



realtimepublishers.comtm

Tips and Tricks Guidetm To

Securing .NET Server

Roberta Bragg

Note to Reader: This book presents tips and tricks for eight Windows .NET Server security topics. For ease of use, the questions and their solutions are divided into chapters based on topic, and each question is numbered based on the chapter, including:

- Chapter 1: Understanding and Utilizing PKI in .NET
- Chapter 2: Securing Web Services and Web Servers—the Administrative Perspective
- Chapter 3: Understanding Active Directory Foundations
- Chapter 4: Fulfilling the Promises of Group Policy
- Chapter 5: Administrative Authority
- Chapter 6: Triple A’s—Authentication, Authorization, and Audit
- Chapter 7: Remote Access
- Chapter 8: Security Tools, Mechanisms, and Emerging Issues.

Chapter 1: Understanding and Utilizing PKI in .NET 1

Q 1.2: Developing a public key infrastructure using Windows Certificate Services seemed like the way to go until I discovered that there is no auto enrollment. Do you expect that users will be able to obtain certificates on their own? 1

 Quick Steps to Auto Enrollment 2

Chapter 2: Securing Web Services and Web Servers—the Administrative Perspective 9

Q 2.2: I keep hearing about the new NetworkService account in Windows .NET Server. Microsoft says that this account is less privileged than the Local System account but doesn’t explain what that means. Any ideas? 9

 What Can the NetworkService Account Do? 10

 What Can the Local System Account Not Do? 10

Chapter 3: Understanding Active Directory Foundations 12

Q 3.2: Help! We have a large, multidomain forest that contains thousands of users, and we use universal groups extensively to provide access to resources. We also have many small branch offices that have only a handful of users each. Because the Global Catalog server must be accessed at logon to determine membership in universal groups, all users at branch offices access Global Catalogs across the WAN even though they have a domain controller locally. If I make the domain controllers at these branch offices Global Catalog servers, I’ve increased the replication traffic by an unacceptable amount. Does Windows .NET Server have an answer?.... 12

 Win2K Branch Office Solutions 14

 .NET Universal Group Membership Caching 16

Chapter 4: Fulfilling the Promises of Group Policy 18

Q 4.2: How can I prevent users from running tools that they should not run? 18

 Setting Restrictive File Permissions 18

 Removing Executables 18

 Using Software Restriction Policies 18

A Simple Policy to Restrict Tool Use.....	19
Creating a Simple Software Restriction Policy	20
Testing the Policy	22
Examining the Policy for Holes.....	22
Certificate Rules.....	23
Trusted Publishers.....	24
Moving On.....	24
Chapter 5: Administrative Authority	26
Q 5.2: I seem to have locked out the Administrator account in the domain. I can no longer log on to the domain using this account. What should I do?	26
Chapter 6: Triple A's—Authentication, Authorization, and Audit	28
Q 6.2: I am tasked with tracing logon behavior across our Windows .NET forest and would like some help understanding the difference between the Audit categories Audit Account Logon and Audit Logon. What is the difference?.....	28
Logon Events	29
Tracing a User's Logon and Logoff Events.....	30
Chapter 7: Remote Access.....	34
Q 7.2: What is the Session Initiation Protocol?.....	34
Internet Engineering Task Force RFC SIP Security	35
Message Integrity, Authorization, and Authentication.....	35
Message Privacy	35
Route Privacy.....	36
Identity Privacy.....	37
Miscellaneous Uses of Cryptography	37
Microsoft Implementation	38
Chapter 8: Security Tools, Mechanisms, and Emerging Issues.....	40
Q 8.2: Managing multiple Windows boxes is like herding cats. It's hard to keep up with which systems have up-to-date patches and which still need them. How can I figure this out?.....	40
Analyzing the Analyzer: MBSA Internals.....	40
Understanding Prerequisites	41
Getting Secure the MBSA Way.....	42
Now What?	44

Copyright Statement

© 2002 Realtimepublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimepublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimepublishers.com, Inc or its web site sponsors. In no event shall Realtimepublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimepublishers.com, please contact us via e-mail at info@realtimepublishers.com.

Chapter 1: Understanding and Utilizing PKI in .NET

Q 1.2: Developing a public key infrastructure using Windows Certificate Services seemed like the way to go until I discovered that there is no auto enrollment. Do you expect that users will be able to obtain certificates on their own?

