Microsoft

Windows Server 2003 evaluation software inside!

INTRODUCING Microsoft WINDOWS SERVER™ 2003

Jerry Honeycutt

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Library of Congress Cataloging-in-Publication Data
Honeycutt, Jerry.
Introducing Microsoft Windows Server 2003 / Jerry Honeycutt.
p. cm.
Includes index.
ISBN 0-7356-1570-5
1. Microsoft Windows Server. 2. Operating systems (Computers). I. Title.

QA76.76.O63H6632 2003

2002043103

Printed and bound in the United States of America.

1 2 3 4 5 6 7 8 9 QWE 8 7 6 5 4 3

Distributed in Canada by H.B. Fenn and Company Ltd.

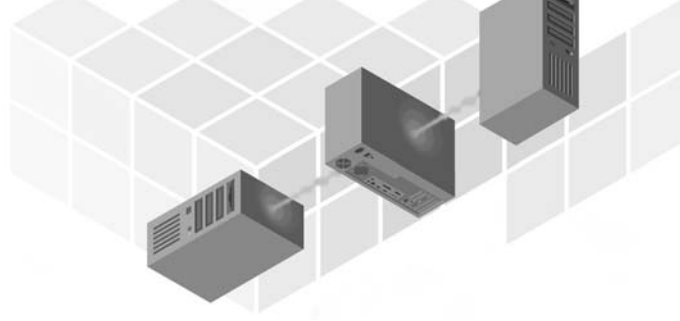
A CIP catalogue record for this book is available from the British Library.

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office or contact Microsoft Press International directly at fax (425) 936-7329. Visit our Web site at www.microsoft.com/mspress. Send comments to *mspinput@microsoft.com*.

Acquisitions Editor: Martin DelRe
Project Editor: Valerie Woolley
Technical Editor: Dail Magee Jr.

Body Part No. X08-68164

Table of Contents



5

Security Services

Businesses have extended the traditional local area network (LAN) by combining intranets, extranets, and Internet sites; as a result, increased system security is now more critical than ever before. To provide a secure computing environment, the Microsoft Windows Server 2003 family includes many important new security features and improves on the security features originally included in Microsoft Windows 2000 Server.

Viruses exist, and software security is an ongoing challenge. To address these facts, Microsoft has made Trustworthy Computing a key initiative for all its products. Trustworthy Computing is a framework for developing devices powered by computers and software that are as secure and trustworthy as the everyday devices and appliances you use at home. While no Trustworthy Computing platform exists today, the basic redesign of Windows Server 2003 is a solid step toward making this vision a reality.

The common language runtime (CLR) software engine is a key element of Windows Server 2003 that improves reliability and helps ensure a safe computing environment. It reduces the number of bugs and security holes caused by common programming mistakes—as a result, there are fewer vulnerabilities for attackers to exploit. The CLR verifies that applications can run without error and checks for appropriate security permissions, making sure that code performs appropriate operations exclusively. It does this by checking where the code was downloaded or installed from, whether it has a digital signature from a trusted developer, whether it has been altered since it was digitally signed, and so forth.

As part of its commitment to reliable, secure, and dependable computing, Microsoft has reviewed every line of code underlying its Windows Server 2003 family as part of an enhanced effort to identify possible fail points and exploitable weaknesses.

This chapter discusses the tools and processes that deliver important security benefits to organizations deploying Windows Server 2003. These include authentication, access control, security policy, auditing, Active Directory, data protection, network data protection, public key infrastructure (PKI), and trusts.

Security Benefits

Windows Server 2003 will provide a more secure and economical platform for doing business than earlier versions of Windows.

- **Lower costs.** Lower costs result from simplified security management processes such as access control lists, Credential Manager, and PKI.
- **Implementation of open standards.** The IEEE 802.1X protocol makes it easy to secure wireless LANs from the threat of eavesdropping within your business environment. For more information about other supported standards, see RFCs 3280, 2797, 2527, and 2459 and public key cryptography standards (PKCS) 1, 5, 8, 10, and 12.
- **Protection for mobile computers and other new devices.** Security features such as Encrypting File System (EFS), certificate services, and automatic smart card enrollment make it easier to secure a full range of devices. EFS is the core technology for encrypting and decrypting files stored on NTFS volumes. Only the user who encrypts a protected file can open the file and work with it. Certificate Services is the part of the core operating system that allows a business to act as its own certification authority (CA) and issue and manage digital certificates. Automatic certificate enrollment and self-registration authority features provide enhanced security for enterprise users by adding another layer of authentication; this is in addition to simplified security processes for security-conscious organizations.

Authentication

Authentication is the process of verifying that a person, an entity, or an object is who or what he, she, or it claims to be. Examples include confirming the source and integrity of information, such as verifying a digital signature or verifying the identity of a user or computer.

Authentication is a fundamental aspect of system security. It confirms the identity of any user trying to log on to a domain or access network resources. Windows Server 2003 family authentication enables single sign-on to all network resources. With single sign-on, a user can log on to the domain once, using a single password or smart card, and authenticate to any computer in the domain.

Authentication Types

In attempting to authenticate a user, several industry-standard types of authentication can be used, depending on a variety of factors. The types of authentication that the Windows Server 2003 family supports are as follows:

- **Kerberos V5 authentication.** This protocol is used with either a password or a smart card for interactive logon. It is also the default method of network authentication for services.
- **Secure Sockets Layer/Transport Layer Security (SSL/TLS) authentication.** This protocol is used when a user attempts to access a secure Web server.
- **NTLM authentication.** This protocol is used when either the client or the server uses a previous version of Windows.
- **Digest authentication.** Digest authentication transmits credentials across the network as an MD5 hash or message digest.
- **Passport authentication.** Passport authentication is a user-authentication service that offers single-sign-on service.

Internet Information Services Security

When you use Internet Information Services (IIS), authentication is critical to security. IIS 6.0 is a full-featured Web server that provides the foundation for the Microsoft .NET Framework and existing Web applications and Web services. IIS 6.0 has been optimized to run Web applications and Web services in a hosting environment. Many new features have been included in IIS to enhance security, reliability, manageability, and performance.

Using IIS, you can isolate an individual Web application or multiple sites into a self-contained Web service process that communicates directly with the kernel. These self-contained Web service processes prevent one application or site from disrupting the Web services of other Web applications on the server. IIS also provides health monitoring capabilities to discover, recover, and prevent Web application failures.

Because security is an important consideration for a Web server, you can use IIS to protect your Web server from real-world attacks. IIS is a robust platform that provides the tools and features necessary to easily manage a secure server. For more information about security features in IIS 6.0, see Chapter 8, "Internet Information Services."

Interactive Logon

Interactive logon confirms the user's identification to the user's local computer or Active Directory account. For more information about Active Directory and security, see Chapter 3, "Active Directory."

Network Authentication

Network authentication confirms the user's identification to any network service that the user is attempting to access. To provide this type of authentication, the security system includes these authentication mechanisms:

- Kerberos V5
- Public key certificates
- Secure Sockets Layer/Transport Layer Security (SSL/TLS) Digest
- NTLM (for compatibility with Windows NT 4.0–based systems)

Single Sign-On

Single sign-on makes it possible for users to access resources over the network without having to repeatedly supply their credentials. For the Windows Server 2003 family, users need to authenticate only once to access network resources; subsequent authentication is transparent to the user.

Two-Factor Authentication

Authentication in the Windows Server 2003 family also includes two-factor authentication, such as smart cards. Smart cards are a tamper-resistant and portable way to provide security solutions for tasks such as client authentication, logging on to a Windows Server 2003 family domain, code signing, and securing e-mail. Support for cryptographic smart cards is a key feature of the public key infrastructure (PKI) that Microsoft has integrated into Windows XP and the Windows Server 2003 family. Smart cards provide the following:

- Tamper-resistant storage for protecting private keys and other forms of personal information.

- Isolation of security-critical computations involving authentication, digital signatures, and key exchange from other parts of the computer that do not have a need to know. These operations are all performed on the smart card.
- Portability of credentials and other private information between computers at work, at home, or on the road.

Logging on to a network with a smart card provides a strong form of authentication because it uses cryptography-based identification and proof of possession when authenticating a user to a domain. For example, if a malicious person obtains a user's password, that person can assume the user's identity on the network simply through use of the password. Many people choose passwords they can remember easily, which makes passwords inherently weak and open to attack.

In the case of smart cards, that same malicious person would have to obtain both the user's smart card and the personal identification number (PIN) to impersonate the user. This combination is obviously more difficult to attack because an additional layer of information is needed to impersonate a user. An additional benefit is that, after a small number of unsuccessful PIN inputs occur consecutively, a smart card is locked, making a dictionary attack against a smart card extremely difficult. (Note that a PIN does not have to be a series of numbers; it can also use other alphanumeric characters.) Smart cards are also resistant to undetected attacks because the card needs to be obtained by the malicious person, which is relatively easy for a user to know about.

To log on to a domain with a smart card, users do not need to press Ctrl+Alt+Del. They simply insert the smart card into the smart card reader, and the computer prompts them for their personal identification number (PIN) instead of their user name and password.

Object-Based Access Control

Along with user authentication, administrators are allowed to control access to resources or objects on the network. To do this, administrators assign security descriptors to objects that are stored in Active Directory. A security descriptor lists the users and groups that are granted access to an object and the specific permissions assigned to those users and groups. A security descriptor also specifies the various access events to be audited for an object. Examples of objects include users, computers, and organizational units (OUs). By managing properties on objects, administrators can set permissions, assign ownership, and monitor user access.

