports high availability, a secure infrastructure, and a management infrastructure.

## Architectural Goals

The architecture described in this document strives to meet four goals:

- Linear *scalability*-continuous growth to meet user demand and business complexity.
- Continuous service *availability*-using redundancy and functional specialization to contain faults.
- *Security* of data and infrastructure-protecting data and infrastructure from malicious attacks or theft.
- Ease and completeness of *management*-ensuring that operations can match growth.

### Scalability

To scale, business Web sites split their architecture into two parts: front-end (client-accessible) systems and back-end systems where long-term persistent data are stored or where business-processing systems are located. Load-balancing systems are used to distribute the work across systems at each tier. Front-end systems generally do not hold long-term state. That is, the per-request context in a front-end system is usually temporary. This architecture scales the number of unique users supported by cloning or replicating front-end systems coupled with a stateless load-balancing system to spread the load across the available clones. We call the set of IIS servers in a clone set a W*eb cluster*. Partitioning the online content across multiple back-end systems allows it to scale as well. A stateful or content-sensitive load-balancing system then routes requests to the correct back-end systems. Business logic complexity is increased in a manageable way by functional specialization. Specific servers are dedicated to task-specific services, including integration with legacy or offline systems. Cloning and partitioning, along with functionally specialized services, enable these systems to have an exceptional degree of

scalability by growing each service independently.

### Availability

Front-end systems are made highly available as well as scalable through using multiple, cloned servers, all offering a single address to their clients. Load balancing is used to distribute load across the clones. Building failure detection into the load-balancing system increases service availability. A clone that is no longer offering a service can be automatically removed from the load-balance set while the remaining clones continue to offer the service. Back-end systems are more challenging to make highly available, primarily due to the data or state they maintain. They are made highly available by using failover clustering for each partition. Failover clustering assumes that an application can resume on another computer that has been given access to the failed systems disk subsystem. Partition failover occurs when the primary node supporting requests to the partition fails and requests to the partition automatically switch to a secondary node. The secondary node must have access to the same data storage, which should also be replicated, as the failed node. A replica can also increase the availability of a site by being available at a remote geographic location. Availability is also largely dependent on enterprise-level IT discipline, including change controls and rigorous test, quick upgrade, and fallback mechanisms.

### Security

Security-managing risks by providing adequate protections for the confidentiality, privacy, integrity, and availability of information-is essential to any business site success. A business site uses multiple security domains, where systems with different security needs are placed and each domain is protected by a network filter or firewall. The three principal domains, each separated by a firewall, are: the public network; a DMZ (derived from the military term, demilitarized zone), where front ends and content servers are placed; and a secure network, where content is created or staged and secure data is managed and stored.

### Management

Management and Operations broadly refers to the infrastructure, tools, and staff of administrators and technicians needed to maintain the health of a business site and its services. Many sites are located in what is often called a *hosted environment*. That is, the systems are collocated with an Internet Service Provider (ISP) or a specialist hosting service, where rich Internet connectivity is available. Consequently, the management and monitoring of the systems must be done remotely. In this architecture, we describe such a management network and the types of management functions the network must support.

## Architectural Elements

The key architectural elements of a business Web site highlighted in this section are client systems; load balanced, cloned front-end systems (that client systems access); load balanced, partitioned back-end systems (that front-end systems access for persistent storage); and three overarching architectural considerations: disaster tolerance, security domains, and management and operations.

### Elements of a large business Web site

Figure 1 captures the concepts and essential elements of a business Web site as described in more detail in the remainder of this section.

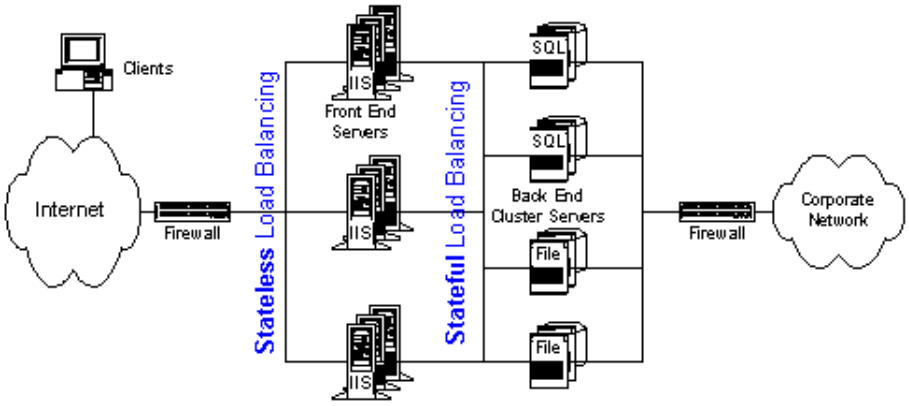


**Figure 1. Architectural elements**

Figure 1 shows the split between the front end and back end and the load-balancing layers as described in this document. Firewall and network segmentation are key security elements.

**Clients**

In this site architecture, clients issue requests to a service name, which represents the application being delivered to the client. The end user and the client software have no knowledge about the inner working of the system that delivers the service. The end user typically types the first URL, for example, **http://www.thebiz.com/**, and then either clicks on hyperlinks or completes forms on Web pages to navigate deeper into the site.

For a Web site with a very broad reach, an important decision is whether to support the lowest common set of features in browsers or whether to deliver different content to different browser versions. Currently, HTML 3.2 is usually the lowest version supported, although there are still older browsers in use. For example, browsers could be classified into those that support HTML 3.2, such as Microsoft Internet Explorer 3.0; those that support dynamic HTML (DHTML), such as Internet Explorer 4.0; and those that support Extensible Markup Language (XML), such as Internet Explorer 5.0. Different content would then be delivered to each. IIS and tools can create pages that can be dynamically rendered to different browsers.

**Front-end systems**

Front-end systems are the collection of servers that provide the core Web services, such as HTTP/HTTPS, LDAP, and FTP, to Web clients. Developers usually group these front-end systems into sets of identical systems called clones. They all run the same software and have access either through content replication or from a highly available file share to the same Web content, HTML files, ASPs, scripts, and so forth. By load-balancing requests across the set of clones, and by detecting the failure of a clone and removing it from the set of working clones, very high degrees of scalability and availability can be achieved.

### *Clones (stateless front ends)*

Cloning is an excellent way to add processing power, network bandwidth, and storage bandwidth to a Web site. Because each clone replicates the storage locally, all updates must be applied to all clones. However, coupled with load balancing, failure detection, and the elimination of client state, clones are an excellent way to both scale a site and increase its availability.

### *Stateless load balancing*

The load-balancing tier presents a single service name to clients and distributes the client load across multiple Web servers. This provides availability, scalability, and some degree of manageability for the set of servers. There are a variety of approaches to load balancing, including Round Robin Domain Name Service (RRDNS) and various network-based and host-based load-balancing technologies.

### *Maintaining client state*

It is not desirable to maintain client state in the cloned front-end systems because this works against transparent client failover and load balancing. There are two principal ways to maintain client state across sessions. One is to store client state in a partitioned back-end server. (Client state can be partitioned perfectly, and therefore it scales well. However, it is necessary to retrieve this state on each client request.) Another way to maintain client state across sessions is to use cookies and/or URLs. Cookies are small files managed by the client's Web browser. They are invaluable for minimizing load on stateful servers and maximizing the utility of stateless front ends. Data can also be stored in URLs and returned when the user clicks on the link on the displayed Web page.

### *Front-end availability*

As application code runs in these front-end servers, either written in a high-level language such as Microsoft Visual Basic® or C++ or written as a script, it is important to isolate programming errors from different Web applications. Running this application code out of process from the Web server is the best way to isolate programming errors from each other and avoid causing the Web server to fail.

## Back-end systems

Back-end systems are the data stores that maintain the application data or enable connectivity to other systems, which maintain data resources. Data can be stored in flat files, database systems such as Microsoft SQL Server?, or inside other applications as shown in the following table.

**Table 1. Different Types of Data Stores**

| | File systems | Databases | Other applications |
|---|---|---|---|
| Example | File shares | SQL | Ad insertion, SAP, Siebel |
| Data | HTML, images, executables, scripts, COM objects | Catalogs, customer information, logs, billing information, price lists | Inventory/stock, banner ads, accounting information |

Back-end systems are more challenging to scale and make highly available, primarily due to the data and state they must maintain. Once the scalability of a single system is reached, it is necessary to partition the data and use multiple servers. Continuous scalability is therefore achieved through data partitioning and a data-dependent routing layer or a stateful load-balancing system, which maps the logical data onto the correct physical partition.

For increased availability, a cluster-which typically consists of two nodes with access to common, replicated, or RAID (Redundant Array of Independent Disks) protected storage-supports each partition. When the service on one node fails, the other node takes over the partition and offers the service.

### *Partitions (stateful back-end systems)*

Partitions grow a service by replicating the hardware and software and by dividing the data among the nodes. Normally, data is partitioned by object, such as mailboxes, customer accounts, or product lines. In some applications partitioning is temporal, for example by day or quarter. It is also possible to distribute objects to partitions randomly. Tools are necessary to split and merge partitions, preferably online (without service interruption), as demands on the system change. Increasing the number of servers hosting the partitions increases the scalability of the service. However, the choice of the partitioning determines the access pattern and subsequent load. Even distribution of requests, avoiding hot spots (a single partition that receives a disproportionate number of requests), is an important part of designing the data partitioning. Sometimes this is difficult to avoid and a large multiprocessor system must host the partition. Partition failover, a situation in which services automatically switch to the secondary node (rolling back uncommitted transactions), provides continuous partition availability.

### *Stateful load balancing*

When data is partitioned across a number of data servers, or functionally specialized servers have been developed to process specific types of Web requests, software must be written to route the request to the appropriate data partition or specialized server. Typically, this application logic is run by the Web server. It is coded to know about the location of the relevant data and, based on the contents of the client request, client ID, or a client-supplied cookie, routes the request to the appropriate server where the data partition is. It also knows the location of any functionally specialized servers and sends the request to be processed there. This application software does stateful load balancing. It is called stateful because the decision on where to route the request is based on client state or state in the request.

### *Back-end service availability*

In addition to using failover and clustering to achieve high availability, an important consideration in the overall system architecture is the ability of a site to offer some limited degree of service, even in the face

of multiple service failures. For example, a user should always be able to log on to an online mail service, possibly by replication of the user credentials, and then send mail using cloned Simple Mail Transfer Protocol (SMTP) routers, even if the user's mail files are unavailable. Similarly, in a business site the user should be able to browse the catalog even if the ability to execute transactions is temporarily unavailable This requires the system architect to design services that degrade gracefully, preventing partial failures from appearing to be complete site failures to the end user.

**Disaster tolerance**

Some business Web sites require continuous service availability, even in the face of a disaster. Their global business depends on the service being available. Disasters can either be natural-earthquake, fire, or flood-or may result from malicious acts such as terrorism or an employee with a grudge. Disaster-tolerant systems require that replicas or partial replicas of a site be located sufficiently far away from the primary site so that the probability of more than one site being lost through a disaster is acceptably small. At the highest level, there are two types of replicated sites. Active sites share part of the workload. Passive sites do not come into service until after the disaster occurs. Where very quick failover is required, active sites are usually used. Passive sites may simply consist of rented servers and connectivity at a remote location where backup tapes that can be applied to these servers when necessary are stored. Even this minimal plan should be considered for any business.

The challenge is keeping replicated sites up-to-date with consistent content. The basic methodology here is to replicate the content from central staging servers to staging servers at the remote sites, which update the live content on each site. For read-only content this method is sufficient. However, for more sophisticated sites where transactions are executed, it is also necessary to keep the databases up to date. Database replication and log shipping is usually used where transactional updates to the database are shipped to a remote site. Typically, the databases will be several minutes out of synchronization. However, this is preferable to the complete loss of a site.

**Security domains**

Security mechanisms protect the privacy and confidentiality of sensitive information from unauthorized disclosure; they protect system and data integrity by preventing unauthorized modification or destruction; and they help ensure availability through prevention of denial-of-service attacks and by providing contingency or disaster planning.