A: Enrollment of thousands of users is no picnic. Windows .NET Server makes this process easier—automatic enrollment is possible with Windows .Net Certificate Services. Automatic enrollment means that users do not have to learn about certificates, nor about how to obtain one. It also means that re-enrollment can be transparent as well. However, you must configure auto enrollment for each type of certificate you intend to distribute. This requirement is a good thing, as you may not want all certificates distributed in this manner. It does, however, mean more planning and administrative work must be done.


The first step is to determine which certificates should be automatically distributed. For our example, we'll use the ordinary User certificate. This certificate can be used for authentication, email encryption, and the Encrypting File System (EFS). To provide User certificates that can be auto enrolled, you must create a new type of certificate template—a Version 2 certificate template—and configure it for auto enrollment.

Windows 2000 (Win2K) Enterprise Certificate Services use Version 1 standard templates, and although a variety of templates exist, it is not possible to customize templates. Windows.NET Certificate Services can use Version 1 and Version 2 templates, and Version 2 templates are customizable. You create customized Version 2 templates by copying Version 1 templates, then modifying them. After you've created and modified a template to suite your business needs, you can use the template to create certificates using the typical end-user or enrollment-agent enrollment process. Templates modified to allow auto enrollment will be automatically issued to new users. Such templates can also be used to automatically replace expiring Version 1 certificates, or even be quickly pushed out to replace existing certificates.

In addition to auto enrollment, other template modifications are possible. Choices for customization are not endless, but you can add to and modify many aspects of a template, including

- Customization of enrollment policies (who can enroll who)
- Certificate authorization (who can authorize a certificate issuance)
- Domain authentication (which domain contains the user's account)
- Certificate administrator (who manages the Certificate Authority—CA)
- Enrollment agent signed (which enrollment agent is used)
- Key creation (where keys are created and how)
- Key type and Cryptographic Service Provider (CSP) type (which key and which CSP will be used)
- Certificate contents (what information will be included in the certificate)

- Validity, issuance, and application policies and key usages (what can the certificate be used for)
- Key archiving (can the key be automatically archived)

 To create Version 2 templates, you must use a .NET Enterprise Server. In addition, only .NET and Windows XP are able to utilize certificates created with Version 2 templates.

Modifying templates is not difficult, but there are some requirements. You should spend some time evaluating your template requirements as part of your plan for your public key infrastructure (PKI) design. The following considerations might impact the design:

- If you require modified templates, your design must include a .NET Enterprise CA installed on a .NET Enterprise Server.
- Although you might have mixed CA hierarchies consisting of Win2K and .NET servers, only .NET and Windows XP computers can utilize Version 2 certificates and the advanced features they offer.

Quick Steps to Auto Enrollment

Before you can modify templates, you must set up a .NET CA. The CA is responsible for issuing certificates. Although an offline root CA is not required, best practices include establishing a standalone root CA that you keep offline and physically secured. Although PKI designs will vary, at least one .NET Enterprise subordinate CA should be established to manage Version 2 certificates. An Enterprise CA is integrated with Active Directory (AD). I'm going to assume for the moment that you either understand how to set up Certificate Services or that you already have a pristine .NET CA in a test environment to experiment with. To prepare User certificates for auto enrollment you have only to do the following:

1. Log on as an Enterprise Administrator or as a member of Domain Admins in the root domain in the forest.
2. Load the new certificate template snap-in either by opening it in an empty Microsoft Management Console (MMC). Alternatively, you can click Start, Run, then type
`certtmpl.msc`

Or you can right-click the Certificate Templates folder in the Certificate Authorities console, and select Manage, as Figure 1.2 illustrates.