Not only can administrators control access to a specific object, they can also control access to a specific attribute of that object. For example, through proper configuration of an object's security descriptor, a user can be allowed to access only a subset of information, such as employees' names and telephone numbers but not their home addresses. To secure a computer and its resources, you must take into consideration the rights that users will have:

- You can secure a computer or multiple computers by granting users or groups specific user rights.
- You can secure an object, such as a file or folder, by assigning permissions to allow users or groups to perform specific actions on that object.

Access Control Concepts

Permissions define the type of access granted to a user or group for an object or object property. For example, the Finance group can be granted Read and Write permissions for a file named Payroll.dat. Permissions are applied to any secured objects such as files, Active Directory objects, or registry objects. Permissions can be granted to any user, group, or computer. (It's good practice to assign permissions to groups.) The permissions attached to an object depend on the type of object. For example, the permissions that can be attached to a file are different from those that can be attached to a registry key. You can assign permissions for objects to the following:

- Groups, users, and special identities in the domain
- Groups and users in that domain and any trusted domains
- Local groups and users on the computer where the object resides

When you set up permissions, you specify the level of access for groups and users. For example, you can let one user read the contents of a file, let another user make changes to the file, and prevent all other users from accessing the file. You can set similar permissions on printers so that certain users can configure the printer and other users can only print from it. If you need to change the permissions on an individual object, you can start the appropriate tool and change the properties for that object. For example, to change the permissions on a file, you can run Windows Explorer, right-click the filename, and click Properties. On the Security tab, you can change permissions on the file.

An owner is assigned to an object when that object is created. By default, the owner is the creator of the object. No matter which permissions are set on an object, the owner of the object can always change the permissions on an object.

Inheritance allows administrators to easily assign and manage permissions. This feature automatically causes objects within a container to inherit all the inheritable permissions of that container. For example, the files within a folder, when created, inherit the permissions of the folder. Only permissions marked to be inherited are inherited.

Effective Permissions

The Effective Permissions tab is a new, advanced option in Windows Server 2003. It lets you see all of the permissions that apply to a security principal for a given object, including the permissions derived from memberships in security groups. The Effective Permissions tab is shown in Figure 5-1.

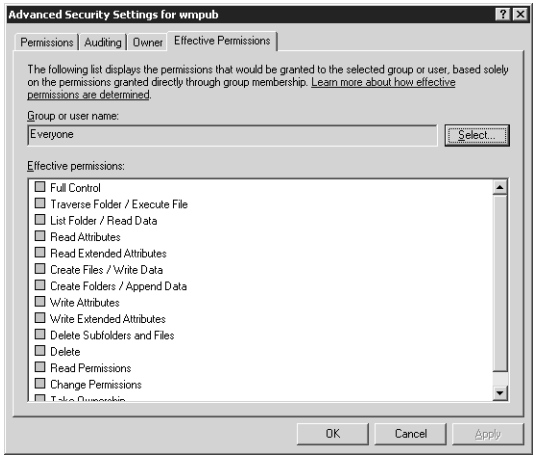


Figure 5-1 The Effective Permissions tab is new with Windows Server 2003.

To view the effective permissions for a user or group, perform the following steps:

1. On the Effective Permissions tab, click the Select button to open the Select User Or Group dialog box.
2. In the Name box, type the name of the built-in security principal, group, or user for which you would like to view Effective Permissions.
3. Optionally, click the Object Types button, and then select Built-In Security Principals, Groups, or Users.
4. Click OK.

> **Note** If the security principal is network based, you can click Locations and select a target, or you can type in the domain name together with the group name, such as reskit\users. It's important to specify the correct object types and the locations for your search. Failure to do so will result in an error message and the suggestion that you refine your search before searching again.

User Rights

User rights grant specific privileges and logon rights to users and groups in your computing environment.

Object Auditing

You can audit users' access to objects. You can then view these security-related events in the security log with the Event Viewer.

Security Policy

You can control security on your local computer, or on multiple computers, by controlling the following: password policies, account lockout policies, Kerberos policies, auditing policies, user rights, and other policies.

To create a systemwide policy, you can use security templates; apply templates using the Security Configuration and Analysis snap-in; or edit policies on the local computer, organizational unit, or domain.

Security Configuration Manager

The Security Configuration Manager tool set lets you create, apply, and edit security variables for your local computer, organizational unit, or domain. The following list describes the components of the Security Configuration Manager tool set:

- **Security Templates.** Defines a security policy in a template. These templates can be applied to Group Policy or to your local computer.
- **Security Settings Extension to Group Policy.** Edits individual security settings on a domain, a site, or an organizational unit.

- **Local Security Policy.** Edits individual security settings on your local computer.
- **Secedit Commands.** Automates security configuration tasks at a command prompt.

Security Configuration and Analysis

Security Configuration and Analysis is a Microsoft Management Console (MMC) snap-in for analyzing and configuring local system security.

Security Analysis

The state of the operating system and applications on a computer is dynamic. For example, you might need to temporarily change security levels so that you can immediately resolve an administration or network issue. However, this change can often go unreversed. This means that a computer might no longer meet the requirements for enterprise security.

Regular analysis enables an administrator to track and ensure an adequate level of security on each computer as part of an enterprise risk management program. An administrator can tune the security levels and, most important, detect any security flaws that might occur in the system over time.

Security Configuration and Analysis lets you quickly review security analysis results. It presents recommendations alongside current system settings and uses visual flags or remarks to highlight any areas where the current settings do not match the proposed level of security. Security Configuration and Analysis also offers the ability to resolve any discrepancies that that analysis reveals.

Security Configuration

Security Configuration and Analysis can be used to directly configure local system security. Through its use of personal databases, you can import security templates that have been created with Security Templates and apply those templates to the local computer. This immediately configures the system security with the levels specified in the template.

Auditing

Auditing gives you a way to track potential security problems, helps to ensure user accountability, and provides evidence in the event of a security breach. To audit effectively, you need to establish an audit policy. This requires you to

determine which categories of events, which objects, and which accesses you want to audit.

Establish a Strategy

Your policy should be based on a strategy. For instance, you might decide that you are interested in a record of who accessed the system or specific data on the system, or that you are interested in detecting unauthorized attempts to tamper with the operating system.

Common Events to Be Audited

The most common types of events to be audited are the following:

- Users logging on to and logging off the system
- Management of user accounts and groups
- Accesses of objects, such as files and folders

Implementing Auditing Policy

When you implement auditing policy, keep the following points in mind:

- Develop your audit strategy. Decide which behaviors you want to audit.
- Select the audit categories that correspond to your auditing strategy, and no more.
- Select an appropriate size and retention policy for the security log. You can view the security log and set the log size and retention policy with Event Viewer, as shown in Figure 5-2.
- If you have decided to audit directory service access or object access, determine which objects must be monitored as part of your strategy. Also determine the minimum number of accesses you need to audit to fulfill the goals of your strategy. You shouldn't audit any more objects or accesses than necessary because a too-broad audit selection could cause audit logs to fill very rapidly on a busy machine.
- Deploy your policy. You can do this with the Local Security Policy tool on a stand-alone machine or with Group Policy on a domain.

- Review your security logs regularly. There's no point in auditing if you're never going to look at your logs. An event log collection system can help make this a manageable task.
- Fine-tune your policy as necessary. This might include adding or removing objects or accesses to your audit policy, or enabling or disabling audit categories. After reviewing your logs, you might find that you have collected more or less information than you want.

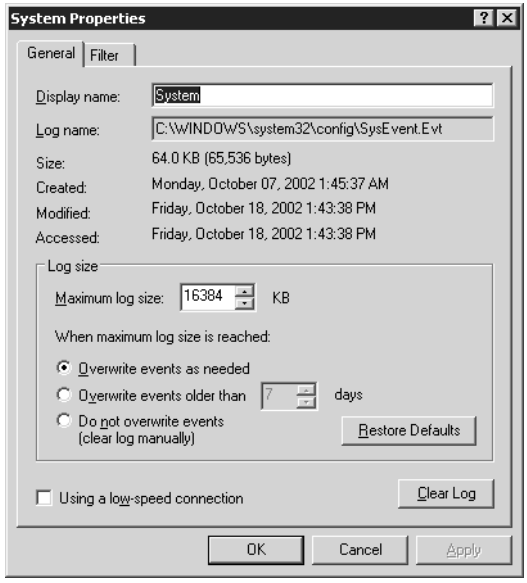


Figure 5-2 Administrators can easily configure the event log's size and retention policy.

Active Directory and Security

The Active Directory directory service ensures that administrators can manage user authentication and access control easily and efficiently. See Chapter 3, "Active Directory," for more information about security and Active Directory.

Active Directory provides protected storage of user account and group information by using access control on objects and user credentials. Because Active Directory stores not only user credentials but also access-control information, users who log on to the network obtain both authentication and authorization to access system resources. For example, when a user logs on to the

network, the security system authenticates the user with information stored in Active Directory. Then, when the user attempts to access a service on the network, the system checks the properties defined in the discretionary access control list (DACL) for that service.