Security domains are regions of consistent security, with well-defined and protected interfaces between them. Application of this concept can help to ensure the right levels of protection are applied in the right places. A complex system, such as an a large business site and its environment, can be partitioned into multiple security domains. *Region* can mean almost any desired division-for example, by geography, organization, physical network, or server or data types. For a business site, major divisions might reasonably correspond to the Internet, the site's DMZ, and the secure, corporate, and management networks. Domains may also be nested or overlapping. For example, credit card numbers within a database may require additional protection. Additional security controls, such as encrypting the card numbers, can provide this.

An analogy may help to visualize security domains. The Internet is like a medieval castle and its surroundings: Outside its walls, few rules apply and unscrupulous characters are in abundance. Corresponding to this castle, the key architectural element used to protect a Web site is to construct a

wall around it, with a main gate that is heavily guarded to keep out undesirables. The wall and any other gates need to be built to equivalent standards in order to maintain a given level of security. It certainly would not do to have an unprotected back door! For large business sites, this wall is known as the site's perimeter. In network terms, it means that the site's own communications facilities are private and isolated from the Internet, except at designated points of entry. The site's main gate is known as a firewall. It inspects every communications packet to ensure that only desirables are allowed in. Continuing the analogy, the stronghold in a castle holds the crown jewels. Additional walls and locked doors, or walls within walls, provide additional protection. Business sites similarly protect very sensitive data by providing an additional firewall and an internal network.

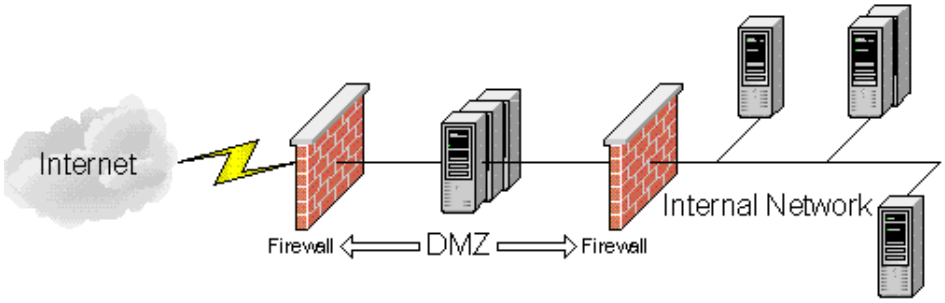


**Figure 2. Firewall/DMZ**

A firewall is a mechanism for controlling the flow of data between two parts of a network that are at different levels of trust. Firewalls can range from packet filters, which only allow traffic to flow between specific IP ports and/or ranges of IP addresses, to application-level firewalls, which actually examine the content of the data and decide whether it should flow or not. Sites often implement outward-facing firewalls that filter packets in conjunction with inward-facing firewalls that filter at the protocol and port layers.

Securing a site is complex; nevertheless the firewall/DMZ is a key architectural component. (It is actually a subset of network segmentation.) It is a necessary, but by no means sufficient, security mechanism to ensure a desired level of protection for a site. The "Security" section of this document is completely dedicated to securing a site.

**Management infrastructure**

A site management system is often built on a separate network to ensure high availability. Using a separate network for the management system also relieves the back-end network of the management traffic, which improves overall performance and response time. Sometimes, management and operations use the back-end network, however this is not recommended for large, highly available sites.

The core architectural components of a management system are management consoles, management servers, and management agents. All core components can scale independently of each other. Management consoles are the portals that allow the administrators to access and manipulate the managed systems. Management servers continuously monitor the managed systems, receive alarms and notifications, log events and performance data, and act as the first line of response to predefined events. Management agents are programs that perform primary management functions within the device in which they reside. Management agents and management servers communicate with each other using

standard or proprietary protocols.

Once systems reach a certain scale and rate of change, the management and operation of a Web site becomes the critical factor. Administrative simplicity, ease of configuration, ongoing health monitoring, and failure detection are perhaps more important than adding application features or new services. Therefore, the application architect must deeply understand the operational environment in which the application will be deployed and run. In addition, operations staff must understand the cloning and partitioning schemes, administrative tools, and security mechanisms in depth to maintain continuously available Internet-based services.

# An Example Site

## Introduction

This example site is intended to be generic so that the core architectural components and infrastructure can be illustrated. Nevertheless, it is representative of the key architectural features from many operational sites that we have reviewed. Owners are naturally reluctant to disclose actual details of their sites for competitive and security reasons.

Our example illustrates a large site and demonstrates both topological and component redundancy. It is a highly available system: Critical services can survive most failure modes, short of a major disaster. Servers in each of the ISP 1 through ISP N groupings support each of the site's critical functions, so even the loss of an ISP will not take the site down. Providing nonstop service through most disaster scenarios requires replication of the entire site in multiple geographies (geoplex). Cisco's Distributed Director is commonly used to support the geoplex. Unfortunately, site replication can more than double a site's cost and may introduce data consistency issues for Web applications.

Both smaller and much larger sites can be derived from this example. Smaller sites may not require as many servers in each cluster. Sites that do not need very high availability need only eliminate the redundant elements, notably the entire top half of the diagram, starting from Internet ISP 1. Sites that do not have high database security can eliminate the secure SQL clusters on the secure network. Much larger sites, on the other hand, can scale significantly by adding:

- Clones to each IIS Web cluster.
- The number of Web clusters.
- Internet connection access points.
- Front-end components, such as firewalls.

Moreover, as the network traffic and the number of managed devices increases, the management network must grow in both size and complexity.

Figure 3 shows the architecture of the example site.

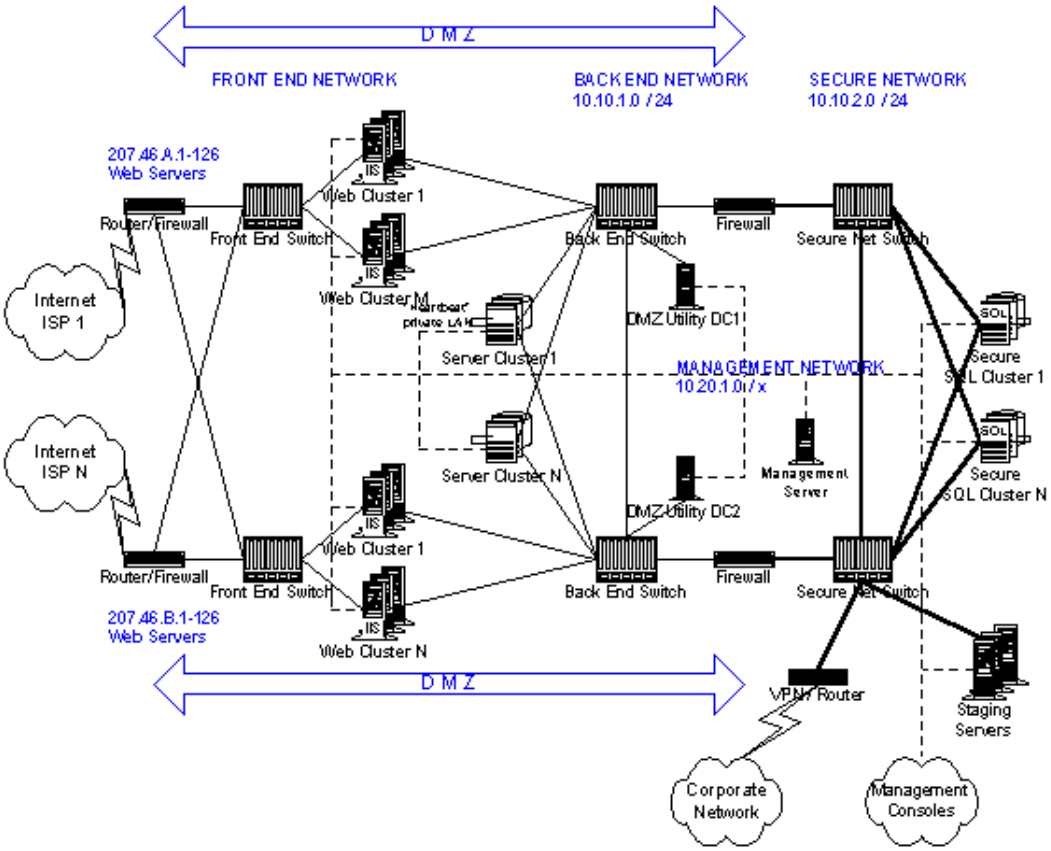


**Figure 3. Example of a large Web site network topology**

In Figure 3, different line styles, thickness, and annotations show the IP addresses and connections for different parts of the network. In particular:

- External (Internet-facing) network (thin).
- DMZ network (medium).
- Secure (internal) network (thick).
- Management network (thin dashed).
- Cluster heartbeat private network (thin-local to each cluster).
- Connections to the corporate network (lightning bolts).

The remainder of this section provides a tour of the example site, starting from the Internet and working through the DMZ and to the secure network, which includes the corporate network and management network.

## Internet

The tour begins with connections to one or more Internet Service Providers (ISPs). Our example illustrates multiple connections for redundancy, labeled ISP 1 through ISP N. These should be provisioned from diverse (physically separate) networks. Domain Name Servers (DNS-not shown in Figure 3) provide forward and reverse mapping between domain names and one or more TCP/IP addresses. For example, www.microsoft.com/ currently maps to the following addresses, each of which is a cluster:

207.46.130.14 207.46.131.28

207.46.130.149 207.46.131.30

207.46.130.150 207.46.131.137

In the event of multiple IP addresses, DNS will resolve successive queries for an IP address for www.microsoft.com by traversing the list-hence the name Round Robin DNS (RRDNS). A disadvantage of RRDNS is that it cannot detect loss of an ISP connection and it will continue to serve up the non-working IP address. This is not fatal, however, because the user need only request a reload of the Web page. Third-party solutions such as Cisco's *Local Director* or F5 Networks' *BigIP* provide more intelligent solutions that route connections dynamically.

## DMZ

Servers on front-end networks are exposed to the Internet. Firewalls are essential security components that provide network isolation by filtering traffic by packet type as well as source and destination addresses. They form one boundary of a DMZ (demilitarized zone) depicted by the double-ended arrows.

### Firewall

The first components in the path are Router/Firewalls, whose functions may be distinct or combined in single devices. Internet-facing routers support the Border Gateway Protocol (www.ietf.org/rfc/rfc1654.txt). High-speed front-end switches support connections to each server in the front-end Web clusters. The cross-connection with the Router/Firewalls provide an alternate path in the event of failure of the ISP connection or any of the components in the path.

### Front-end network

The front-end provides the core Web services, such as HTTP/HTTPS, using Microsoft Internet Information Server (IIS) to serve up HTML and ASP pages, and LDAP (Lightweight Directory Access Protocol) servers to authenticate customers. Site Server Commerce Edition may also be loaded on the front-end servers to provide additional database-driven services.

Front-end servers are grouped by service and function-for example, www.microsoft.com/, http://search.microsoft.com/, SMTP (email), or FTP (download). SSL service (HTTPS) is similarly segregated from normal HTTP traffic. This allows specially configured servers with expensive hardware security accelerator modules to support high-speed encryption functions. Further, SSL sessions are inherently stateful and may require special failover treatment.

Each of the Web clusters, running Windows 2000 in our example site, employs NLBS (Network Load Balancing Service-in Windows NT this is also known as Windows Load Balancing Service). Each clone is configured identically within each NLBS Web cluster delivering the same content. This provides transparent failover for stateless Web applications, radically improving service availability compared to individual servers. Web clusters support extensive scalability by adding clones to share the cluster's workload.

Client requests are made to each Web cluster using a virtual IP address that all of the front-end servers in the NLBS cluster can respond to. The front-end servers access the site's content data located on back-end clustered file share servers, and back-end clustered SQL servers.

All COM objects necessary to provide Web services, including objects called from ASP pages, are installed and registered on each front-end server. ASP pages for the site can either be loaded on the front-end servers' local disks, or kept on the back-end cluster file share servers.