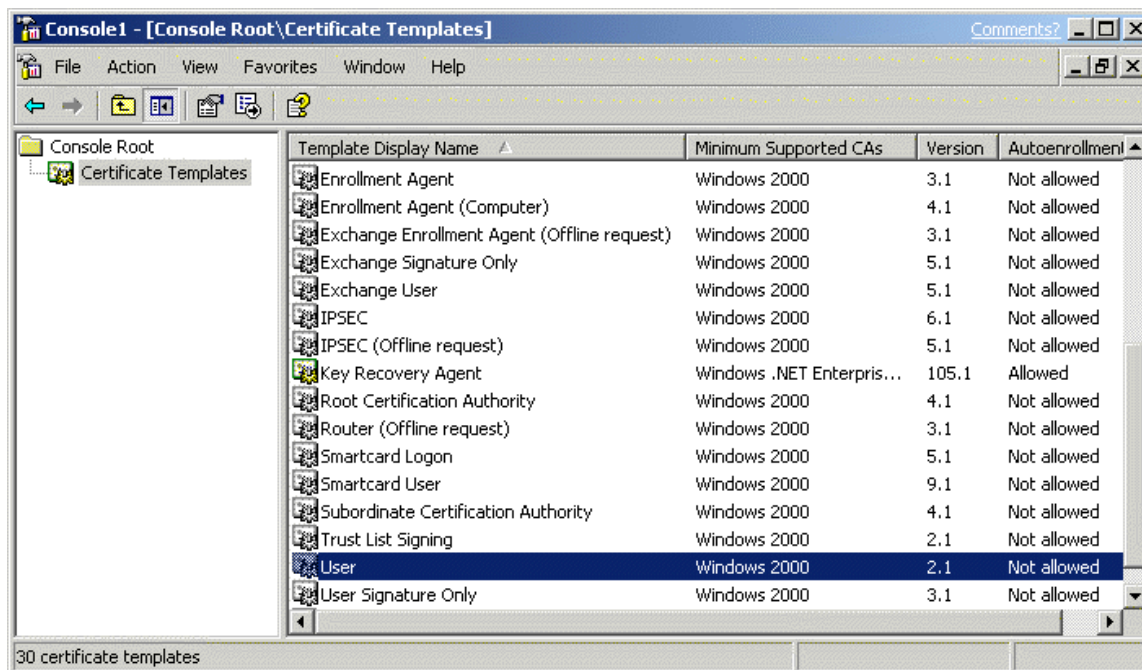




Figure 1.2: .NET provides the new Certificates Templates MMC console.

3. To create a Version 2 template, you must copy a Version 1 template. To do so, right-click the template you want to copy, and select Duplicate Template from the resulting menu.
4. A copy of the template is made and its property pages displayed. Change the template display name (which automatically changes the template name). This opportunity is the only one you will have to do so. After you've saved the new template, you can't change the template's name. In Figure 1.3, the User template has been copied and renamed User2.

 You can also make other changes at this time on the various property pages. You can modify the validity period, renewal period, and storage location of the certificate on the General tab.

 Caution should be taken not to change the validity period to exceed the lifetime of the CA ticket.

No changes are made in our example.