Because Active Directory allows administrators to create group accounts, administrators can manage system security more efficiently than ever before. For example, by adjusting a file's properties, an administrator can permit all users in a group to read that file. In this way, access to objects in Active Directory is based on group membership.

Data Protection

Stored data (on line or off line) can be protected by using Encrypting File System (EFS) and digital signatures. Stored data security refers to the ability to store data on disk in an encrypted form.

Encrypting File System

With EFS, data can be encrypted as it is stored on disk. EFS uses public key encryption to encrypt local NTFS data. Once a user has encrypted a file, the file automatically remains encrypted whenever the file is stored on disk. And once a user has decrypted a file, the file remains decrypted whenever the file is stored on disk. EFS provides the following features:

- Users can encrypt their files when storing them on disk. Encryption is as easy as selecting a check box in the file's Advanced Attributes dialog box (accessed via the file's Properties dialog box), as shown in Figure 5-3.
- Accessing encrypted files is fast and easy. Users see their data in plain text when accessing the data from disk.
- Encryption of data is accomplished automatically and is completely transparent to the user.
- Users can actively decrypt a file by clearing the Encrypt Contents check box in the file's Advanced Attributes dialog box.
- Administrators can recover data that was encrypted by another user. This ensures that data is accessible if the user who encrypted the data is no longer available or has lost his or her private key.

Figure 5-3 Encrypting files is as easy as selecting the Encrypt Contents check box.

> **Note** EFS encrypts data only when it is stored on disk. To encrypt data as it is transported over a TCP/IP network, two optional features are available—Internet Protocol security (IPSec) and PPTP encryption.

The default configuration of EFS requires no administrative effort—users can begin encrypting files immediately. EFS generates an encryption key pair for a user if one does not exist. EFS can use either the expanded Data Encryption Standard (DESX) or Triple-DES (3DES) as the encryption algorithm. Encryption services are available from Windows Explorer. Users can also encrypt a file or folder using the command-line utility *cipher*. For more information about the *cipher* command, type **cipher /?** at a command-line prompt. Users encrypt a file or folder by setting the encryption property for files and folders just as you set any other attribute, such as read-only, compressed, or hidden. If a user encrypts a folder, all files and subfolders created in or added to the encrypted folder are automatically encrypted. It is recommended that users encrypt at the folder level. Files or folders that are compressed cannot also be encrypted. If the user marks a compressed file or folder for encryption, that file or folder will be uncompressed. Also, folders that are marked for encryption are not actually encrypted. Only the files within the folder are encrypted, as well as any new files created or moved into the folder. Once decrypted, a file remains decrypted until you encrypt the file again. There is no automatic reencryption of a file, even if it exists in a directory marked as encrypted.

Data recovery refers to the process of decrypting a file without having the private key of the user who encrypted the file. You might need to recover data with a recovery agent if a user leaves the company, a user loses the private key, or a law enforcement agency makes a request. To recover a file, the recovery agent does the following:

1. Backs up the encrypted files
2. Moves the backup copies to a secure system
3. Imports their recovery certificate and private key on that system
4. Restores the backup files
5. Decrypts the files, using Windows Explorer or the EFS *cipher* command

You can use the Group Policy snap-in to define a data recovery policy for domain member servers, or for stand-alone or workgroup servers. You can either request a recovery certificate or export and import your recovery certificates. You might want to delegate administration of the recovery policy to a designated administrator. Although you should limit who is authorized to recover encrypted data, allowing multiple administrators to act as recovery agents provides you with an alternative source if recovery is necessary.

Digital Signatures

A digital signature is a way to ensure the integrity and origin of data. A digital signature provides strong evidence that the data has not been altered since it was signed and confirms the identity of the person or entity that signed the data. This enables the important security features of integrity and nonrepudiation, which are essential for secure electronic commerce transactions.

Digital signatures are typically used when data is distributed in clear text, or unencrypted form. In these cases, while the sensitivity of the message itself might not warrant encryption, there could be a compelling reason to ensure that the data is in its original form and has not been sent by an impostor because, in a distributed computing environment, clear text can conceivably be read or altered by anyone on the network with the proper access, whether authorized or not.

CAPICOM

Windows Server 2003 includes support for CAPICOM 2.0. This support enables application developers to take advantage of the robust certificate and cryptography features available in CryptoAPI by employing an easy-to-use COM inter-

face. Using this functionality, application developers can easily incorporate digital signing and encryption functionality into their applications. Because CAPICOM is based on COM, application developers can access this functionality in a number of programming environments, such as the Visual C# development tool, the Visual Basic .NET development system, Visual Basic, Visual Basic Scripting Edition, JScript development software, and others.

CAPICOM allows you to do the following:

- Digitally sign and verify arbitrary data with a smart card or software key
- Digitally sign and verify executables with Authenticode technology
- Hash arbitrary data
- Graphically display certificate selection and detailed information
- Manage and search CryptoAPI certificate stores
- Encrypt and decrypt data with a password, or with public keys and certificates

Network Data Protection

Network data within your site (local network and subnets) is secured by the authentication protocol. For an additional level of security, you can also choose to encrypt network data within a site. Using Internet Protocol security, you can encrypt all network communication for specific clients or for all clients in a domain. Network data passing in and out of your site (across intranets, extranets, or an Internet gateway) can be secured by using the following utilities:

- **Internet Protocol Security (IPSec)** Comprises a suite of cryptography-based protection services and security protocols
- **Routing and Remote Access** Configures remote access protocols and routing
- **Internet Authentication Service (IAS)** Provides security and authentication for dial-in users

Internet Protocol Security

The long-term direction for secure networking, IPSec is a suite of cryptography-based protection services and security protocols. Because it requires no changes to applications or protocols, you can easily deploy IPSec for existing networks.

IPSec provides computer-level authentication, as well as data encryption, for virtual private network (VPN) connections that use the Layer 2 Tunneling Protocol (L2TP). IPSec is negotiated between your computer and a L2TP-based VPN server before an L2TP connection is established. This negotiation secures both passwords and data. L2TP uses standard PPP-based authentication protocols, such as Extensible Authentication Protocol (EAP), Microsoft Challenge Handshake Authentication Protocol (MS-CHAP), MS-CHAP version 2, CHAP, Shiva Password Authentication Protocol (SPAP), and Password Authentication Protocol (PAP) with IPSec.

Encryption is determined by the IPSec Security Association (SA). A security association is a combination of a destination address; a security protocol; and a unique identification value, called a Security Parameters Index (SPI). The available encryptions include

- Data Encryption Standard (DES), which uses a 56-bit key
- Triple DES (3DES), which uses three 56-bit keys and is designed for high-security environments

Routing and Remote Access

The Routing and Remote Access service for the Windows Server 2003 family is a full-featured software router and is an open platform for routing and internetworking. It offers routing services to businesses in LAN and WAN environments or over the Internet by using secure VPN connections.

An advantage of the Routing and Remote Access service is integration with the Windows Server 2003 family. The Routing and Remote Access service delivers many cost-saving features, and it works with a wide variety of hardware platforms and hundreds of network adapters. The Routing and Remote Access service is extensible with application programming interfaces (APIs) that developers can use to create custom networking solutions and that new vendors can use to participate in the growing business of open internetworking.

Internet Authentication Service

Internet Authentication Service (IAS) in the Standard Edition, Enterprise Edition, and Datacenter Edition of Windows Server 2003 is the Microsoft implementation of a Remote Authentication Dial-In User Service (RADIUS) server and proxy:

- As a RADIUS server, IAS performs centralized connection authentication, authorization, and accounting for many types of network access, including wireless, authenticating switch, remote access dial-up, and VPN connections.
- As a RADIUS proxy, IAS forwards authentication and accounting messages to other RADIUS servers. RADIUS is an Internet Engineering Task Force (IETF) standard.

Public Key Infrastructure

Computer networks are no longer closed systems in which a user's mere presence on the network can serve as proof of identity. In this age of information interconnection, an organization's network can consist of intranets, Internet sites, and extranets—all of which are potentially susceptible to access by unauthorized individuals who intend to maliciously view or alter an organization's digital information assets.

There are many potential opportunities for unauthorized access to information on networks. A person can attempt to monitor or alter information streams such as e-mail, electronic commerce transactions, and file transfers. Your organization might work with partners on projects of limited scope and duration having employees about whom you know nothing but who, nonetheless, must be given access to some of your information resources. If your users have a multitude of passwords to remember for accessing different secure systems, they might choose weak or common passwords to more easily remember them. This provides an intruder with not only a password that is easy to crack but also one that will provide access to multiple secure systems and stored data.

How can a system administrator be sure of the identity of a person accessing information, and given that identity, control which information that person has access to? Additionally, how can a system administrator easily and securely distribute and manage identification credentials across an organization? These are issues that can be addressed with a well-planned public key infrastructure. A public key infrastructure (PKI) is a system of digital certificates, certification authorities (CAs), and other registration authorities (RAs) that verify and authenticate the validity of each party that is involved in an electronic transaction through the use of public key cryptography. Standards for PKIs are still evolving, even as they are being widely implemented as a necessary element of electronic commerce.