Each front-end server is specially hardened for security and connects to three networks:

- Front-end network-Internet access.
- Back-end network-access to DMZ servers and, through inner firewalls, to the Secure Network.
- Management network-supports management and operations functions.

This network segregation improves security while increasing total available bandwidth and improving redundancy.

Note that the only publicly accessible IP addresses on any of the servers in this site are the NLBS virtual IP addresses, to which only the front-end servers can respond. IP filtering applied to Internet-facing NICs (Network Interface Cards) ensures that only the correct type and source of traffic for the functions supported can enter the front-end server. IP forwarding has also been disabled between these networks.

**Back-end network**

The back-end network supports all DMZ servers through use of a high-speed, private 10.10.1.*x* LAN. This architecture prevents direct network access from the Internet to the DMZ servers, even if the firewall were to be breached, because Internet routers are not permitted to forward designated ranges of IP addresses (see "Address Allocation for Private Internets," available at www.ietf.org/rfc/rfc1918.txt) including the 10.x.x.x range. As with the front-end network, redundant switches provide access to all front-end and back-end servers. All back-end switches share a common network, so back-end traffic loading can become problematic for active sites, especially if a separate management network is not available for logging and other front-end management network traffic.

The major components on the back-end network are security-hardened server clusters that provide services for storing Web content and temporary persistent state, such as intra-session transactional data (the contents of a shopping cart, for instance). Because all persistent data is available elsewhere, there is no need to provide backup facilities. Scalability is achieved by adding clusters, through partitioning the databases.

These servers employ Microsoft Cluster Service on Windows 2000, to achieve exceptionally high availability with failover capability. The failure of a server does not cause failure of the data services or even interruption in service. When the failed server is placed back online, it resumes delivering data services. Since hard drives can and do fail, the use of RAID drive arrays provides needed redundancy protection for data.

File shares within the cluster support file storage services. Microsoft SQL Server running on the cluster provides database services. Each cluster server employs at least four NICs: one for each switch, one for the private heartbeat LAN (which should use another private network address, for example 192.168.10.x), and one for the management LAN. In addition to server physical addresses, clusters have multiple virtual IP addresses to support the cluster itself and an address-pair (for redundancy) for each clustered service.

Hardened *DMZ Utility DC* servers support local domain accounts for all DMZ servers, local DHCP and name services (WINS or, preferably, DNS) and local utility file services. One-way trust relationship(s) to internal corporate domains provide authenticated access to secure internal systems.

## Secure Network

Another firewall forms the inner boundary for the DMZ and isolates what we term the *secure network* from the back-end network. The firewall is configured to only allow required communications between permitted port and source/destination pairs. The secure network again comprises a private network (10.10.2.0 in this example), a pair of coupled switches, a variety of servers, and a device labeled VPN/Router that provides connectivity to the internal corporate network. The secure network is logically part of the corporate network. Servers on the secure network are often members of an internal corporate domain, so domain controllers and address and name servers are assumed to be internal.

Other servers may be desired in this section to support additional functionality. There are many possibilities for processing and then moving transactional data between secure data stores and internal systems. Transactions range from traditional synchronous (MTS-Microsoft Transaction Service) to asynchronous (MSMQ-Microsoft Message Queue) to batch-based or e-mail-based store and forward. These are beyond the scope of this document.

However, it is important to note that the Internet, for many organizations, is but one delivery channel among many to provide customer services. As examples, consider a bookseller or a bank. Most business logic and processing take place internally on legacy systems. The Internet solution must interoperate with and serve these existing systems.

### Secure data stores

The secure SQL clusters are optional and are only required for more complex, transactional sites. They provide high availability, persistent storage for authentication databases, long-lived transaction storage, and keep customer information and account data confidential. Unlike server clusters in the DMZ, these servers must be backed up, either using directly connected removable storage devices or through the corporate network. They are otherwise similar to DMZ clusters. Each server again connects to both switches on the secure network for redundancy. Scalability is again achieved through partitioning the database and adding clusters.

### Staging servers

Staging servers appear in the secure network section, although they could be located in the corporate network or even in the DMZ. They receive and stage content from within the corporate network, or from an external content provider, and then deploy content to the Web servers so that the site is in a consistent state. A number of mechanisms are commonly used, including the Microsoft Content Replication System and tools such as RoboCopy.

### Corporate network connectivity

A device shown as a VPN/Router that connects the site to the corporate network is actually a router that, if required, may incorporate VPN security features to sign and encrypt traffic. Alternatively, using the Windows 2000 built-in IPSec features may add VPN functionality. This supports end-to-end security on an as-needed basis, while eliminating the cost of VPN hardware support.

For a site hosted in a corporate data center, connecting to the corporate network is very simple. In this case, the VPN/Router is connected directly to the corporate network.

Large business sites are frequently hosted remotely from corporate data centers. Dedicated lines generally connect the site to the corporate network, especially if high-performance, low-latency access is required. Alternatively, the Internet itself may be used for transport, in which case it is essential to secure all communications using VPN technology.

### Management network connectivity

We end our tour with a discussion of the management network, which provides the essential capability to monitor and manage the site. For simplicity, we show only computers with LAN connections to a separate management network. These are implemented with separate NICs. Some sites do not employ a separate management network. Instead, they collapse management traffic onto the back-end network. We do not recommend this practice for security, network loading, and management reasons.

Not shown are management connections to routers, switches, and firewalls. Also not shown are serial port dial-up connections, used for emergency out-of-band (OOB) access. Their absence does not imply they are not needed. When the management network (or the back-end network that is used for management) is unavailable, each host can still be accessed using this setup.

## Summary

The following sections use the model described in the previous example to discuss in detail how the architecture described in this document meets these four goals:

- Linear scalability-continuous growth to meet user demand and business complexity.
- Continuous service availability-using redundancy and functional specialization to contain faults.
- Security of data and infrastructure-protecting data and infrastructure from malicious attacks or theft.

- Ease and completeness of management-ensuring that operations can match growth.

# Scalability

## Introduction

Figure 4 illustrates two different dimensions of site scalability. The first dimension, the horizontal axis, represents scaling in terms of the number of unique clients that access the site on a typical day. As the number of unique clients goes up, so must the number of systems configured to support the growing client base. Typically, the content on the site required to support the client base must scale up as well.

The second dimension, the vertical axis, represents a measure of the business complexity of the site. We have identified three major categories. There are, of course, many variations in between. The categories typically build on each other with each category subsuming the functions of the category below. The lowest category, and the simplest in terms of business logic complexity, is the content provider category. The next category has transactions in addition to content but with much of the business processing done offline. While the top category has both content and transactions, much of the business processing logic is fully integrated with the online processing.

As sites move from left to right and bottom to top in the figure, the operational and application deployment challenges of the site increase significantly.

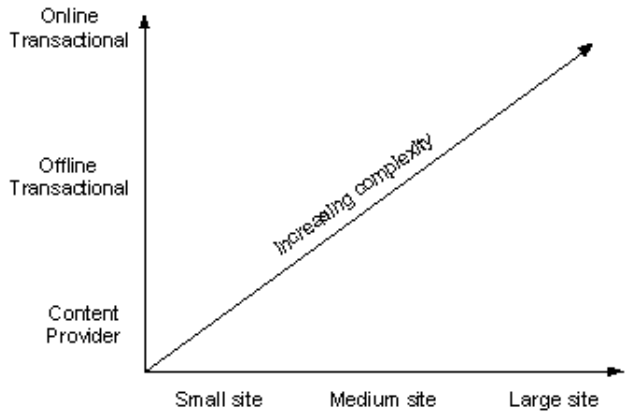


**Figure 4. Scaling dimensions**

In the rest of this section, we consider scalability in the context of Figure 4. First, we look at scaling the number of unique clients and content and then increasing the business complexity.

## Scaling Clients and Content

The two illustrations that follow, Figures 5 and 6, show how the number of front-end systems to serve the growing client base increases, as well as how to increase the number of partitioned data stores to scale the online content.

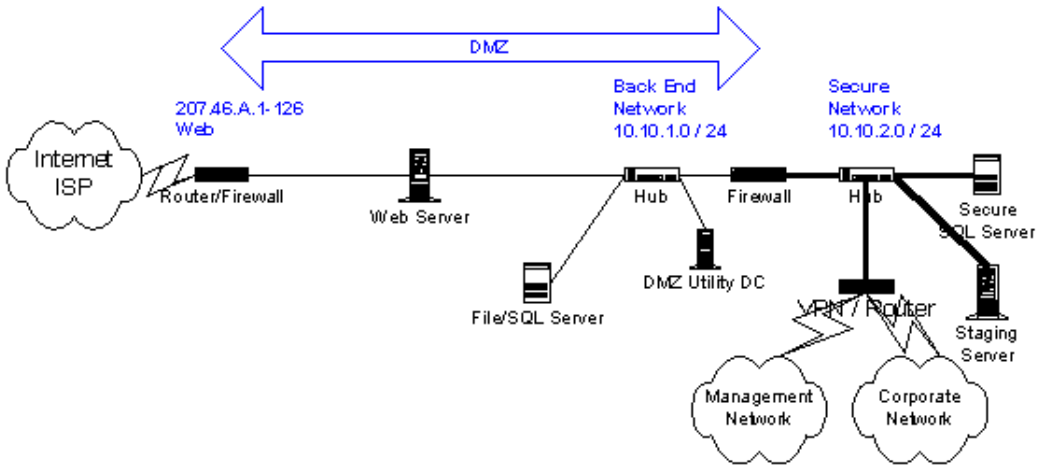


**Figure 5. A small site**

The preceding figure represents a basic site with one IIS Web server, one file or SQL Server, and one utility server in the DMZ with connections to a secure SQL Server or file server and a secure staging server. The following figure represents how a small site such as the one illustrated in Figure 5 could be scaled up to support more clients and more content.

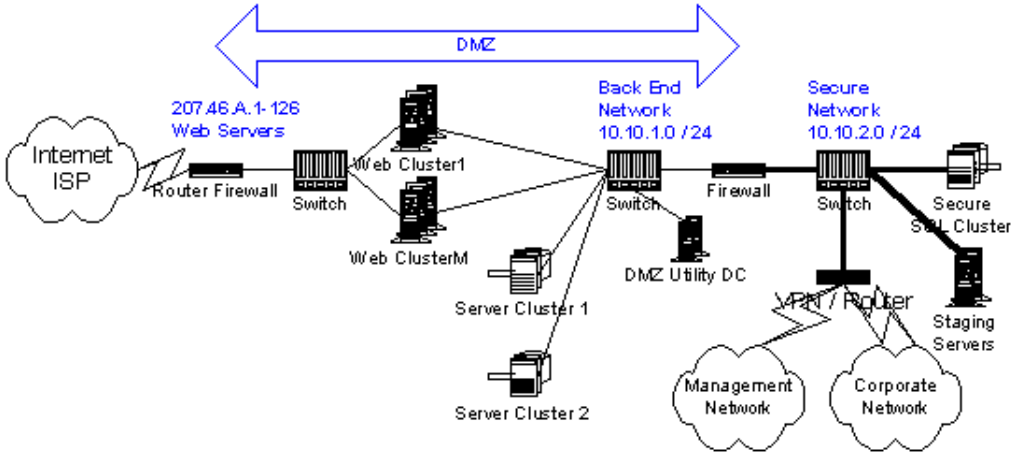


**Figure 6. A scaled-up site**

At the front end, the number of IIS Web servers and the number of Web clusters of these servers are increased and the load is balanced across them using NLBS. In the back end, the number of file and SQL Server clusters is increased, and therefore logic needs to be included in the front-end Web servers to route the data requests to the correct back-end partition of the data. We describe these two techniques next.

### Scaling the front-end systems

Increasing the number of cloned IIS Web servers, grouping them into Web clusters, and using a

load-balancing system are the principal techniques for increasing the number of unique clients supported. Note, however, that this involves important application state considerations, which we discuss later.