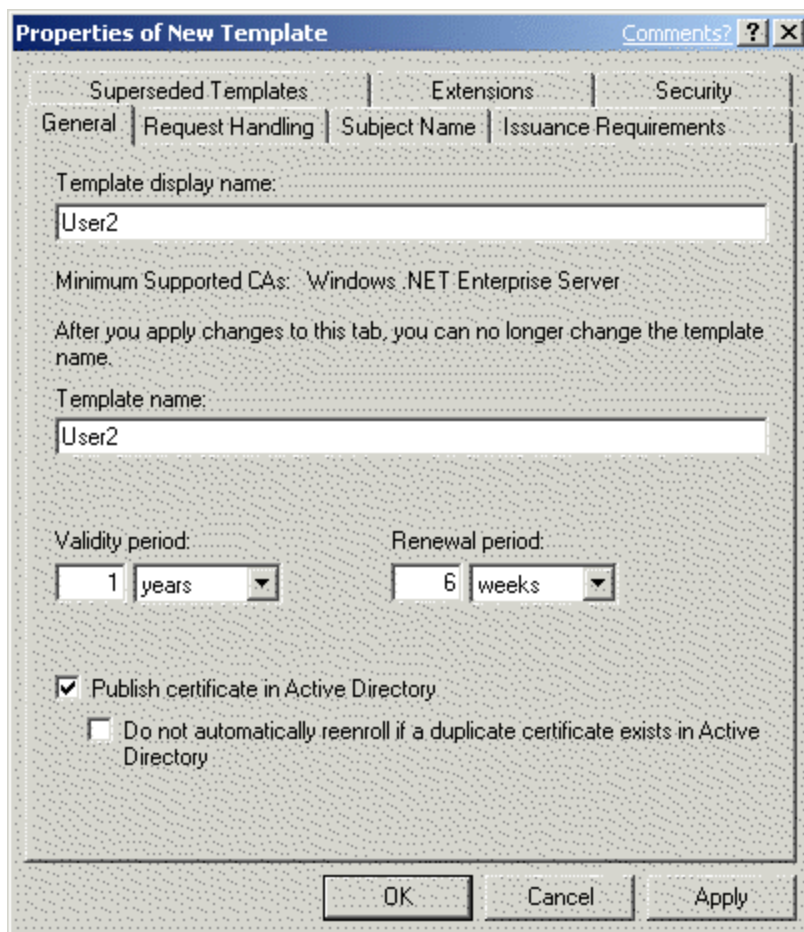


Figure 1.3: Changing the template's display name.

5. Select the Request Handling tab, which Figure 1.4 shows. Because the certificate can be used for EFS, it is tempting to select *Archive subject's encryption private key* check box. However, many other steps must be taken to make this choice usable. In addition to creating a certificate template that allows this, a key recovery agent must be appointed and enrolled and the CA must be configured. We'll leave the check box clear for this example. Other adjustable items on this tab are key size, CSP, permission to export the private key, and the requirement for user input (for example, confirmation that a user wants a certificate) during auto enrollment.

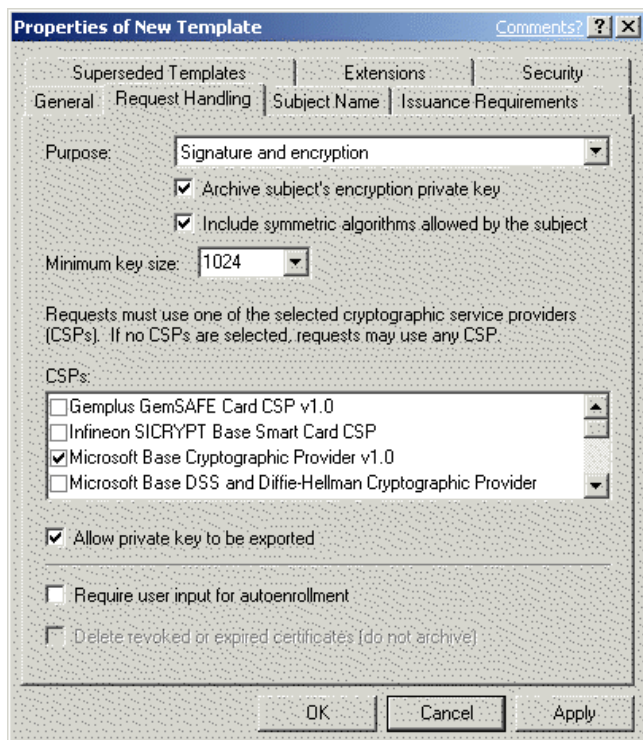


Figure 1.4: The Request Handling tab.

6. View, but do not change, the Issuance Requirements tab, which Figure 1.5 shows. On this tab, you can require approval before certificate issuance. However, requiring approval before auto enrolment is a little counter-productive.

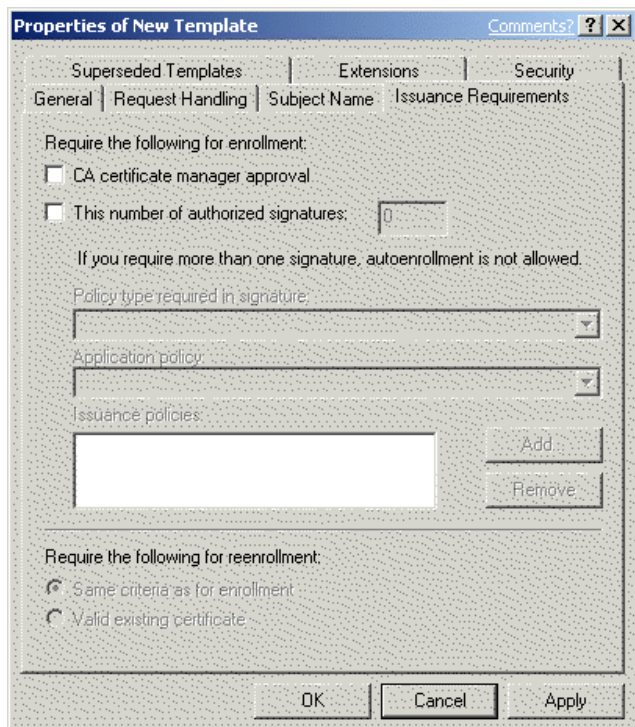



Figure 1.5: The Issuance Requirements tab.