An organization might choose to deploy a PKI using Windows for a number of reasons:

- **Strong security.** You can have strong authentication with smart cards. You can also maintain the confidentiality and integrity of transmitted data on public networks by using IPSec, and you can protect the confidentiality of your stored data using EFS.
- **Simplified administration.** Your organization can issue certificates and, in conjunction with other technologies, eliminate the use of passwords. You can revoke certificates as necessary and publish certificate revocation lists (CRLs). There is the ability to use certificates to scale trust relationships across an enterprise. You can also take advantage of Certificate Services integration with Active Directory and policy. The capability to map certificates to user accounts is also available.
- **Additional opportunities for PKI.** You can exchange files and data securely over public networks, such as the Internet. You have the ability to implement secure e-mail using Secure Multipurpose Internet Mail Extensions (S/MIME) and secure Web connections using Secure Sockets Layer (SSL) or Transport Layer Security (TLS). You can also implement security enhancements to wireless networking.

The following sections describe the features in the Windows Server 2003 family that can help your organization implement a public key infrastructure.

Certificates

A certificate is basically a digital statement issued by an authority that vouches for the identity of the certificate holder. A certificate binds a public key to the identity of the person, computer, or service that holds the corresponding private key. Certificates are used by a variety of public key security services and applications that provide authentication, data integrity, and secure communication across networks such as the Internet.

The standard certificate format used by Windows certificate–based processes is X.509v3. An X.509 certificate includes information about the person to whom or the entity to which the certificate is issued, information about the certificate, and optional information about the certification authority issuing the certificate. Subject information can include the entity's name, the public key, and the public key algorithm. The entity receiving the certificate is the subject of the certificate. The issuer and signer of the certificate is a certification authority.

Users can manage certificates using the MMC snap-in for certificates, as shown in Figure 5-4. Users can also allow certificate autoenrollment to manage their certificates automatically.

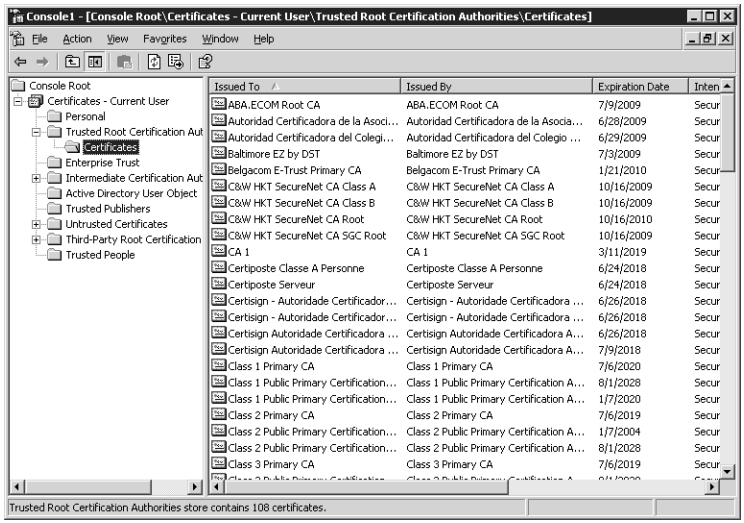


Figure 5-4 You manage certificates using Microsoft Management Console.

Certificates can be issued for a variety of functions, such as Web user authentication, Web server authentication, secure e-mail (S/MIME), IPSec, TLS, and code signing. Certificates are also issued from one CA to another in order to establish a certification hierarchy. Typically, certificates contain the following information:

- The subject's public key value
- The subject's identifier information, such as name and e-mail address
- The validity period (the length of time that the certificate is considered valid)
- Issuer identifier information
- The digital signature of the issuer, which attests to the validity of the binding between the subject's public key and the subject's identifier information

A certificate is valid only for the period of time specified within it; every certificate contains Valid From and Valid To dates, which set the boundaries of the validity period. Once a certificate's validity period has passed, the subject of the now-expired certificate must request a new certificate.

In instances in which it becomes necessary to undo the binding that is asserted in a certificate, the issuer can revoke the certificate. Each issuer maintains a certificate revocation list that can be used by programs when checking the validity of any given certificate.

One of the main benefits of certificates is that hosts no longer have to maintain a set of passwords for individual subjects who need to be authenticated as a prerequisite for access. Instead, the host merely establishes trust in a certificate issuer. When a host, such as a secure Web server, designates an issuer as a trusted root authority, the host implicitly trusts the policies that the issuer has used to establish the bindings of certificates it issues. In effect, the host trusts that the issuer has verified the identity of the certificate subject. A host designates an issuer as a trusted root authority by placing the issuer's self-signed certificate, which contains the issuer's public key, into the trusted root certification authority certificate store of the host computer. Intermediate or subordinate certification authorities are trusted only if they have a valid certification path from a trusted root certification authority.

Certificate Services

Certificate Services is the component in the Windows Server 2003 family that is used to create and manage CAs. A CA is responsible for establishing and vouching for the identity of certificate holders. A CA also revokes certificates if they should no longer be considered valid and publishes CRLs to be used by certificate verifiers.

The simplest PKI design has only one root CA. In practice, however, a majority of organizations deploying a PKI will use a number of CAs, organized into certification hierarchies. Administrators can manage Certificate Services by using the Certification Authority MMC snap-in.

Certificate Templates

Certificates are issued by the CA based on information provided in the certificate request and on settings contained in a certificate template. A certificate template is the set of rules and settings that are applied against incoming certificate requests. For each type of certificate that an enterprise CA can issue, a certificate template must be configured.

Certificate templates are customizable in Windows Server 2003, Enterprise Server, and Windows Server 2003, Datacenter Server, enterprise CAs, and they are stored in Active Directory for use by all CAs in the forest. This allows the administrator to choose one or more of the default templates

installed with Certificate Services or to create templates that are customized for specific tasks or roles.

Certificate Autoenrollment

Autoenrollment enables the administrator to configure subjects to automatically enroll for certificates, retrieve issued certificates, and renew expiring certificates without requiring subject interaction. Such configuration requires no knowledge by the subject of any certificate operations—unless the certificate template is configured to interact with the subject or the cryptographic service provider (CSP) requires interaction (such as with a smart card CSP). This greatly simplifies the experience of the client with certificates and minimizes administrative tasks. Administrators can configure autoenrollment through configuration of Certificate Templates and CA settings.

Web Enrollment Pages

Web enrollment pages are a separate component of Certificate Services. These Web pages are installed by default when you set up a CA and allow certificate requesters to submit certificate requests using a Web browser.

Additionally, the CA Web pages can be installed on servers running Windows that do not have a CA installed. In this case, the Web pages are used to direct certificate requests to a CA that, for whatever reason, you do not want requesters to directly access.

If you choose to create custom Web pages for your organization to access a CA, the Web pages provided with Windows Server 2003 can be used as samples. Refer to the Microsoft Platform Software Development Kit for information about customizing Certificate Services and CA Web pages.

Smart Card Support

Windows supports logon via certificates on smart cards, as well as the use of smart cards to store certificates and private keys. Smart cards can be used for Web authentication, secure e-mail, wireless networking, and other activities related to public key cryptography.

Public Key Policies

You can use Group Policy in Windows to distribute certificates to subjects automatically, establish common trusted certification authorities, and manage recovery policies for EFS.

Trusts

The Windows Server 2003 family supports domain trusts and forest trusts. Domain trust allows a user to authenticate to resources in another domain. To establish and manage domain trust relationships, you must take into consideration trust direction.

Trust Direction

The trust type and its assigned direction will have a substantial impact on the trust path used for authentication. A trust path is a series of trust relationships that authentication requests must follow between domains.

Before a user can access a resource in another domain, the security system on domain controllers running Windows Server 2003 must determine whether the trusting domain (the domain containing the resource the user is trying to access) has a trust relationship with the trusted domain (the user's logon domain). To determine this, the security system computes the trust path between a domain controller in the trusting domain and a domain controller in the trusted domain. In Figure 5-5, trust paths are indicated by arrows showing the direction of the trust.

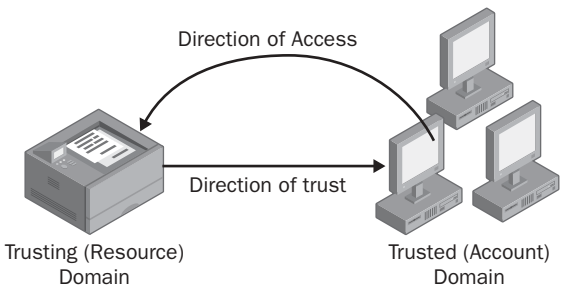


Figure 5-5 This diagram shows trust paths and the direction of each trust.

All domain trust relationships have only two domains in the relationship: the trusting domain and the trusted domain.

Trust Types

Communication between domains occurs through trusts. Trusts are authentication pipelines that must be present for users in one domain to access resources in another domain.

- **One-way trust.** A one-way trust is a unidirectional authentication path created between two domains. This means that in a one-way trust between domain A and domain B, users in domain A can access resources in domain B. However, users in domain B cannot access resources in domain A. Some one-way relationships can be nontransitive or transitive depending on the type of trust being created:
  - A transitive trust flows throughout a set of domains, such as a domain tree, and forms a relationship between a domain and all domains that trust that domain. For example, if domain A trusts domain B and domain B trusts domain C, domain A trusts domain C. Transitive trusts can be one-way or two-way, and they are required for Kerberos-based authentication and Active Directory replication.
  - A nontransitive trust is restricted to two domains in a trust relationship. For example, even if domain A trusts domain B and domain B trusts domain C, there is no trust relationship between domain A and domain C. Nontransitive trusts can be one-way or two-way.
- **Two-way trust.** All domain trusts in a Windows .NET forest are two-way transitive trusts. When a new child domain is created, a two-way transitive trust is automatically created between the new child domain and the parent domain. In a two-way trust, domain A trusts domain B and domain B trusts domain A. This means that authentication requests can be passed between the two domains in both directions. Some two-way relationships can be nontransitive or transitive depending on the type of trust being created.