In addition to increasing the number of IIS Web servers, it is also important to optimize the Web application code that is executed in the Web servers. (This is beyond the scope of this document.)

***Web front-end load balancing for scalability***

Load balancing presents a single service name to clients in the form of a virtual IP address and then distributes the clients across a set of servers that implement the service.

There are three principal techniques for service load balancing:

- Round Robin DNS (RRDNS).
- Intelligent IP load balancing with a dedicated third-party outboard box.
- Intelligent IP load balancing within the servers, using NLBS in Windows 2000.

RRDNS is a method of configuring the Domain Name Servers (DNS) so that DNS lookups of a host name are sequentially distributed across a set of IP addresses, each of which can offer the same service. This gives a basic form of load balancing. The advantages to this method are that it is: free, simple to implement, and requires no server changes. The disadvantages are: there is no feedback mechanism about the load or availability of individual servers, and no fast way to remove a server from the set of available servers due to the propagation delays of DNS changes resulting in requests continuing to be sent to failed servers.

With server-based load balancing, a set of servers is grouped into an NLBS cluster and the load balancing is done by having each server in the cluster decide whether to process a request based on its source IP address. When a server in the cluster fails, the remaining members of the cluster regroup and the partitioning of the source IP address ranges is adjusted. The advantages of NLBS are its low cost (NLBS is part of the Windows 2000 operating system), no special purpose hardware or changes to the network infrastructure are required, and there is no single point of failure. The current limitations are that the servers do not dynamically adjust to load, and the regrouping is based on server failure, not application failure (although, third-party tools such as NetIQ and Microsoft's HTTPMon can be used to alleviate these limitations).

Combining RRDNS and NLBS results in a very scalable and available configuration. All of the nodes in an NLBS cluster must be on the same LAN subnet and all respond to the same IP address. It is possible to configure multiple NLBS clusters on different subnets and to configure DNS to sequentially distribute requests across the multiple NBLS clusters. This increases the scalability of NLBS and avoids the disadvantages of RRDNS, as there are multiple machines available to respond to each request sent to each NLBS cluster. Microsoft.com works in this way.

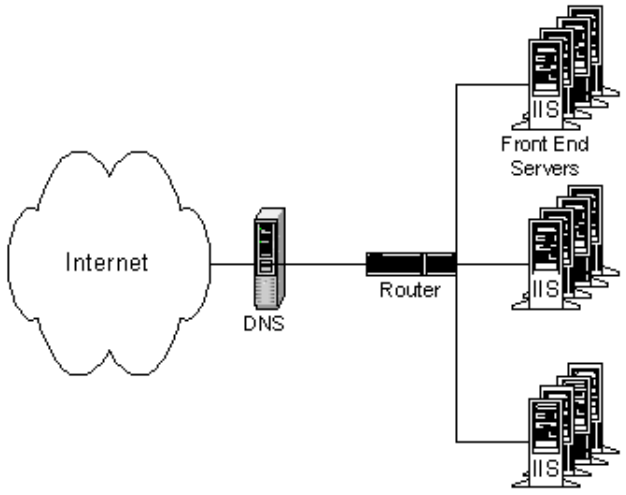


**Figure 7. RRDNS and NLBS: Three separate LAN segments, one domain name**

***Application state considerations***

To mask the failure of a server from the client, do not store application client state in an IIS Web server. Client requests cannot be dynamically load balanced. It is preferable to store client state in a data store and retrieve it if necessary on each client request based on either URL-encoded data or a client cookie. A client-cached cookie is also a very effective way to scale by storing per-client information in each client's system, passing the information to the Web server on each client request and using that data to personalize content or take some other client specific actions. RFC 2109 ("HTTP State Management Mechanism," available at www.ietf.org/rfc/rfc2109.txt) describes the HTTP cookie protocol.

However, some applications and some protocols require a persistent client-to-server connection. Using Secure Sockets Layer (SSL) to send encrypted data and authenticate the server is a prime example. Most IP load-balancing products support a mechanism that allows applications or protocols to maintain connections to the same server so that they function correctly, although without failure transparency.

**Scaling the back-end systems**

Adding more memory and more processors to a multiprocessor system can scale back-end systems. The Windows 2000 Advanced Server operating system has support for up to 8 CPUs and 8 gigabytes of memory. However, at some point that is no longer possible, or it becomes undesirable to have so much data dependent on the availability of a single system. At that point, it is necessary to scale the back-end systems by partitioning the data they serve or the logical services they provide. We call this *partitioning*. Unlike cloning, which is used to scale the front-end systems (where the hardware, software, and data are replicated), partitioning replicates the hardware and software but the data is divided among the nodes. Requests for a particular data object then need to be routed to the correct partition where that data is stored. This data-dependent routing needs to be implemented by application software running in the Web servers. This data-dependent routing layer can be thought of as stateful load balancing as opposed to the stateless load balancing used to scale work across the cloned front-end systems. Software also needs to be developed to manage the splitting and merging of partitions so that the load can be evenly spread across all of the partitions, thus avoiding any single partition becoming a hot spot.

The responsibility, however, is normally on the application architect to partition data into business objects that will be distributed evenly across an increasing number of servers as the size of the data and the workload increases. Fortunately, many site services are relatively easy to partition by object, as discussed earlier. However, the selection of the granularity of the objects to partition is difficult to change after site deployment, making it an extremely important upfront design decision.

Another method of scaling is to partition the services provided in the back-end systems into functionally specialized systems that offer services to their clients. This is often called an *n-tier* model. We discuss this in more detail in the section on scaling business complexity that follows.

**Scaling the network infrastructure**

As the traffic goes up to the site, from both the Internet and internal traffic within the DMZ and to the corporate network, the network infrastructure must also be enhanced. To support this increased demand, increase the bandwidth of links, upgrade hubs to switches, and install additional networks (for example, a dedicated management network to relieve the load on the back-end network).

## Scaling Business Complexity

The following diagrams illustrate that as the number of business processes integrated into the system increases, and the online nature of the business processes increase, the need for security and the number of systems grows. Maximum system capacity-whether the system design can grow smoothly and quickly with business growth-is usually the primary concern of any site.

The three models of site business complexity are: Content Provider, Offline Transactional, and Online Transactional.

**Content Provider**. In this model, no transactional access to internal systems is required. All Web services and content servers come from within the DMZ. All content is assembled on the staging server and then pushed, using replication, to the DMZ servers. This model is scaled, as described in the previous section, by adding Web clusters, adding clones to Web clusters, and adding back-end clustered servers.

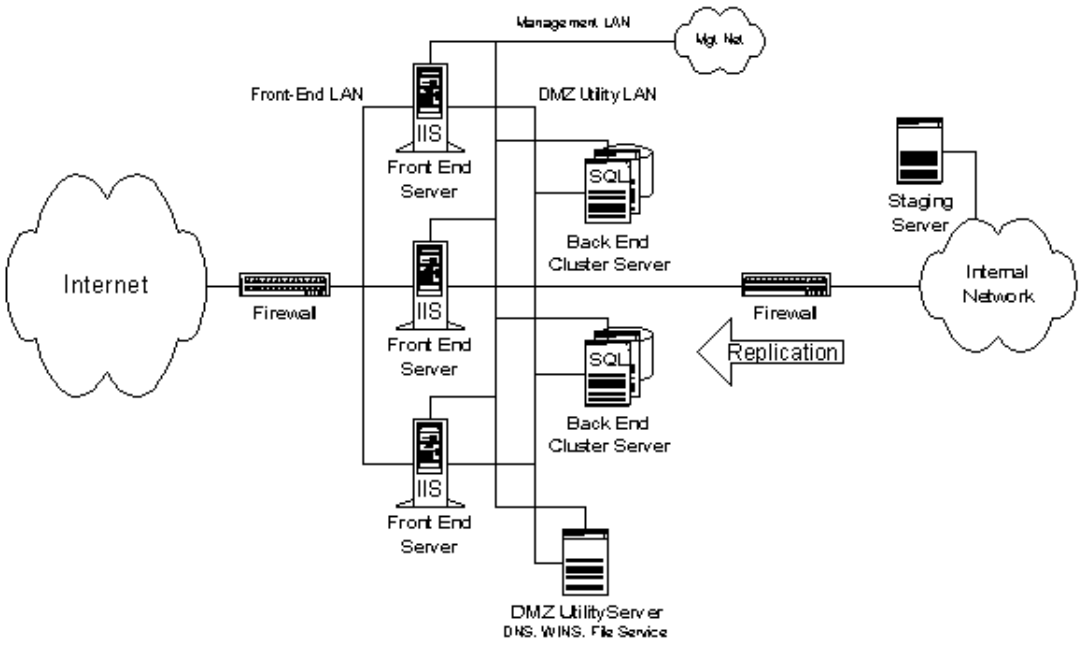


**Figure 8. Content Provider model**

**Offline Transactional**. This model is similar to the Content Provider model, with the addition of offline transactional access to existing business applications on the internal network. As with the Content Provider model, replication is used to update DMZ server content from the staging server. To support offline (non-real-time) transactions, asynchronous transfer of transaction data from the DMZ to the internal network is required. Microsoft Message Queue (MSMQ) service can be used to reliably deliver these offline transactions. In order to support legacy delivery channels, application systems and databases are implemented on the internal network. The device labeled "other delivery channels" represents traditional presentation devices, such as client workstations, Interactive Voice Response Units (IVRUs), or specialized input devices such as point-of-sale terminals or ATMs. This model increases in complexity as it scales the number of interactions with back-end servers in the internal network behind the inner firewall. MSMQ is useful for this type of interaction because it supports asynchronous communications as well as guaranteed delivery of messages. Batching requests together is also another successful technique for amortizing the costs of sending messages to the internal network.

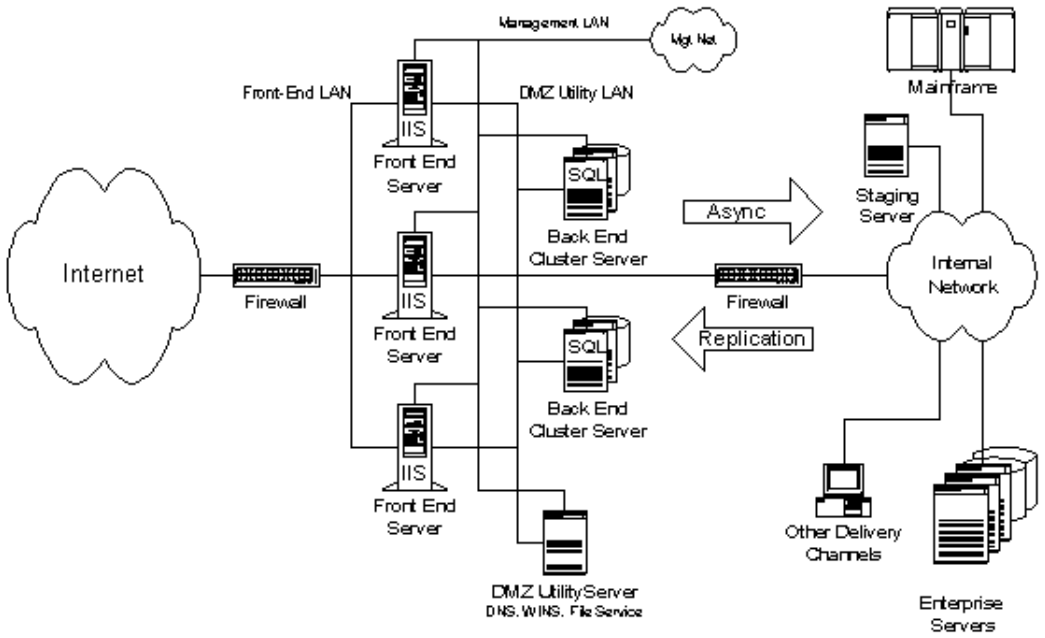


**Figure 9. Offline Transactional model**

**Online Transactional.** In this model, Web browsers have true online access to traditional applications that are resident on the internal network. Business application functionality is implemented on the internal network, which typically supports multiple delivery channels. Transactional communication from the DMZ to the internal network is achieved using standard synchronous communication mechanisms. The requirement to integrate with online business applications while the client is connected increases the complexity of this model considerably. This model has the most complexity, and scaling it is challenging because the interactions with the internal systems have to operate synchronously, or at least while the client is interacting with the online service. These types of interactions need to be carefully designed, and the number of them minimized. For example, a shopping basket should be built up using interactions only within the DMZ and only after the client requests and actual purchases should the internal systems be used.