7. Select the Superseded Templates tab, which Figure 1.6 shows. On this tab, click Add to add the user Version 1 template to the Superseded window. (Superseded templates are templates that this new template will replace automatically.) In my design, I want only the User2 template to be used, so I have indicated that the User2 template supersede the User template. In another design, one in which both .NET or Windows XP and Win2K systems coexist, leaving the User template available makes more sense. Win2K systems cannot use Version 2 templates but might need to be able to enroll User certificates.

 Be very careful in your design—if you intend, for example, to require key archival for certificates that can be used for EFS, Win2K clients must not be able to obtain an EFS certificate at all. Neither the EFS Version 1 nor User Version 1 templates can be configured for key archival, and Win2K cannot use Version 2 templates.

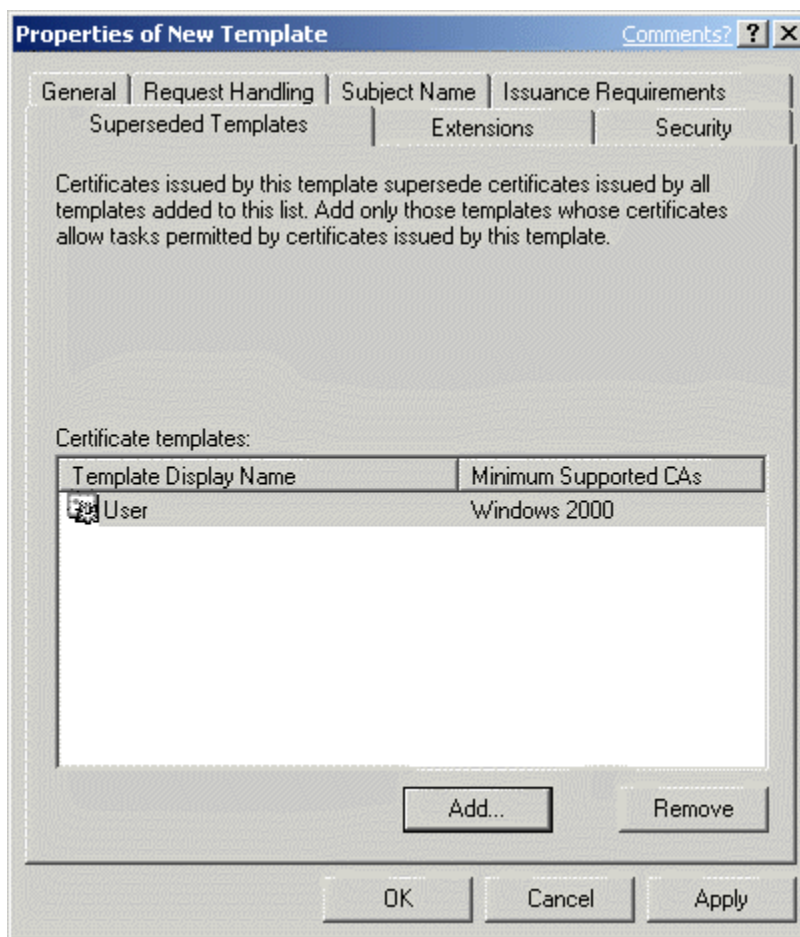


Figure 1.6: Identifying templates for the new template to supersede.

8. Select the Security tab, which Figure 1.7 shows. Select the group or groups that should use auto enroll for this template. In my scenario, I want Domain Users to auto enroll; however, your design might require you to create a special group. .NET, like Win2K, uses the discretionary access control lists (DACLS) set on certificate templates to determine who can manage or obtain a particular certificate type. .NET uses the auto enroll DACL to determine whether a group of users or computers should auto enroll the certificate.