Trust Relationships

A Windows .NET domain can establish a one-way or two-way trust with

- Windows .NET domains in the same forest.
- Windows .NET domains in a different forest.
- Windows NT 4.0 domains.
- Kerberos V5 realms.

Forest Trusts

In a Windows Server 2003 forest, administrators can create a forest trust to extend two-way transitivity beyond the scope of a single forest to a second Windows Server 2003 forest. In other words, with forest trusts you can link two disjoined Windows Server 2003 forests to form a two-way transitive trust relationship between every domain in both forests. Forest trusts provide the following benefits:

- Simplified management of resources across two Windows Server 2003 forests. Forest trusts reduce the number of external trusts needed to share resources with a second forest.
- Complete two-way trust relationships with every domain in each forest.
- Wider scope of UPN authentications. User principal name authentications can be used across two forests.
- Greater trustworthiness of authorization data. Both the Kerberos and NTLM authentication protocols can be used to help improve the trustworthiness of authorization data transferred between forests.
- Flexibility of administration. Administrators can choose to split collaborative delegation efforts with other administrators into forestwide administrative units.
- Isolation of directory replication within each forest. Schema changes, configuration changes, and the addition of new domains to a forest have forestwide impact only within that forest, not on a trusting forest.

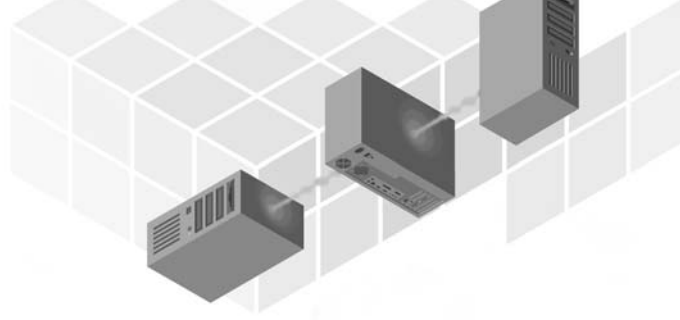
Forest trusts can be created only between two forests and therefore will not be implicitly extended to a third forest. This means that if a forest trust is created between Forest1 and Forest2, and a forest trust is also created between Forest2 and Forest3, Forest1 will not have an implicit trust with Forest3.

> **Note** In Windows 2000, if users in one forest needed access to resources in a second forest, an administrator could create an external trust relationship between the two domains. External trusts are one-way and nontransitive and therefore limit the ability for trust paths to extend to other domains only when explicitly configured.

For More Information

See the following resources for more information:

- What's New in Internet Information Services 6.0 at *http://www.microsoft.com/windowsserver2003/evaluation/overview/technologies/iis.mspx*
- Windows 2000 Security Services at *http://www.microsoft.com/windows2000/technologies/security/*
- What's New in Security for Windows XP at *http://www.microsoft.com/windowsxp/pro/techinfo/planning/security/whatsnew/*
- PKI Enhancements in Windows XP Professional and Windows .NET Server at *http://www.microsoft.com/windowsxp/pro/techinfo/planning/pkiwinxp/*
- Data protection and recovery in Windows XP at *http://www.microsoft.com/windowsxp/pro/techinfo/administration/recovery/*
- Securing Mobile Computers with Windows XP Professional at *http://www.microsoft.com/windowsxp/pro/techinfo/administration/mobile/*
- Wireless 802.11 Security with Windows XP at *http://www.microsoft.com/WindowsXP/pro/techinfo/administration/wirelesssecurity/*
- Institute of Electrical and Electronics Engineers at *http://www.ieee.org/*



18

Testing for Application Compatibility

There are many concerns during the deployment of a new operating system. If you are considering deploying the Microsoft Windows Server 2003 family and Microsoft Windows XP, you're probably concerned about how well these operating systems will support the applications that actually run your day-to-day business. Addressing these concerns will occupy much of the planning and testing phases of your deployment project.

The Application Compatibility Toolkit (ACT) contains several tools that make this process easier to manage. You can download this tool from Microsoft's Web site at *http://www.microsoft.com/downloads/release.asp?releaseid=42071*. The ACT provides tools to test applications both during the development phase and during deployments. It also provides tools that allow you to gather data about the applications installed on every Windows computer on the network and to package the necessary compatibility fixes for each of those computers. Those tools include the following:

- **Analyzer.** This tool gathers information about every program installed on your network. The analyzer can be used to automate the process of creating an inventory of the software used in your enterprise.
- **Application Verifier.** This tool assists developers looking for compatibility issues with a new application. It's also possible for information technology (IT) professionals to use this tool to determine whether a proposed software package has any common compatibility issues.

- **Compatibility Administrator.** This tool determines the necessary compatibility fixes to support an application in Windows. This tool can also package fixes into a custom compatibility database that can then be distributed to computers on the network.

In addition to describing these tools, this chapter describes how to collect an application inventory. It also shows how to test applications for compatibility, how to create fixes for application compatibility, and how to distribute those fixes. Last it includes a checklist you can use during compatibility testing. For more information about the tools mentioned in this chapter, including more detailed documentation, see *http://www.microsoft.com/windowsserver2003/compatible/appcompat.mspx*.

Collecting an Application Inventory

Before testing for application compatibility can begin, you need to know and understand which applications are present in your environment. Many organizations will miss the vital nature of this inventory by assuming that they already have a list of every application approved for use. This does not take into account limited-use applications for special projects within the organization, nor does it include nonapproved software that is inevitably present. The need for a proper software inventory then becomes clearer.

There are multiple approaches to the problem of creating a software inventory. Many of those methods are beyond the scope of this book. Microsoft currently offers two methods for collecting a software inventory: Systems Management Server (SMS) and the Analyzer that ships as part of the Application Compatibility Toolkit.

The Compatibility Analyzer tool collects application information from computers, along with identifying machine information, and writes it to log files in XML format. Compatibility Analyzer then consolidates the log files into a database in a central location, from where you can analyze the applications for compatibility status as well as review reports. Compatibility Analyzer comprises three distinct parts:

- **Collector.** Collector is the first part to run. Collector is a command-line tool that runs quietly in the background without interrupting the user while it collects data about every application on the computer. It then records the data in a log file in a specified location. (It defaults to the user's desktop but can be directed to a network share for central collection.)

- **Merger.** Merger (Merger.exe) combines the various collected log files into a single database file. By default, Merger enters the data in a Microsoft Access database file (.MDB), but the logs can also be sent to a SQL database.
- **Analyzer.** Analyzer is the graphical workspace for viewing the collected data and generating meaningful reports from the data.

Collecting Information

You collect application information with Compatibility Analyzer by distributing and running a command-line tool (Collector.exe) on the computers where you want to inventory applications. You can configure this tool to define the scope of the inventory: You can specify which drives, either network or local, and which paths to search and whether to collect device information. You can also specify where you want the logs to be saved. You can collect inventory information on the following platforms:

- Windows 98 clients
- Windows Me clients
- Windows NT 4.0 servers and clients
- Windows 2000 servers and clients
- Windows XP clients
- Windows Server 2003 family servers
- Mixed domains of clients, including any of Windows 98, Windows Me, Windows NT 4.0, Windows 2000, or Windows XP
- Mixed domains of servers, including any of Windows NT 4.0, Windows 2000, or Windows Server 2003 family

Collector detects the client operating system when it starts and loads the appropriate support. For example, the native character type on Windows 98 or Windows Me is ANSI, so Collector would load ANSI support to store the log information. On Windows NT or Windows XP, Collector would load Unicode support to store its data.

The most important function of Compatibility Analyzer is the collection of application compatibility information from client computers. In fact, each successive step in the process assumes that you've already gathered this data from at least one client computer into a log file. The performance of Collector can be customized through the use of command-line switches. The following shows the syntax of the Collector command:

```
collector.exe [-o filename] [-f source] [-e department]
              [-n] [-d days] [-a] [-p profile]
```

-o *filename*	Directs Collector to produce output on the specified path. By default, Collector places output file on the user's desktop.
-f *source*	Provides the source path, either a file or a directory, for Collector to gather information from. If a file or directory is not specified, directs Collector to gather information from all drives on the machine.
-e *department*	Provides department information for use in processing collector logs. This data helps to separate collected information into useful categories once the logs are merged later in the process.
-n	Directs Collector not to collect information from mapped (network) drives. By default, network drives are included.
-d *days*	Directs Collector to collect information only if Collector has not run within the number of days specified by the parameter; if the number of days is not specified, Collector will not run if it has already been executed on the machine once.
-a	Collects information from the shell and installed programs and combines it with information from specified drives and paths.
-p *profile*	Directs Collector to use a specified profile (initialization file).

Reporting Information

The analysis component of Compatibility Analyzer runs on the administrator's computer, where all operations are sent and received. From here, you can analyze compatibility information and generate reports. This component consolidates all the logs into a database, combining identical application information into one record. You can use an ODBC SQL database or an Access database.