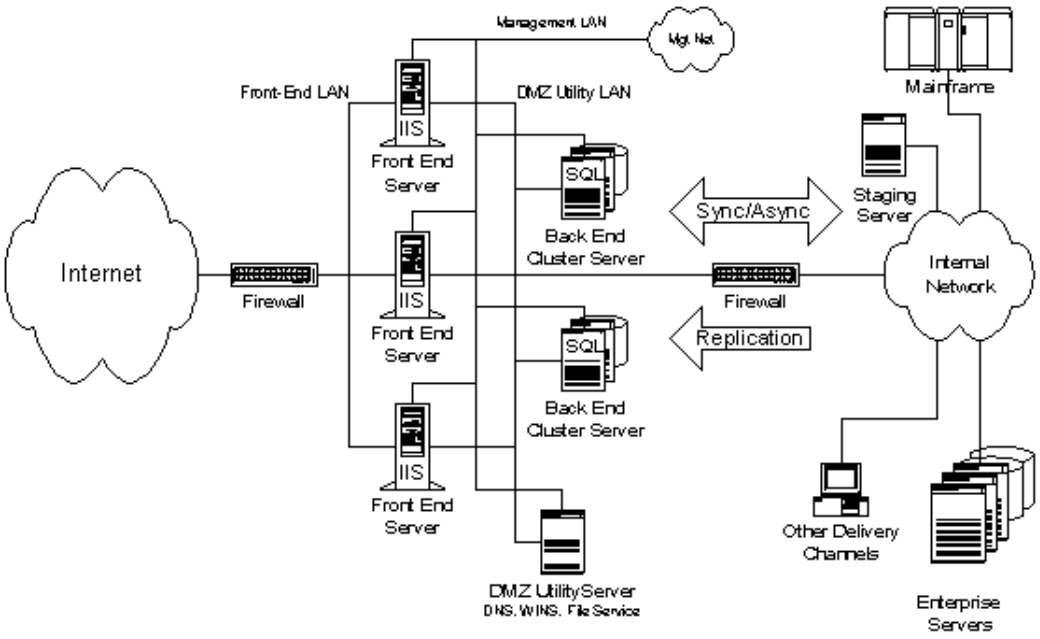


**Figure 10. Online Transactional model**

# Availability

## Introduction

The main technique for increasing the availability of an site is to add redundant components. These redundant components can be used to create multiple communications paths, multiple servers that offer the same service, and standby servers that take over in the event of a server failure.

Consider the two diagrams that follow. The first has some degree of high availability in both the front-end systems and the back-end systems. The second, however, has redundancy of all components and network links.

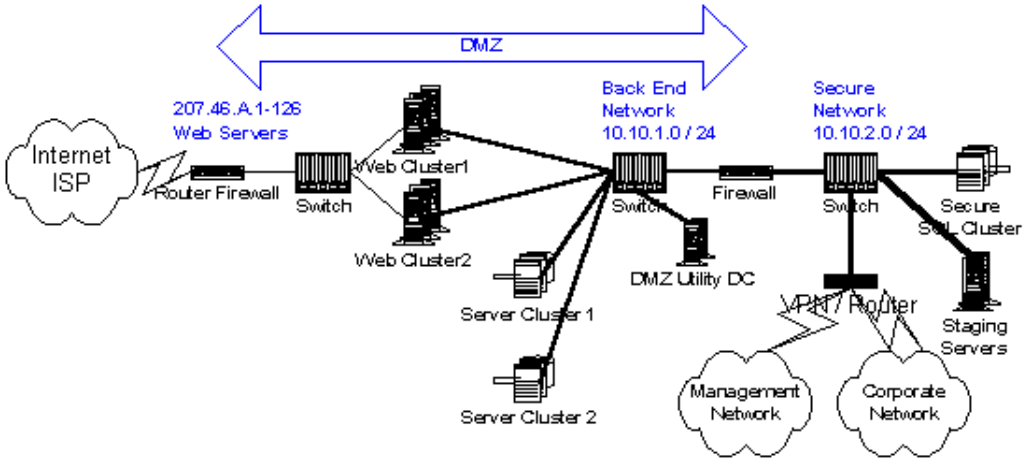

**Figure 11. A medium-sized site with some redundancy**

In Figure 11, we have two Web clusters, each with multiple servers, and we have two server clusters, each of which is configured as a failover cluster using Microsoft Cluster Services. We discuss these basic building blocks for increasing the availability of services in the following sections.

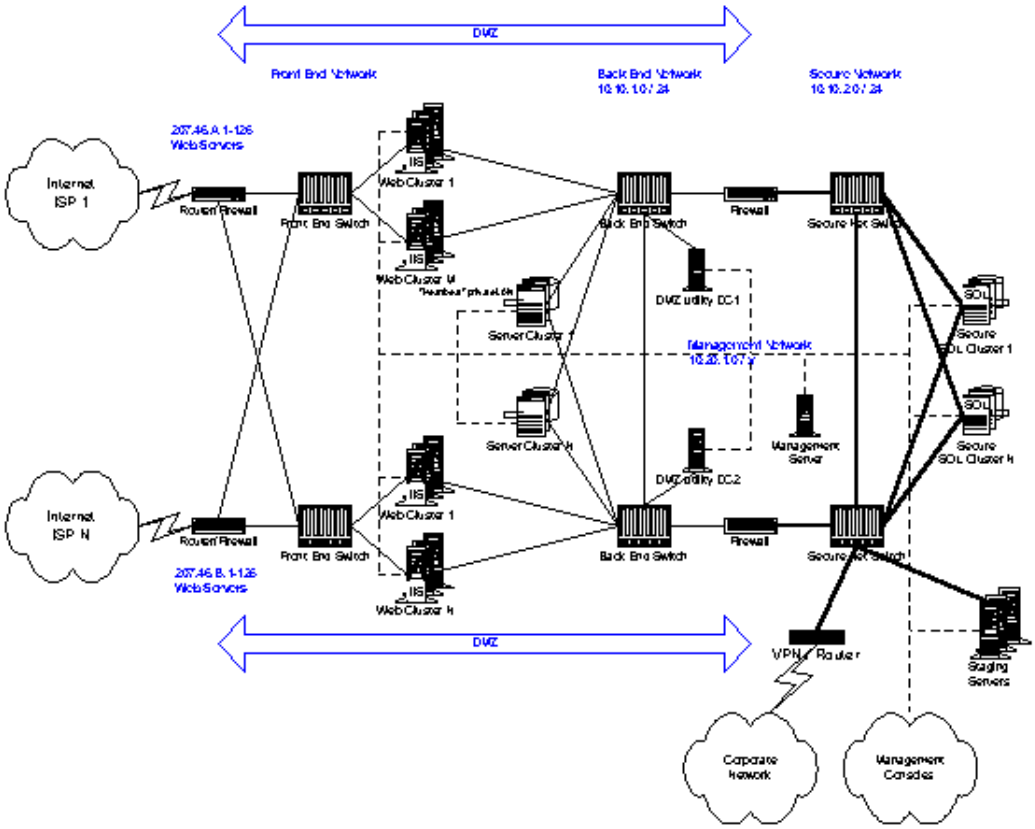


**Figure 12. A large site with complete redundancy**

In a large site with complete redundancy, not only are there multiple Web clusters, but each server is also configured as a failover cluster using Microsoft Cluster Services. In addition, there are connections to multiple ISPs and a separate management network.

## Availability of Front-End Systems

The cloning technique, described in section 4 on scalability, when coupled with NLBS load balancing and the use of stateless Web servers can be used to provide very highly available front-end Web servers. When multiple NLBS Web clusters are configured with Round Robin DNS, as described previously, it is possible to make the Web servers resilient to networking infrastructure failures as well.

The basic idea is exactly as described in the "Scalability" section, with the additional requirement that, when a clone fails or a Web server running on the clone stops responding, the load-balancing system must remove the clone from the Web cluster until it is repaired. NLBS automatically keeps track of the

operating members of the Web cluster and regroups when one fails. When the IIS Web server fails on Windows 2000, it restarts automatically. However, when the IIS Web server hangs, it must be detected with a monitoring tool. Microsoft's HTTPMon or a third-party tool such as NetIQ (www.netiq.com/) can be scripted to do this.

### Availability of the Networking Infrastructure

It is critical that the networking infrastructure and the connectivity of the site to the Internet are continuously available. As shown in the example site, the first important technique is to have multiple connections to the Internet using multiple ISPs. Connections should be diverse; that is, communication facilities should take physically separate paths from the provider to the customer's premises. This eliminates failure of the site due to a cut cable-not an uncommon occurrence.

For the highest availability, diverse power and redundant uninterruptible power sources should also be considered. Diversity in the infrastructure is often one of the major attractions of hosting a site at a facility dedicated to offering a hosting service collocated with multiple ISPs.

Within the site, switches and routers should be interconnected in such a way that there are always multiple paths to each service. Finally, a separate management network and an out-of-band network, as described in the "Management and Operations" section, are important for being able to manage performance and recover functions even in the face of various network infrastructure failures.

### Availability of Back-End Systems

Back-end systems can be made highly available by clustering them using Microsoft Cluster Services, a core technology that provides redundancy at the data layer and failover capability for services running on the cluster. Microsoft Cluster Services enables multiple SQL databases and file shares to share a RAID device, so if a primary file or database server fails, a backup comes online automatically to take its place. Like NLBS, no specific programming is required to take advantage of this system-level service

The data for both the database and the Web content needs to be further protected by being stored on a RAID disk array. In the event that a hard disk fails, the data will continue to be available, and a functioning hard disk can be hot-swapped into the array with no interruption in service.

The back-end servers send periodic messages, called *heartbeats*, to each other to detect failed applications or servers. The heartbeats are sent on a dedicated network (shown as the cluster heartbeat network), using NICs dedicated to this purpose. In the event that one server detects a heartbeat network communication failure, it requests verification of the cluster state. If the other server does not respond, it automatically transfers ownership of resources (such as disk drives and IP addresses) from a failed server to a surviving server. It then restarts the failed server's workload on the surviving server. If an individual application fails (but the server does not), Microsoft Cluster Services will typically try to restart the application on the same server. If that fails, Microsoft Cluster Services moves the application's resources and restarts them on the other server.

## Security

### Introduction

Security is about managing risks by providing adequate protections for the confidentiality, privacy, integrity, and availability of information. Security mechanisms and services, such as encryption, authentication, authorization, accountability, and administration, all support these objectives. Because protection mechanisms are never perfect, detection mechanisms (monitoring and auditing) generate alarms or other notifications to trigger reaction (corrective action) in the event of possible intrusions.

The security domain concept, presented in the "Architectural Overview," is invaluable toward ensuring consistent policy and the most cost-effective application of security controls. Within a domain, security is like a chain, whose strength is only as good as its weakest link. It is necessary to apply controls consistently across the entire domain, which may encompass network, platform, and application layers and includes all functions and components within the domain. Compensating controls are needed at boundaries with lower security domains to increase security to required levels.

The first step toward securing a site is to analyze business risks, the nature of systems and data to be protected, and the costs of applicable security mechanisms, and then determine what is optimal for the business.

In general, business sites merit higher levels of protection than browse-only informational sites. Many business sites include multiple functions with differing security needs. Applying the highest security protections to the whole site may not be necessary. By careful partitioning of these complex sites into security domains, it is possible to selectively implement the highest security protection mechanisms, which can be expensive.