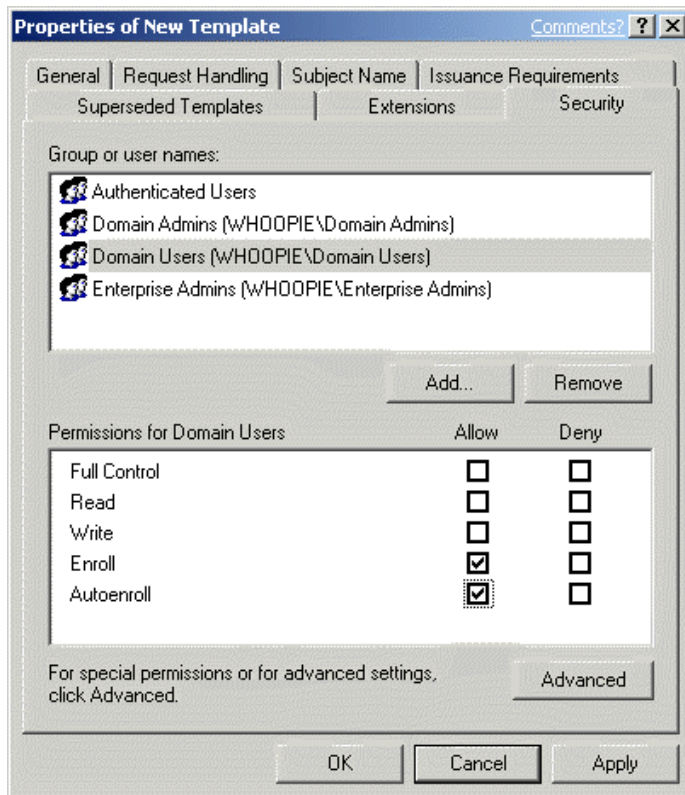


Figure 1.7: You must modify template DACLs to determine who will auto enroll.

9. Click OK to save the template, then close the Certificate Templates console.
10. Open the Certification Authority console.
11. Right-click Certificate Templates, select New, then select Certificate Template to Issue. In the resulting window, which Figure 1.8 shows, select the User2 template, then click OK.

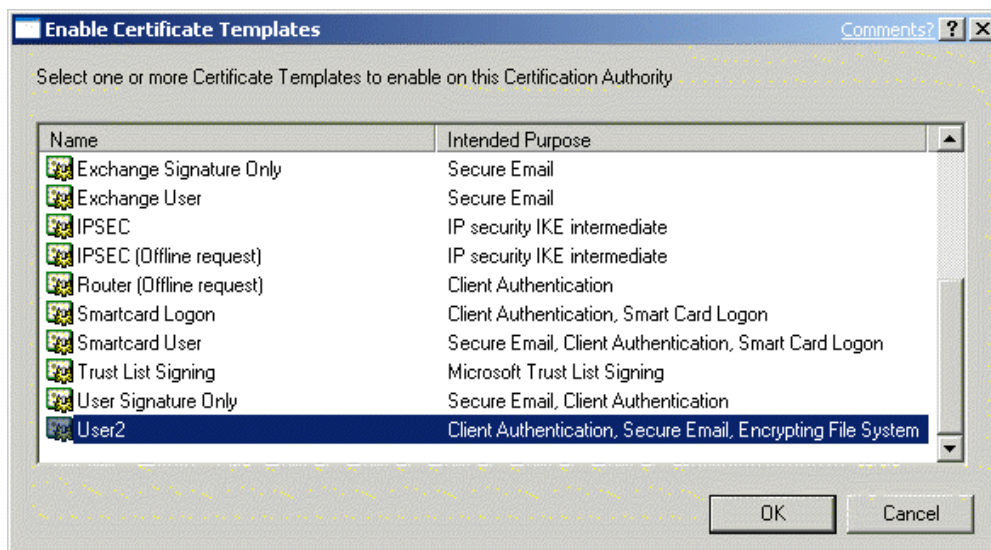



Figure 1.8: The certificate must be enabled for each CA that will issue it.

12. If you would like to replace current User certificates with User2 certificates, right-click the Certificates container, and select *Re-enroll All Certificate Holders*.


 Before you replace certificates, be aware of the capabilities of the certificate you may be replacing. For an example of what you don't want to happen, look no further than EFS. When a new certificate is issued, subsequent files will be encrypted using the new certificate, but what about those already encrypted? The old private key will still be needed to decrypt those files. Changing a certificate has ramifications beyond the simple act of modifying the certificate and superseding an older one. You must carefully consider all sides of the issue.