You can analyze application compatibility and generate reports on the following platforms, all of which must be running Internet Explorer 5.0 or later:

- Windows NT 4.0 servers or clients
- Windows 2000 servers or clients
- Windows XP clients
- Windows Server 2003 family servers

Here's an overview of using Compatibility Analyzer:

1. Install the analysis component on the administrator's computer where you want to review reports.
2. Define the analysis database, either as an ODBC SQL database or as an Access database.

3. Configure the collection component to define the scope of inventory and the location of the logs.
4. Distribute the collection component to the computers where inventory information is to be collected, and run it. This component does not need to run under an administrator account. You can distribute the component in the following ways:
   - ❑ Floppy disk
   - ❑ CD-ROM
   - ❑ Logon scripts
   - ❑ Group Policy in an Active Directory environment
   - ❑ Hyperlink in e-mail
   - ❑ Network distribution share
   - ❑ SMS
5. Consolidate the log files into a database.
6. Analyze the compatibility status.
7. Review the reports.

   You can review reports by application or by computer, and you can filter and sort the results. When viewing reports by computer, you can see all the applications installed on a specific computer. When viewing reports by application, you can see how many instances of the application are installed on the network.

As you make your test plan, you will want to focus most of your efforts on the applications that are installed on many computers and the ones that are incompatible or whose compatibility status is unknown.

Testing for Compatibility

Once the software inventory has been created and verified, you can formulate a test plan. A valid software test plan for the deployment of a new operating system must include basic details, such as whether an application will run on the new operating system, as well as more complex testing that includes combinations of the applications found in the organization.

This section describes the strategies for testing applications during a Windows deployment. It also provides information regarding the tools in the Application Compatibility Toolkit and how they can help during this phase of your deployment.

The applications to be tested for a Windows deployment should include every program being used within your organization, both desktop and server. Organizations that have standardized on a set of approved applications might find this task somewhat easier than those who have no standardization at all. Once you have gathered the information in your software inventory and analyzed the business priority of each application, the test plan should be formulated.

In a perfect world, you would test every application present in the organization for compatibility with the new Windows operating systems being deployed. Very few IT departments have this luxury both in time and in budget. By assigning a priority to each application in the organization, you can make intelligent choices about where to spend your testing time. Priorities for applications should be assigned according to their relevance to daily business functions. A desktop application that is used occasionally would have a much lower priority than a client-server application that manages the main product of your organization. A suggested priority scale is as follows:

1. **Business-critical.** Applications in this category are absolutely required for business to be performed. Business-critical programs cannot endure downtime without a significant loss in revenue.
2. **Business-function.** This category includes applications that are used by a majority of users within the organization for their daily work. An example of a business-function application would be Microsoft Word 2002 if most people in the organization use it for daily business. Some downtime or problems can be tolerated, but the core functionality of the application must be ensured.
3. **Specialty.** An application in this category would be important to a very small segment of the user population within an organization and not tied directly to the continued success of the organization. An example would be an art program used to process photographs for marketing materials, important to a small segment of users but not impossible to live without if there were a serious compatibility issue.
4. **Other.** Applications in this last category include nonstandard programs that users have installed on their own. Typically, these programs have little or no bearing on the daily business of the organization, and application compatibility issues here will have no impact on the business.

Microsoft provides the "Designed for Windows XP" Application Test Framework document to describe a suggested set of test cases and procedures, including test lab setup, for evaluating whether an application meets the

requirements for the Designed for Windows XP logo. While you might not be concerned about logo requirements of an application during a Windows deployment, this document does provide excellent insight into testing procedures that have been designed to test for Windows application compatibility.

The Test Framework assumes that the tester has the skills to create an effective test plan, run the tests, and evaluate the results. These skills include the following:

- Software testing experience and familiarity with creating test plans
- Experience testing Windows-based desktop applications
- Experience installing and configuring computer hardware
- Familiarity with Windows operating systems
- Ability to install and configure Windows XP and Windows Server 2003 options
- Experience monitoring test applications using kernel-mode debuggers

The depth of testing described in the Test Framework is beyond most IT professionals; still, the benefit of this document is in the description of testing environment and techniques.

Gathering Information About Applications

As mentioned earlier in this chapter, the Analyzer can be used to collect a list of the applications in use within your organization. However, it can't tell you the business purpose of each program. To obtain this information, you must conduct interviews with users that represent a wide variety of environments within your organization. As a general rule of thumb, try to involve the following groups when collecting information about the applications in use in your organization:

- **Management.** Top-level management might have specific input regarding the applications in use. Also, collect data from the management of individual departments in the organization.
- **Information technology.** The IT department in the organization is in a unique position to understand which applications are actually used on a day-to-day basis within the organization.
- **Users.** This group is often overlooked or underutilized during the interview phase. Try to gather information from representative groups of users from various parts of the organization to assemble a cross-section view of what is actually useful to them on a daily basis.

It's entirely possible that applications on the list of installed software are not even in current use. These, then, can be safely listed as low priority for compatibility with the new operating system. This is actually the reason for gathering application usage data. It enables you to set priorities for testing and helps to reduce or eliminate concerns about applications that have compatibility issues with the new operating system, if in fact they are not used very much. A suggested priority scale would be the following:

1. **Business-critical.** Applications that are vital to daily business and cannot tolerate any downtime without adversely affecting the organization. An example would be the electronic commerce application for an online merchant. Without the ability to process new orders, the business is down.
2. **Daily-use.** This category describes the applications that are used on a daily basis by a majority of users in the organization but can tolerate some downtime. A failure of an application in this category would be irritating, but business would still be conducted with only minimal interruption.
3. **Minimal-use.** The final category describes applications that are in use but not essential to business. You might find it necessary to divide this category into subcategories to deal with applications that are infrequently used but still quite important to the organization, and those applications that are simply nice to have on your computer.

By carefully reviewing the usage patterns of the users within your organization, you can adjust the time allocation for application testing to focus primarily on those programs that are most important to the organization. You can also reduce or eliminate the time spent testing applications that are not important to the organization.

Using Compatibility Administrator

Compatibility Administrator can be a useful tool during the latter portions of your testing schedule. This tool will not help you to automate your software testing, nor will it find compatibility issues in your programs. What Compatibility Administrator can do is help to identify possible solutions to the compatibility issues raised by your software testing.

As the testing process identifies areas of concern with the current or planned applications in your organization, Compatibility Administrator can be used to identify possible solutions for those issues. Consider the following example:

- **Application.** MyApp16, a 16-bit Windows-based application used for entering customer service call data and for tracking incidents.
- **Business impact.** Daily-use application, used by the customer service department and used to handle all of its incoming customer call data. Can tolerate some downtime but is important to the daily workflow.
- **Compatibility Issue.** MyApp16 was designed to run on Windows 3.1. It performed adequately when the organization upgraded to Windows 95 but does not run at all on Windows XP Professional or Windows Server 2003.

With an example application like MyApp16, you might be tempted to recommend upgrading to a newer, 32-bit, application that would accomplish the same tasks. Imagine that the budget constraints for the planned deployment of Windows XP Professional to all of the desktops in the organization preclude the simultaneous upgrade of the customer service application. The task then becomes finding a solution that will enable the application to function on Windows XP. Compatibility Administrator can be used to test possible solutions for the problems this application encounters on Windows XP.

Creating Compatibility Fixes

Independent software vendors (ISVs) have long made it a practice to write their products to run as well as possible on the customer's computer. To accomplish this end, they have looked for ways to work with the operating system to perform their tasks in the most efficient ways possible. The result is an application that is highly optimized for the version of Windows that it was originally written for. Application compatibility issues can arise when a customer tries to run a favorite program on a newer version of Windows than the application was originally written for. This might be particularly true with the move to Windows XP because it's built upon the foundation of Windows NT and Windows 2000.

Many ISVs have been developing applications for the home user market, and for years that focus has meant supporting Windows. Windows NT and Windows 2000 have been seen as business operating systems, so some application developers have chosen to write their programs solely for Windows 95 and Windows 98. When applications are moved to Windows XP, they will encounter new ways of doing familiar tasks. Some of these differences will be a result of the new features of Windows XP, but some will be a result of the more stringent rules laid down by the Windows NT heritage of Windows XP.

Applications that worked on other versions of Windows might fail to function properly on Windows XP. This applies mostly to applications written for Windows 95 and Windows 98, but applications written for Windows NT or Windows 2000 might also be affected. This can happen for any of the following reasons:

- The application refuses to run when Windows reports new, higher version numbers. Often the application will work well on the new version of Windows if the user can get past this block in the application.
- The application calls older versions of Win32 API functions that return unexpected values on machines with large amounts of resources such as disk free space.
- The application expects older formats of Windows data.
- The application expects user information, such as personal and temporary folders, to be in specific locations or specific formats.

Understanding the Application Compatibility Process

Windows XP and Windows Server 2003 introduced a new level of commitment to application compatibility. There are methods within the user interface to correct application compatibility issues, as well as more advanced tools meant for developers or administrators. All of these technologies depend on compatibility information gleaned from the files on the local computer that are then matched to data in a system-level compatibility database. This matching information is used to uniquely identify an application that might require additional support from the operating system to function correctly.