Security policies and physical security procedures are essential aspects of effective security programs. Moreover, despite the inherent complexity of large sites and the additional complexity imposed by security controls, it is essential to implement the simplest possible user and management interfaces. Complexity invites misconfiguration and avoidance. Implement policy-based automation for security administration and configuration wherever feasible. Because our emphasis in this document is on security architecture and technology, we will not discuss these issues further. However, it is important to note here that effective security is very much a people and processes issue. The best technologies are ineffective if people are unaware of, or indifferent to, security needs.

In the remainder of this section, we discuss various protection mechanisms, including network and platform protections. (Application protections are outside the scope of this document.) We next consider client authentication and authorization approaches required for complex Web applications.

## Network Protection

### DMZ structure

The example site illustrates the use of firewalls and a DMZ. The DMZ is an essential architectural element that provides multiple layers of protection between the Internet and internal system resources. It comprises:

- An Internet-facing firewall that filters Internet traffic and separates it from network traffic within the DMZ.

- Special-function, high-security (hardened) components within the DMZ to support required

services, such as Web or e-mail services.

- An internally facing firewall that separates DMZ traffic from secure, internal networks while providing controlled access to a limited number of hardened systems and services within those networks.

Internet-facing firewalls are, in practice, universally employed (although often in the form of security routers). All Web sites that permit network connection to corporate or other secure networks should also employ internal firewalls to provide isolation of the DMZ from the internal network.

A DMZ is necessary, but in itself not sufficient, to protect the site. Any component within the DMZ, if penetrated, can be used to attack the entire site. Sensitive customer and account information and authentication/authorization databases should not reside within the DMZ. Instead, protect these within the secure, internal network. Performance considerations may force the need to replicate sensitive data to the DMZ. Where this is necessary, enhance data security using alternative mechanisms discussed in the "Platform Protection" section.

**Firewall types**

Firewalls generally function at network protocol layers and serve to exclude all network traffic except for permitted source/destination IP addresses and ports (protocols, such as HTTP or SMTP).

In its simplest form, a firewall can be built from a network router that has been configured with appropriate Access Control Lists (ACLs). Such security routers are, in fact, frequently used as firewalls. They are able to filter unwanted traffic (based on protocol type and source and destination addresses) and protect the DMZ from some-but not all-denial-of-service attacks. Some sites employ security routers for performance reasons, because firewalls that are more complex are unable to support the required throughput (sometimes in the gigabit-per-second range). Low-risk sites may also choose to deploy security routers for cost reasons.

Packet-screening or stateful firewalls provide complete network-layer isolation by maintaining communications state. They are able to detect known denial-of-service attacks and provide additional security features, such as network address translation (NAT, which completely hides internal devices), and active FTP (for which the data transmission port is selected dynamically). Firewalls commonly used include Cisco’s PIX or Check Point’s Firewall-1. These devices are now able to support network throughput in excess of 150 megabits per second.

Firewalls are not a cure-all, however. Because they generally function at the network level, they are not able to protect against attacks launched at higher protocol layers. For example, an application or Web server inside the DMZ that does not correctly check incoming strings is vulnerable to buffer overflow attacks. This could cause the service to crash or, worse, could allow a cracker to take control of the component. This exploit is, unfortunately, more common than it should be.

**Firewall configuration**

Internet-facing firewalls should permit access to only those services that are required to support Web site business functions, typically HTTP and LDAP, and less-frequently FTP and SMTP mail. Virtual Private Network (VPN) support for limited business-to-business or other remote access may also be

required. Review very carefully and resist opening access to all other ports unless a strong business need exists for doing so. Off-the-shelf platforms used to construct firewalls (for example, Windows 2000 and Checkpoint's *Firewall-1*) must be rigorously hardened.

Internally facing firewalls should similarly restrict traffic, based on only those protocols and services necessary to support access to internal data and system resources and, conversely, required for supporting and managing the DMZ components. Internal firewalls are essential to protect critical information resources on the internal network, in the event that a cracker does manage to penetrate the DMZ. The number and variety of ports and addresses required to traverse the inner firewall is frequently far greater than for the outer firewall, especially if management functions are supported through the DMZ's back-end network. These are prime targets for a cracker to gain access to the internal network. It is essential to restrict access to very limited sets of necessary ports and target hosts, to ensure that a compromised system on the DMZ has only limited opportunity to support attacks on internal systems.

**Network segregation**

Data networks internal to the DMZ should be segregated for security, as well as to add bandwidth. Generally, each computer is equipped with two or more Network Interface Cards (NICs). The example site section illustrates network segmentation and discusses this at some length. The key principles are:

- Segregate different types of Internet traffic to different Web clusters-for example, HTTP, HTTPS, and FTP. Each cluster can then be configured to reject traffic other than the type it is designed to service.

- Segregate Internet traffic from back-end traffic. This prevents direct access from the Internet to the internal network, and permits filters to be configured for each NIC, thereby limiting traffic to only types appropriate for the server.

- Use non-routable network addresses for internal Web site networks, as described in the example site.

- Implement a management network in order to segregate management from all other traffic. This also permits configuration of NIC filters to restrict only traffic appropriate to that NIC. Powerful management functions should then be restricted to the management network and unable to traverse the service networks. It also eliminates management traffic from passing through firewalls, which further reduces vulnerabilities. Securing the management LAN itself is of crucial importance.

**HTTPS-SSL for encryption**

Sending data over the Internet is like sending a postcard through the mail: Anyone along the path can intercept it. Therefore, a standard, secure communications channel is important for Web sites and other applications that transfer sensitive customer information across public networks. Secure Sockets Layer (SSL), in conjunction with HTTPS protocol and server certificates, provides the needed encryption functions and further supports Web server authentication. It is important to understand that SSL *only* protects in-flight communications and is not a replacement for other site security mechanisms.

Server authentication is transparent to the client, because most Web browsers in use today automatically validate certificates issued from all the major Certification Authorities. It is clearly important for clients

to know they are communicating with the correct Web site and not with someone pretending to be that site.

SSL encryption provides confidentiality and integrity for transmitted data, which is especially important for protecting user passwords and credit card information. Due to export restrictions imposed by the U.S. Commerce Department (DOC), there are currently two versions of encryption. The stronger, which uses 128-bit keys, can freely be used within the United States and internationally for designated industries, such as banking and health care. For all other uses, encryption software exports are limited to 56 bits.

SSL does pose some problems. First, SSL is stateful. State must be maintained for the setup and duration of a secure session, which imposes the requirement for session stickiness on front-end load-balancing systems. Second, encryption/decryption is computationally intensive for Web servers, which may not have enough processor cycles to support these functions in software. Hardware accelerators are available to offload servers, but they are costly and typically deployed on only a limited set of front-end servers. In other words, HTTPS traffic is usually segregated from HTTP and supported by specialized front-end servers. For both these reasons, SSL use should be limited to only those communications that really need this level of protection.

**Intrusion detection**

Intrusion detection systems (IDS), such as Cisco’s NetRanger or ISS’s RealSecure, provide real-time monitoring of network traffic. An IDS can detect a wide range of hostile attack signatures (patterns), generate alarms to alert operations staff and, in some cases, cause routers to terminate communications from hostile sources.

Deployment of some form of intrusion detection is essential in high-security environments, consistent with the "prevent, detect, and react" approach to security discussed in the "Security, Introduction" section. IDS sensors should be placed on every distinct network, even in front of a firewall or border router. These sensors communicate with management consoles, as described in the "Management and Operations" section later in this document.

Unfortunately, there are several reasons why IDS are still not in widespread use:

- Performance-real-time monitoring of very high-performance networks is still not feasible.
- False accepts, false rejects-the ability of IDS to differentiate attacks from normal network traffic is improving, but still far from adequate. IDS managers are awash in logs with poor signal-to-noise ratios.
- Cost-IDS is expensive to implement and operate.

There are alternative techniques for intrusion detection-for example, routing telnet port traffic to a special trap or sacrificial server. Use of third-party server log analysis tools can provide information about intrusions, although possibly too late to prevent them. Cisco provides a NetFlows feature on some of its routers that can be used to detect network intrusions, but this is not currently well supported with analysis tools. The state of the art in intrusion detection is unfortunately not well developed.

## Platform Protection

### Hardening components

Hardening is another essential practice that protects individual server operating systems. All DMZ and internal systems that they communicate with require hardening. This includes carefully restricting and configuring access privileges to all resources (for example, files and registry entries) using ACLs, and eliminating all protocols, services, and utilities not required to support the business functions and management of the computer. Careful attention must be paid to security and audit settings.

TCP/IP protocol stacks should employ filtering, where feasible. The IPSec (IP secure) implementation in Windows 2000 has many sophisticated filtering policies, even if its integrity and encryption capabilities are not used. Selective lock-down of ports by IP address/subnet is possible without requiring reboots. All filtering happens at a low level, so services such as IIS never even see the packets.

The Security Configuration Editor (SCE), introduced with Service Pack 4 for Windows NT 4.0, is an important new tool for implementing consistent, policy-based controls. Windows 2000 adds Group Policy Editor (GPE), which extends this capability (and much more) to the domain, to active directory organization units, and even to groups of computers. Most operating system security configuration settings can now be defined in sets of policy templates. These templates can be constructed for each class of machine and implemented systematically for all computers in a site. Because these templates should enforce tight lock-down of production computers, it may be desirable to construct alternate sets for maintenance and diagnostic use. A related analysis feature permits a server's current security configuration to be analyzed and verified against policy, a valuable way to verify continued compliance with policy.

Third-party network and system security scanning tools are another important aid to ensure effective security configuration for site servers. Well-known products from vendors like Internet Security Systems (ISS-www.iss.net/) and Network Associates (www.nai.com/) include wide-ranging attack scenarios to provide network and system vulnerability assessments.

It is essential to monitor security alerts, such as from CERT (www.cert.org/), in order to determine the latest cracker exploits and patches needed to keep security protection current. Microsoft provides e-mail security bulletins for its products. (Administrators can sign up for this service at www.microsoft.com/security/.)

Key service components, such as domain controllers, Domain Name Service, Internet Information Server, and Microsoft SQL Server all have specific additional requirements. An excellent and very complete security checklist for configuring Windows NT 4.0 / IIS 4.0 Servers is available at www.microsoft.com/security/. Many of the Windows NT configuration items are equally applicable to Windows 2000 and other DMZ hosts.

### Monitoring

Platforms must be monitored (audited) periodically in order to ensure that configurations and policies do not drift from initial, secure configurations. Several logs and tools provide this capability, including Windows 2000 event logs, IIS logs, Security Configuration and Analysis, and sysdiff (NT Resource Kit). Windows 2000 employs code signing and System File Checker to verify the integrity of important

system modules. A number of third-party tools support integrity checking, including anti-virus scanners from various sources and Tripwire's tripwire (www.tripwiresecurity.com/), which verifies selected files and registry settings.

It is also essential to audit administrator, group and service accounts periodically, to ensure that access privileges are available to only authorized personnel in accordance with site policies. For sites that have a relatively small number of accounts, this is a straightforward manual process.

**Windows domain structure**

Site farms consist of a few classes of servers, each of which may contain large numbers of devices. A very large site may contain a thousand or more servers. A single back-end network usually supports all servers in the DMZ. Because the number of administrative and service accounts required to support the site is small, a single-domain structure is adequate to manage all DMZ server accounts. One-way trust relationships with internal domains may be needed to support authentication to the secure network and internal servers and their databases. Windows 2000 offers flexible integration of site accounts with the enterprise's Active Directory, while supporting high-security needs of the site.

**Securing site data**

Security mechanisms that protect the integrity and confidentiality of data include network and system access controls, encryption, and audit or monitoring facilities.

It is first necessary to categorize the kinds of data available in the site, such as programs and HTML code; customer information, including passwords or other authentication/authorization information; advertising; product catalogs; and other content. Understanding the effect of unauthorized disclosure (confidentiality/privacy) and unauthorized destruction, or modification (integrity) should be determined for each type. For example, most static HTML pages are public and do not require any protection from disclosure. On the other hand, vandalism of these pages could seriously undermine consumer confidence in the site.

Once the characteristics of the data are understood, the associated risks have been assessed, and the costs of protective controls determined, sound business judgment should determine what protections are needed.