Three fundamental levels of compatibility can be applied, depending on the scenario:

- **Compatibility fix.** This level of compatibility support provides a *shim*, or small piece of code that corrects a particular behavior that results in a compatibility issue. Typically, a compatibility fix addresses only one compatibility issue.
- **Compatibility mode.** A compatibility mode is a collection of several compatibility fixes that are commonly applied together. An example would be the application of the Windows 98 compatibility layer through the user interface. The Windows 98 compatibility layer is a compatibility mode; that is, it comprises multiple compatibility fixes that are required to support most applications written specifically for Windows 98.

- **AppHelp.** An AppHelp message is the final option when a compatibility issue cannot be completely resolved. Simply put, an AppHelp message is a message that will be triggered whenever you attempt to run an application with a known unresolved compatibility issue. AppHelp messages range from advisory in nature, wherein the user is simply notified that the program might not support all of its features on the operating system, to a full block, wherein the application is completely blocked from running. The blocking AppHelp message is used only in a situation in which running the application would cause damage to the operating system, such as running a disk utility that was not written to handle the current operating system. Frequently the text of an AppHelp message will direct the user to a Web address where more information and possibly a fix can be found.

The compatibility fix technologies used in Windows XP and Windows Server 2003 are dependent on several database files:

- **MigDB.inf** This file is used to support the migration from the Windows 95, Windows 98, and Windows Me operating systems. It contains the matching information used to flag applications that are incompatible or require user intervention prior to system upgrade. Problematic applications are listed along with hardware compatibility information in the upgrade report generated by Setup. This file was first included as part of Windows 2000 Setup and has now been updated to run as part of the Windows XP and Windows Server 2003 Setup programs.
- **NTCompat.inf** This database contains the same type of information as MigDB.inf but is used to support upgrades from the Windows NT 4.0 and Windows 2000 operating systems. This file is also included in the Windows XP and Windows Server 2003 Setup programs.
- **SysMain.sdb** This file contains both matching information and compatibility fixes. SysMain.sdb contains the information used to provide compatibility fixes for applications that require some help to run correctly on Windows XP and Windows Server 2003. It's in the %windir%\AppPatch folder.
- **AppHelp.sdb** This database stores only the Help messages that prompt users for patches, provide them with a URL from which to download third-party patches, or tell them where to find further information. This file is also found in the %windir%\AppPatch directory.

These database files form the core functionality of the compatibility technologies in Windows XP and Windows Server 2003. The operating system itself contains no code to verify the compatibility of any software but rather depends on a short system check routine to tell the operating system to refer to the database files for compatibility information. This approach has the effect of giving a high level of support for applications while minimizing the performance impact on the operating system itself.

Creating Compatibility Fixes

Once you have identified and tested fixes for your target applications, you can use Compatibility Administrator to create a custom fix database. Figure 18-1 shows what Compatibility Administrator looks like. You can create a custom fix database that contains applications supported by compatibility layers, as well as applications supported by specific compatibility fixes.

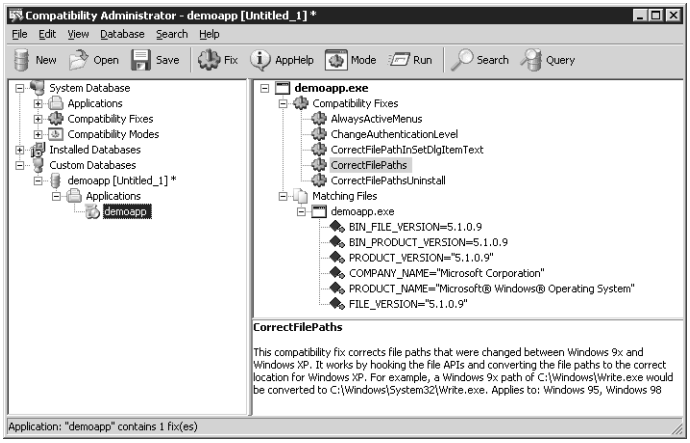


Figure 18-1 Use Compatibility Administrator to build custom fix databases for applications that don't work properly in Windows XP and Windows Server 2003.

To add compatibility fixes to a custom fix database, you click Create New, Application Fix on the Database menu. The program asks you for the name and filename of the program you're fixing. It prompts you for a compatibility mode as well as individual compatibility fixes that you want to apply to the application. Last it asks you for a list of files that identify the application on target computers. Choose files associated with your application that are installed in the same location. For example, choose a .hlp file that resides in the same directory

as the .exe file. Try to uniquely identify your application without choosing an unnecessarily high number of matching files.

The application compatibility technology in Windows XP and Windows Server 2003 provides a way to distinguish files with the same or similar names. The operating system does this through the use of file matching information. If you were creating a compatibility fix for a Setup.exe but did not want this compatibility fix used every time you ran a file named Setup.exe, you would specify a group of other files belonging to the application. By gathering data about the specific properties of these files, the operating system can uniquely identify the application requiring the compatibility fix wherever it exists on the computer.

Distributing Compatibility Fixes

Once the compatibility database containing your compatibility fixes has been delivered to the client computers, it must be installed there. There are a few different methods that you can use, and these are described next. Each method relies on the Compatibility Database Installer (Sdbinst.exe) to install and register the compatibility database.

Local Installation

The simplest method of installing a compatibility database is to perform a local installation using Sdbinst.exe. The following shows the syntax of the Sdbinst.exe command:

```
sdbinst.exe [-?] [-q] [-u] [-g] [-n] database | {GUID} | name
```

-?	Displays the help for the command.
-q	Quiet mode; does not display message boxes during the process.
-u	Uninstalls the named database. (The compatibility database to be uninstalled must be identified in some way—as a filename, a GUID, or an internal name.)
-g	Uses the GUID of the compatibility database to identify the information to be uninstalled.
-n *name*	Specifies the internal name of the compatibility database. This is the name assigned internally for the database when it was created in Compatibility Administrator.

Remote Installation

Installing the compatibility database file on a remote computer involves some method of running Sdbinst.exe on the remote computer. Typically, this will be accomplished through a logon script, although other methods such as Remote Desktop Connection (RDC) can be used as well. The benefit of using a logon script to distribute the compatibility database file is that you can use Active Directory to target the users that will need the compatibility database and not install it where it will not be required. Here's how to install a database remotely:

1. Create a shared folder on a server that will be accessible by all client computers requiring the compatibility database. Set permissions to Read for all users that will be receiving the file.
2. Edit the logon script for some level of Active Directory (site, domain, or OU) where the compatibility database will be required, and add a line to install the database:

   ```
   Sdbinst.exe \\servername\sharename\database.sdb -q
   ```

3. Test the script to verify that it installs the compatibility database correctly.
4. Assign the logon script to the group that requires the database (site, domain, or OU). If you are modifying an existing script, this step will not be required.

Compatibility Testing During Development

During a deployment of Windows within a large organization, there might be concerns about the compatibility of in-house applications. These programs are frequently very important to the flow of normal business and critical in the deployment process. The testing ideas presented in this section are useful for verifying these internal applications, but they can also be used as part of a software evaluation process when considering new programs for the organization.

Testing applications for compatibility with any operating system can be a complicated process. Fortunately, the Application Verifier (AppVerifier) tool developed by the Windows Application and Customer Experience group at Microsoft can assist with this task. AppVerifier encompasses several tools that are specifically designed to test for commonly encountered application compatibility issues and some very subtle kernel issues. These tests can also reveal compatibility issues related to the requirements found in the Designed for Windows XP Application Specification.

AppVerifier is a useful tool for identifying some of the common security issues that applications might create in the Windows operating system, such as writing information to incorrect locations within the registry or the file system, where the information can later be modified by a malicious program. Using the Application Verifier will not prevent every possible security or compatibility issue, but it does provide an easy opportunity to avoid and correct the most commonly identified problems.

Using Application Verifier

Perhaps the first thing to note about AppVerifier is that it is not an automated test program for your applications. AppVerifier will attach to a program and perform its tests whenever you run the program. It's possible to use AppVerifier and an automated test procedure simultaneously. AppVerifier attaches a *stub*, or small piece of code, to the executable program you are testing so that anytime it's run, the AppVerifier tests you have selected will be engaged.

Using AppVerifier to test an application is a relatively simple process. You choose the program files that you want to test. Then choose the tests you want to perform. Some of the tests available to you are the following:

- **Heap corruption detection.** This test performs regular checks of the heap and adds guard pages at the end of each allocation to catch possible heap overruns.
- **Locks usage checking.** This test looks for common errors with locks. The output is displayed in a separate debugger application. Note that this test can cause access violations if an error is found.
- **Invalid handle usage detection.** Checks for common problems with handles. The output is displayed in a separate debugger application. Note that this test can cause access violations if an error is found.
- **Thread stack size checking.** This test disables stack growth. This will cause a stack overflow exception if the initial allocation was too small.
- **LogStartAndStop.** This option simply enters log information when the application starts or stops. This helps to make the logs easier to read when reviewing test data.
- **FilePaths.** This test monitors the application's attempts to obtain file path information to determine whether the program uses hard-coded paths or a nonstandard method of gathering the information. Note that this test can cause the application to crash if an improper method of determining file paths is used.

- **HighVersionLie.** In the past, many applications were written to run on a single version of Windows. This test will return a very high version number when the application attempts to determine which version of Windows it's running in.
- **RegistryChecks.** This test monitors the application's use of the system registry for any inappropriate or dangerous calls. Any problems detected will be logged.