One of the most costly and important decisions that must be made is whether to duplicate site systems and databases geographically, or adopt alternative contingency plans. Natural disasters, fires, terrorist attacks, and major network disruption may be very unlikely, but have the potential to put a site out of business.

Carefully planned, implemented, and maintained DMZs can be fairly secure. Nevertheless, highly sensitive data often merits additional protection. Several approaches are commonly used:

- Sensitive data should reside inside the inner firewall (for example, on the *Secure Network* for the example site). Because some access paths must be opened through the firewall to enable legitimate access to the data, this solution is not a cure-all and may reduce performance to unacceptable levels.

- Databases often contain mostly low-sensitivity data, but with some data that must be protected, such as user credit card numbers. Such data, if located in the DMZ, should be encrypted in the database and decrypted for use as required.

- Passwords are only stored after being transformed using one-way algorithms; they are never stored in the clear.

- Directories used for customer authentication require special attention, because penetration of this subsystem exposes all customer data.

Microsoft's SQL Server 7.0 relational database enforces the ANSI/ISO SQL standard, which specifies that users cannot view or modify data until the owner of the data grants them permission. In order to ensure strong and secure authentication, it is important to run SQL Server in Windows NT Authentication mode. (We do not recommend SQL Server Authentication mode, because system administrators define user login accounts with passwords that are transmitted in the clear.) Securing SQL Server is generally outside the scope of this document. For further information, see the SQL Server 7.0 Books Online, which are available on MSDN.

## Client (Member) Access Control

Client access controls include authentication mechanisms, which verify the client's identity, and authorization mechanisms, which dictate which resources the authenticated client can access.

Authentication for large sites can be as simple as presentation of a cookie by an anonymous client's browser. (Client cookies are also widely used to maintain client authentication and authorization state. In order to prevent tampering, Web servers should sign cookies using a secret key. They should also encrypt sensitive information stored in the cookie.) Anonymous registration is frequently employed for keeping track of advertising and for customer personalization.

The most widely used authentication mechanism for consumer Web site applications is forms-based logon, comprising user ID and password within an encrypted SSL session. The use of X.509 client certificates is becoming more widespread for business applications, but is unlikely to become popular for consumer-oriented sites in the near future.

Authorization suffers from a total lack of industry standardization. Existing third-party approaches are proprietary. Traditionally, most authorization functions have been hard-coded in business logic and therefore expensive to develop and maintain.

LDAP-based directory servers-often cloned for scalability and availability-are located in the DMZ to support authentication and, less common, authorization. Windows 2000 Active Directory, which offers LDAP natively, can support over a million users out of the box. Active Directory's extensible schema can form the basis for secure access control that integrates nicely with the Windows file system, IIS, and other Microsoft products.

Microsoft's Site Server 3.0 and Site Server 3.0 Commerce Edition can scale to support virtually an unlimited number of users. Site Server implements the LDAP protocol over both native Windows directory services and-important for large sites-a SQL Server datastore. Site Server Membership in conjunction with IIS supports both authentication and authorization, including users, groups, credentials,

permissions, roles, and preferences. Membership's extensible schema support site-specific, fine-grained ACLs at the object and even attribute level. Site Server Membership can eliminate much of the burden associated with client access functions, while providing powerful, fully integrated built-in capabilities.

Microsoft's Passport service (www.passport.com/) provides another, very high-level approach to client authentication. Passport is a universal logon service that partners integrate into their own sites. The advantage to the client is single logon access to those partner sites, along with secure, single-click purchasing through Microsoft's companion Wallet service. Because the authentication function is off-loaded to Passport, this reduces the burden on the partner sites to develop and support this capability. Partners also are able to share client profile information and have access to secure credit card data stored in Wallet. Partners must install Passport manager at their sites to broker Passport logins, manage cookies, and transfer wallet data. Passport uses Kerberos-style authentication that makes extensive use of browser cookies.

### Key Points

Site security is not an add-on feature. It is essential to plan security in advance, and base it on assessment of risks and the costs to implement desired protections. The security domain model is a valuable tool to ensure adequate, cost-effective security is implemented throughout the site.

Protection mechanisms can be divided into network security, platform security, and application security (outside the scope of this document). The key elements of network security are firewalls/DMZ, network segmentation and SSL encryption. Platform security consists of hardening operating systems and services, as well as implementation of audit features and monitoring tools.

Client authentication and authorization are required for e-commerce and to support customer personalization. The LDAP protocol supports access control functions. Confidential customer information, especially including passwords and account information, should reside inside the inner firewall.

# Management and Operations

## Introduction

The dependence of sites on networks and uninterrupted services puts considerable pressure on operations staff to ensure ongoing service availability, health, and performance. A well-designed management system is critical to the successful management and operation of large business sites. Great deployment tools allow smooth and rapid growth of Web sites. Great monitoring and troubleshooting tools allow operations staff to quickly deal with problems of components and services before business is affected. The management system itself must be highly available to ensure continuous operations.

Many large sites are geographically remote from operations staff, and hosted in a managed data center close to high-capacity Internet bandwidth. To reduce costs associated with staffing and travel to remote locations, the management network must offer remote capabilities to deploy, provision, monitor, and troubleshoot geographically dispersed sites.

Management and operations of site systems are a very complex and challenging task. The operations

staff faces significant challenges in deploying, fine-tuning, and operating Web site systems. Microsoft, as well as many third-party vendors, offers a wide range of products for managing and administering Windows NT systems. In addition, the suite of Microsoft development tools allows operations staff to customize the management system to better operate a site.

Management of a site should incorporate both system and network management. Microsoft's System Management Server can be used for system management tasks such as planning, deployment, and change and configuration management. A suite of Microsoft tools and services such as Performance Monitor, SNMP Services, WMI, event logging, and backup tools are used for event management, performance management, and storage management (operations management).

Large sites often outsource network management to the Web hosting companies that provide the network infrastructure, services, and facilities (for example, troubleshooting and fixing minor problems as well as around-the-clock monitoring of servers, backbone paths, routers, power systems, or anything else that could affect the delivery of a site's content anywhere in the world). This section concentrates on system and operation management aspects, leaving network management to the Web-hosting providers, and explains how reliable and powerful management systems can be built using Microsoft products and technologies.

The key points addressed in this section are:

- Separating the management network from service networks for high availability and increased security.
- Distributing management network components to:
  - Eliminate or reduce performance bottlenecks.
  - Eliminate single point of failure.
  - Allow independent scaling.
  - Increase availability of the management system.
- Employing Microsoft tools and products where possible to achieve greater performance due to tight integration with the underlying platform.
- Automating tasks where possible.
- Monitoring everything to improve infrastructure and identify problems before they occur.

## Management Infrastructure

### Management network

Management and operations can either share the back-end network or exist on a separate LAN. Management using the back-end network (sometimes called *in-band management*) is less costly and easier to operate. However, it may be unsuitable for managing around-the-clock services for the following reasons:

- In-band management hampers performance of the service network. Management-related notifications such as SNMP traps can flood the network, creating and/or amplifying performance bottlenecks.
- Failure notification is impossible when the service network is down.
- Security implications.

Therefore, for large Web sites, developers should build a separate management network for scalability, availability, and security.

**Management system components**

Normally a management system consists of management consoles, management servers, and management agents.

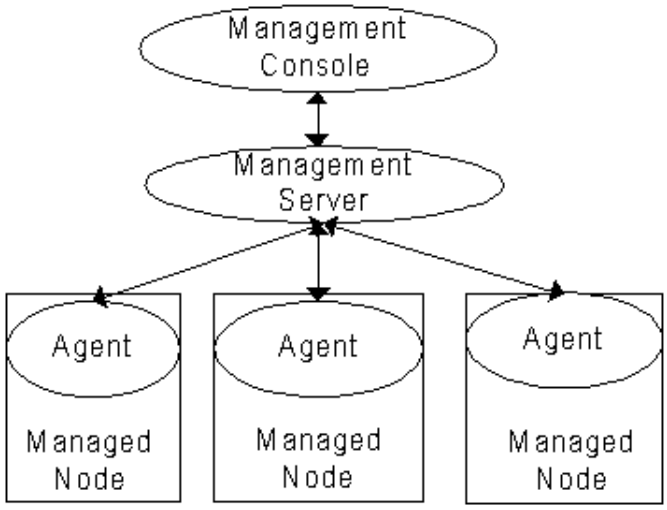


**Figure 13. Management system components**

Figure 13 is a simple diagram illustrating the core management system components and communication between them.

***Management consoles***

Management systems interface with users through *management consoles*. Management consoles are responsible for:

- Logging in and authenticating users (network operators, administrators).
- Providing access to all management servers: Once a management server is accessed, the user can view the status of all managed nodes in that server's jurisdiction and issue commands for software and configuration updates on those nodes.

- Providing response to the user-issued commands.

Many current solutions implement management consoles and management servers, which should be thought of as two logical tiers, in a single tier for reasons of cost savings and ease of use. However, it is sometimes desirable or necessary to decouple the two for availability, scalability, and remoteness of operations centers from managed networks.

***Management servers***

*Management servers* are the workhorses of the management system. Management servers communicate with managed nodes (Windows NT servers, Cisco routers, and other network equipment) through proprietary or standard protocols. Management servers are responsible for:

- Accepting, filtering, and correlating events from managed nodes in its jurisdiction.
- Gathering, storing, and analyzing performance information.
- Distributing and installing software on the managed nodes.
- Updating configuration parameters on managed nodes.

Because management servers collect a lot of information (up to gigabytes of data a day), this information is often stored on separate machines, the *back-end servers*.

***Management agents***

*Management agents* are programs that reside within a managed node. In order to be managed, each device-whether a Windows NT server or a simple network hub-must have a management agent. Management agents perform primary management functions such as:

- Monitoring the resources of the managed device and receiving unsolicited notifications and events from those resources.
- Providing means of configuring and fine-tuning the resources of the managed device.
- Querying on demand the resources of the managed device for their current configuration, state, and performance data.

Some nodes may have only one agent, such as an SNMP-managed network router. Others, such as Windows NT server, are more complex and include multiple agents using different protocols. Agents and servers communicate using standard and proprietary protocols.

**Scaling management infrastructure**

To preserve initial investment, a management system must be able to start small and grow along with the site it manages. As a site expands and new equipment and services are added, the management system has to scale adequately.

A small Web site can be managed with a very simple management system that typically uses the back-end network. The simplest management system is a *centralized system:* a small number of machines installed with management server and console software. Each machine is capable of managing the entire site. The centralized management system is described next. To scale such a management system, the developers must distribute it. Then, we describe the steps necessary to distribute the management system. Finally, we describe an example distributed management system.

***Centralized management system***

A management system can be *centralized* or *distributed*. A single central managing entity, which controls all management systems, characterizes centralized management systems. Centralized management is implemented with one (or more) powerful machine(s) that allow access to all components of the site system, monitor all devices, and accept alarms and notifications from all managed elements. Central management is often done using the main service network.

Because of its simplicity, low cost, and easy administration, a centralized management system may be desirable in small environments such as a start-up site with just a few servers. Microsoft offers a rich set of tools and applications for centralized management such as SMS, PerfMon, Event Log, RoboCopy, and scripting tools. Other applications and tools are available from third-party vendors.

***Distributed management systems***

With rapid growth of a Web site, a centralized management system may prove inefficient. A centralized management system, concentrated on one or two machines, has significant problems: It lacks scalability, it creates performance bottlenecks, and it has a single point of failure. These issues make centralized management systems unsuitable for managing very large, rapidly expanding, and highly available sites. To address scalability and availability problems, management systems should be distributed in the following ways:

- Decouple management consoles from management servers.
- Add more servers so that each manages smaller numbers of nodes.
- Add more consoles to allow access to more administrators and technicians.
- Partition workload between management servers geographically or by management functionality.

***Example management system***

Our example site uses a distributed management system implemented on a separate LAN.

Figure 14 depicts the management system used in our example site. Because the focus of this chapter is on the management system, we show the managed system-the site itself-as a cloud. Refer to Figure 3 for details on example site architecture.