After choosing the tests you want to perform, click Options to configure your tests, as shown in Figure 18-2. Then start the application by clicking the Run button in AppVerifier or by starting the program normally. Exercise the application by trying to use all of the functionality in the program to generate the best data for the AppVerifier logs. After closing the application, view the test results in the AppVerifier log file: click View Logs.

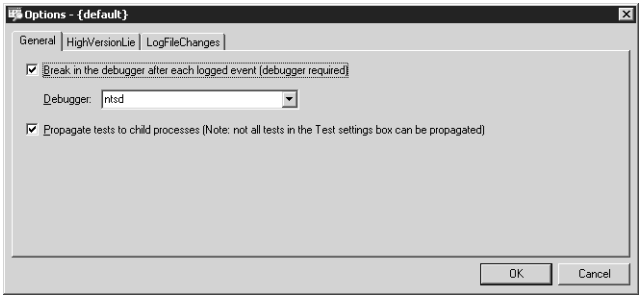


Figure 18-2 Select the options you want to use for testing the application.

The test settings you specify in AppVerifier for a particular application will remain active anytime you run the program until it is removed from the list of applications in AppVerifier. This helps to run programs repeatedly while working out issues.

The first four tests in AppVerifier look for issues that might be found at the kernel level. Because of this, the best output from these tests can be acquired only with the use of a separate kernel debugger. The kernel tests are designed to generate access violation errors when they encounter an error in the program being tested so that the kernel debugger will break in at precisely that point in the application's execution. If you run an application through AppVerifier without a debugger attached and one of the kernel tests finds an error, the application will appear to crash.

To run the app using a debugger, just set all the options and tests desired in AppVerifier and then launch the application with a debugger according to the directions for that debugger. For example, to debug Myapp.exe with NTSD

(the Windows XP system debugger), go to a command line and type **ntsd myapp.exe**.

Any debugger can be used. The assumption is that the user running the tests is familiar with using a debugger. If you are not comfortable with using a debugger, you should have problems investigated by an experienced developer, who can then run the application with a debugger.

Testing for Logo Compliance

The Designed for Windows logo program identifies products that have been proven to maintain a high level of compatibility with Microsoft Windows. Application Verifier contains several tests that directly relate to the Designed for Windows logo program to make testing easier for every independent software vendor planning to submit a product for the logo. These tests are identified in the user interface by a number at the end of the test name, as shown in Figure 18-3.

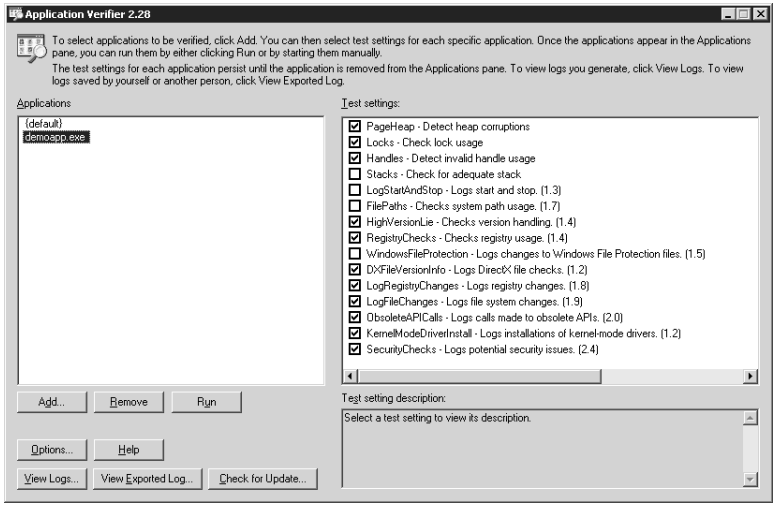


Figure 18-3 You can use Application Verifier to test for Windows logo compliance.

The numbering used in the interface indicates the specific requirement within the Designed for Windows XP Application Specification that the test setting refers to. For example, the WindowsFileProtection (1.5) test applies to section 1, requirement 5, of the Designed for Windows XP Application Specification, Support Fast User Switching and Remote Desktop, because correct use of system paths is one step toward supporting fast user switching in Windows XP and later.

Using Application Verifier during the development of applications destined for the Designed for Windows logo is strongly suggested. This tool is able to detect approximately 90 percent of the problems that Microsoft finds in products that fail the Designed for Windows logo auditing process. Using Application Verifier as a normal step in your development cycle means that you have eliminated the majority of issues that can block your product from receiving the logo. It also helps to ensure a high-quality user experience for your customers.

Application Compatibility Checklist

This section outlines a series of tests taken from the Windows XP Logo Test Framework that you can use to evaluate your application. If you are interested in applying for the Designed for Windows XP logo for your application, go to *http://www.windowslogo.com/*.

- Does the application perform its primary functions and maintain stability during functionality testing?
- Does the application remain stable when a mouse with more than three buttons is used?
- Does the application use the user's temporary folder for temporary files?
- Does the application store its temporary files only in the user's temporary folder during installation?
- Does the application store its temporary files only in the user's temporary folder during functionality testing?
- Does the application not crash or lose data when presented with long pathnames, filenames, and printer names?
- Does the application maintain stability when a file is saved by drilling down through the User1 LFNPath1 path in User1's My Documents folder?
- Does the application maintain stability when a file is saved by entering the full User1 LFNPath2 path?
- Does the application maintain stability when a file is saved using a long filename?
- Does the application maintain stability when a file is opened by drilling down through the User1 LFNPath1 path in User1's My Documents folder?

- Does the application maintain stability when a file is opened by entering the full User1 LFNPath2 path?
- Does the application maintain stability when a file is opened using a long filename?
- Does the application maintain stability when printing to a printer with a long name?
- Does the application perform primary functionality and maintain stability on a dual-processor computer?
- Does the application not crash when devices it uses are not installed?
- Does the application maintain stability when printing if no printer is installed?
- Does the application maintain stability when attempting to use devices that are not installed?
- Does the application switch the system's display mode back to the previous color mode if the application automatically changes to 256-color mode when it runs?
- Do all related kernel-mode drivers pass testing as Windows XP loads them?
- Do all related kernel-mode drivers pass functionality testing with standard kernel testing enabled?
- Do all related kernel-mode drivers pass low-resources simulation testing?
- Are proofs of Windows Hardware Quality Labs (WHQL) testing attached to the submission for all required drivers?
- Do no warnings appear about unsigned drivers during testing?
- Does the application install correctly under current and future versions of Windows?
- Does the application perform all functionality tests correctly under current and future versions of Windows?
- Does the application properly support Fast User Switching?
- Does the application properly support Remote Desktop?
- If the application installs a replacement Graphical Identification and Authentication (GINA) DLL, does the GINA properly support Remote Desktop?

- Does the application pass all functionality tests with a Windows XP theme applied?
- Does the application display normally and not lose data when focus is switched among other applications with Alt+Tab?
- Does the application display normally and not lose data when the Windows logo key and the taskbar are used to switch among applications?
- Does the Windows Security dialog box or the Task Manager display normally, and can the application be canceled or closed without losing data?
- Does the installation finish without any Windows File Protection messages appearing?
- Does the application successfully migrate from Windows 98 to Windows XP Home Edition?
- Does the application successfully migrate from Windows Me to Windows XP Home Edition?
- Does the application successfully migrate from Windows 98 to Windows XP Professional?
- Does the application successfully migrate from Windows Me to Windows XP Professional?
- Does the application successfully migrate from Windows NT 4.0 Workstation to Windows XP Professional?
- Does the application successfully migrate from Windows 2000 Professional to Windows XP Professional?
- Does the application not overwrite nonproprietary files with older versions?
- Do all application executable files have file version, product name, and company name information?
- Does the installation finish without requiring a reboot?
- Can all Test Framework testing be completed without the application requiring a reboot?
- Does the application offer a default installation folder under C:\Program Files?
- Does the application install shared files only in correct locations?

- Does installation add all necessary entries to the registry?
- Does uninstalling the application as Owner remove and leave all the correct files and registry settings?
- Does uninstalling the application as User1 either degrade gracefully or both remove and leave all the correct files and registry settings?
- Can the application be reinstalled after uninstalling it?
- Does the application default to an "all users" installation or provide an "all users" installation option when installed by Owner?
- Does the application default to an "all users" installation or provide an "all users" installation option when installed by User1?
- Does the application's installer start by way of Autorun?
- Does the application's installer correctly detect that the application is already installed and avoid restarting the installation?
- Does the application offer a correct location for opening User1's user-created data?
- Does the application offer a correct location for saving User1's user-created data?
- Does the application offer a correct location for opening User2's user-created data?
- Does the application offer a correct location for saving User2's user-created data?
- Does the application store less than 128 KB of application data in the registry for User1?
- Does the application store configuration data for User1 only in acceptable folders?
- Does the application prevent User1 from saving to the Windows system folder?
- Does the application prevent User1 from modifying documents owned by User2?
- Does the application prevent User1 from modifying systemwide settings?
- Does the application's installer either allow User1 to install the application or degrade gracefully if the installation fails?

For More Information

See the following resources for further information:

- Using the Application Compatibility Toolkit at *http://www.microsoft.com/windowsserver2003/compatible/appcompat.mspx*.
- Windows Application Compatibility Toolkit download at *http://www.microsoft.com/downloads/release.asp?releaseid=42071*.