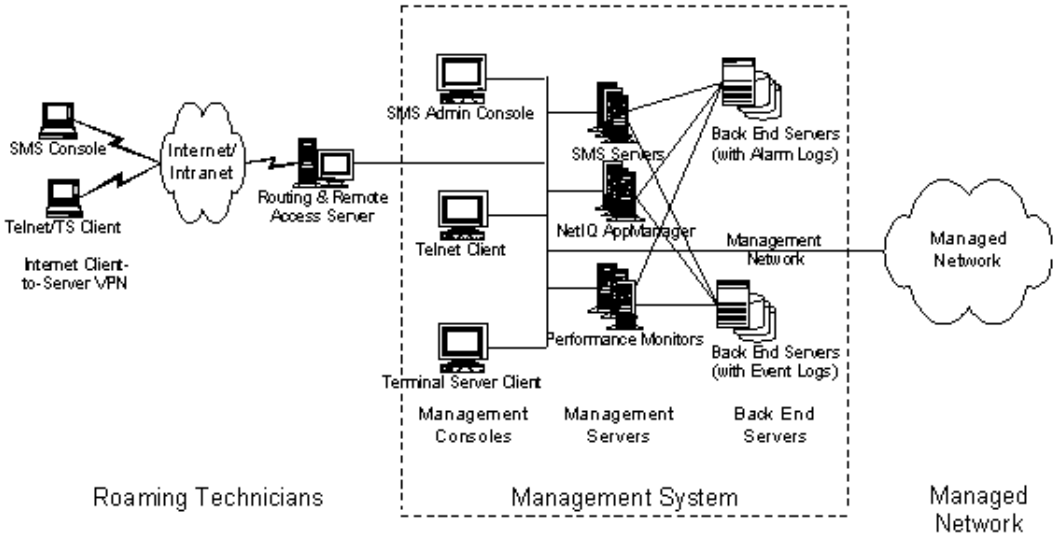


**Figure 14. Example management system**

In this example management system, different line styles, thickness, and annotations show the management LAN, remote access, and applications installed on the management system components. In particular:

- Management network (thick solid).
- RAS Dial-in into the management network (thin solid).

*Management consoles*

In this example management network, management consoles are decoupled from management servers and thus can be concentrated in the (highly secured) Network Operation Center(s). Management tools and applications must be carefully chosen to provide almost all management capabilities remotely.

Management consoles can run Windows NT Server, Workstation, or Professional editions. They would normally have a number of applications installed: System Management Server Administrator Console, Terminal Server (TS) Client, Telnet, Internet Explorer, and SNMP MIB browsers. These tools all provide remote management capabilities and therefore can be used in roaming environments by traveling technicians.

*Management servers*

In a distributed system, each management server serves only managed nodes in its jurisdiction-such as a farm or partition, a floor, a building, a campus, or a city. For example, a local management server can be running in each of the offices in Europe and North America, managing events and networks locally. Distributing management servers and partitioning them to manage only a limited number of nodes allows one to:

- Lock the management servers in secure cabinets.

- Reduce or eliminate network traffic (that is, Windows NT servers in Asia are upgraded by the management server in Tokyo, not the one in New Jersey).

- Eliminate single point of failure.

The same management server does not interact with managed nodes in other areas (however, it is not precluded from doing so).

We recommend that management servers run either Windows NT 4.0 server or Windows 2000 server editions to ensure better stability of the system, and provide additional services that are available only in server editions. Management servers host management applications that provide system and network management capabilities required for a site. Services and applications provided by Microsoft should be installed on the management servers (Performance Monitor, System Management Server (SMS), and Event Logging). SNMP trap managers or trap receivers should also be installed on the management servers.

*Back-end servers (BES)*

Back-end servers are machines with large storage disks that are used for persistent storage of information collected by management servers. It is not necessary to use separate machines to store the management data. However, most large business Web sites log gigabytes of data each day (for later data mining and exploitation) and use separate machines to store this information. Large databases are often used to store events logged by the managed nodes, performance counters, and statistical data. SMS databases can be located on the BESs as well. Back-end servers can also host utilities and tools that manipulate the data stored in the databases: harvesters, parsers, and so on. This allows many customers to use their own highly customized or legacy tools.

### Distributed vs. centralized

Distributed management systems have several key advantages over a centralized management system. They offer better scalability and availability, and reduce or eliminate performance bottlenecks and a single point of failure. However, distributed management systems introduce some deficiencies, such as higher costs (associated with adding more equipment and administration), and growing complexity. When designing a management system for a site, carefully weigh the pros and cons of taking a centralized or distributed management approach.

## Management System Requirements

### Deployment and installation

To successfully deploy new services and equipment, the management system must provide tools for deployment and installation. Deployment includes installing and configuring new equipment and replicating Web site content and data on new machines. The following tools and techniques are most often used to deploy new services and machines.

#### *Unattended/automated server installation*

To deploy new servers, use scripts to build a *golden* (or ideal) version of the server. Then, capture an

image copy of the golden server's system disk using a tool such as Norton Ghost and Ghost Walker (www.ghost.com/) and use that golden image to build new servers.

***SysPrep (Windows 2000)***

SysPrep is a tool (available in the Resource Kit for Windows 2000) designed to deploy fully installed Windows 2000 installations on multiple machines. After performing the initial setup steps on a single system, administrators can run SysPrep to prepare the sample machine for duplication. Web servers of a site farm are normally based on the same image with minor configuration differences like name and IP addresses. Additionally, the combination of SysPrep and a winnt.sif answer file provide the tools for making the minor configuration necessary for each respective machine.

***Content replication***

Content Replication Service and RoboCopy are most often used for content replication. Content Replication Service is part of the Microsoft Site Server product line (www.microsoft.com/siteserver/site/). RoboCopy is a 32-bit Windows command-line application that simplifies the task of maintaining an identical copy of a folder tree in multiple locations. RoboCopy is available in the Windows NT/2000 Resource Kit.

**Change and configuration**

System Management Server provides all the means necessary for change and configuration management of site servers. SMS automates many change and configuration management tasks, such as hardware inventory/software inventory, product compliance, software distribution/installation, and software metering.

More information on Microsoft System Management Server is located at www.microsoft.com/smsmgmt/. Other tools available from third-party vendors are listed at www.microsoft.com/ntserver/management/exec/vendor/ThrdPrty.asp.

**Performance monitoring**

Continuous monitoring is essential for operating a site's continuous services. Many sites use extensive logging and counter-based monitoring, along with as much remote administration as possible, to both ensure continuous availability and to provide the data with which to improve their infrastructure. Tools used to monitor performance of site servers include Performance Monitor, SNMP MIB Browsers, and HTTPMon.

**Event management**

Event management entails monitoring the health and status of site systems (usually in real time), alerting administrators to problems, and consolidating the event logs in a single place for ease of administration. The event monitoring tools may track individual servers or network components, or they may focus on application services like e-mail, transaction processing, or Web service. Event filtering, alerting, and visualization tools are an absolute necessity for sites with hundreds of machines in order to filter out important events from background noise. Tools such as Event Log, SNMP Agents, and SMS (for event to SNMP trap conversion) can be used for event managing.

## Out-of-band emergency recovery

Repairing failed nodes when the management network itself is down presents a difficult manageability problem. When in-band intervention is impossible, out-of-band (OOB) management comes to the rescue.

OOB management refers to products that give technicians access to managed nodes using dial-up telephone lines or a serial cable and not using the management network. Therefore, a serial port must be available on every managed node for out-of-band access. Use OOB management to bring a failed service or node online to repair it in-band, analyze the reasons for failure, and so on.

### *OOB requirements*

OOB management should provide all or some of the following capabilities:

*Operating system and service control*

- Restart the failed service or node.
- Take the failed service or node offline. (This is important because the failed node can flood the network with notifications of failure.)
- Setup and control firmware.
- Change firmware configuration.
- Setup OS and service.

*BIOS and boot device control*

- Hardware power management.
- BIOS configuration and hardware diagnostics.
- Remote console input and output.

### *OOB solutions*

Many solutions are available for performing the tasks just described. Table 2 summarizes the most widely used solutions from Microsoft and third-party vendors.

**Table 2. Out-of-band solutions**

| Capability | Name | Vendor |
|---|---|---|
| Terminal Server | Available with Windows NT 4.0 Terminal Server Edition or 2000 Server<br><br>TermServ | Microsoft<br><br>Seattle Labs |
| Setup and installation | Unattended OS and Post OS shell Scripts<br><br>Ghost<br><br>IC3<br><br>Remote Installation Service (Windows 2000 only) | Microsoft<br><br>Norton<br><br>ImageCast<br><br>Microsoft |
| BIOS configuration and hardware diagnostics | Integrated Remote Console (IRC)<br><br>Remote Insight Board (RIB)<br><br>Emerge Remote Server Access | Compaq<br><br>Compaq<br><br>Apex |
| Hardware power management | Integrated Remote Console (IRC)<br><br>Remote Insight Board (RIB)<br><br>Remote Power Control | Compaq<br><br>Compaq<br><br>Baytech |

### *OOB security*

Dialing into the console port exposes the network to access. Prevent this by securing OOB operations. At a minimum, strong authentication of administrative staff should be required, usually with one-time (challenge-response) passwords provided by security tokens. Administrators are provided with either hardware or software-based tokens, which negotiate with an access server at the target site. This opens a connection to a terminal server, which in turn provides serial port access to a specific host. Ideally, employ link encryption as well in order to prevent snooping or possible compromise by an intruder. An increasingly popular solution is to use public-key based VPNs (Virtual Private Networks) to provide both strong authentication as well as encryption.

### Automation of management tasks

Management system design should allow implementation of automated actions such as stopping or starting a service or entire node, running a script or a batch file when certain events occur, or attempting an out-of-band recovery if the management network is unavailable. Well-designed systems will automatically notify IT technicians of events or problems using e-mail, telephone, pagers, or cell phones.

Many tools and applications can automate management tasks:

- Set an alert on a counter in the Alert View of **Performance Monitor**, thereby triggering a message to be sent, a program to be run, or a log to be started when the selected counter's value equals, exceeds, or falls below a specified setting.

- **SNMP Managers** provide automation tools to generate notifications and start a program, batch, or script when certain traps are received.

- **BackOffice** components such as Microsoft Exchange and SQL Server can trigger exception when service-specific events occur (for example, remote mail server doesn't respond to messages within a predefined interval). Microsoft Exchange Server, for example, can send e-mail, display on-screen alerts, or route notifications to an external application.

- Use the **Windows Script Host (WSH)** or any other scripting mechanism to write flexible scripts, which monitor the system and generate messages or trigger automated jobs when needed.

Many third-party solutions that allow automation are also available, listed on: www.microsoft.com/ntserver/management/exec/vendor/ThrdPrty.asp.

### Security

Security of the management infrastructure is of paramount importance, because compromise of this subsystem can lead to compromise of every other component of a site. All of the elements of security discussed in the preceding section, "Security," which discusses security architecture, apply here.

Although very widely used, one of the most popular management protocols-SNMP-is poor from a security standpoint. The SNMP community string is a very weak password. While it does not permit a user to log on, it does permit someone to take control of a node. Carefully choose and tightly control management protocols for each site.

# Summary

In this document, we have shown how the Microsoft Windows platform and other Microsoft technologies are effectively used to build a scalable, available, secure, and manageable site infrastructure. We have stressed keeping operations and application design of a large site simple and flexible, and have emphasized how a dot-com can successfully deploy and operate a site based on the four goals of the architecture:

- Linear *scalability*-continuous growth to meet user demand and business complexity.

- Continuous service *availability*-using redundancy and functional specialization to contain faults.

- *Security* of data and infrastructure-protecting data and infrastructure from malicious attacks or theft.

- Ease and completeness of *management*-ensuring that operations can match growth.

We anticipate that this will be the first of many documents covering the breadth and depth of designing,

developing, deploying, and operating great business Web sites that use Microsoft's products and technologies.

---