 For more information about PKI enhancements in .NET Server and Windows XP Pro, check out <http://www.microsoft.com/windowsxp/pro/techinfo/planning/pkiwinxp/whatsnew.asp>.

Chapter 2: Securing Web Services and Web Servers—the Administrative Perspective

Q 2.2: I keep hearing about the new NetworkService account in Windows .NET Server. Microsoft says that this account is less privileged than the Local System account but doesn't explain what that means. Any ideas?

A: Most services in Windows 2000 (Win2K) run under the Local System or SYSTEM account. Thus, they operate with 21 security privileges. Services, then, can do just about anything, whether they need to or not. Because of this power they often attract privilege escalation attacks that are successful. If an attacker can break the service, the attacker might be able to run code under the security context of the operating system (OS), and thus fully own that system. Microsoft has long indicated that best practices require services to run under the least privileged account possible but has not provided much help for those who want to restrict the services that Microsoft provides. The NetworkService and LocalService accounts are two new accounts in Windows XP and Windows .NET Server that are suitable for use by many services. (For information about the Local System approach vs. .NET's approach, see the sidebar "Privilege Best Practices.")

 One of the prime candidates for the NetworkService account is IIS 6.0. In .NET, IIS will install as a static Web server, and requires very few privileges to run. Expectations are that Web masters that require additional abilities will add the abilities as they are needed and adjust the privileges of the service account if necessary. This administrative method is an about-face from the traditional Microsoft everything-is-on out-of-the-box configuration.

If the two accounts have the same privileges, why are there two? The answer lies in how the OS makes use of each. A service running in the security context of the LocalService account will access a remote system anonymously. Let's pretend that a service running on computer LocalDesk attempts to connect to a network share on FileSrv2. If anonymous privileges are appropriately curtailed on the remote system, access will be denied. If we change the account assigned to the service to NetworkService and attempt to access the same share, we might succeed because the service is now accessing the remote system FileSrv2 as the local computer LocalDesk. If the computer account LocalDesk has been given appropriate privileges on the share on FileSrv2, it will be able to connect.

These conditions help to set some guidelines for the use of these new accounts. If a service does not need network access privileges, use the LocalService account. If it does, use NetworkService.

Privilege Best Practices

It's difficult to think rationally about this topic. On one hand, everyone can understand that running a service with an account that only has the privileges it needs is a good thing. It follows the security dictum of least privilege. On the other hand, creating and maintaining a large number of accounts—one for each service—could be an administrative nightmare. Although creating the account and assigning privileges could be a function of the service installation, how can passwords be easily maintained? I can understand why using Local System seemed to be easy, but I believe it has contributed to the problem. After all, processes running with bloated privilege assignments eventually expand their operations to include the use of those privileges, and therefore make it harder to replace the Local System account with a less privileged account—no one knows anymore what the process really needs. .NET's approach is long overdue. Less privileged, yet default, accounts are present, and services can be designed around their use.

What Can the NetworkService Account Do?

In *Writing Secure Code* (Microsoft Press, 2001), David LeBlanc and Michael Howard report four privileges for both accounts: SeSystemtimePrivilege, SeAuditPrivilege, SeChangeNotifyPrivilege, and SeUndockPrivilege. In an April 2002 Microsoft TechEd session, five privileges were identified. This disparity is not at all uncommon in a product under development. It is sufficient to say that the number of privileges for these accounts lies far shy of those of Local System. An explanation of NetworkService's account privileges as identified at the April TechEd session is shown in Table 2.1.


Privilege	Discussion	Required for IIS?
SeAuditPrivilege	Required to generate audit log entries.	Yes
SeIncreaseQuotaPrivilege	Required, along with SeAssignPrimaryTokenPrivilege to access a process token and create a new process that uses the security context of the user of the other process—possibly resulting in a privilege escalation.	Required for Common Gateway Interface (CGI) programs
SeAssignPrimaryTokenPrivilege	Required, along with SeIncreaseQuotaPrivilege, to access a process token and create a new process that uses the security context of the user of the other process—possibly resulting in a privilege escalation.	Required for CGI programs
SeChangeNotifyPrivilege	This privilege lets a user receive notification of changes to files and directories. It is used to bypass directory traversal checks and for NTFS optimization.	Yes
SeUndockPrivilege	Required to remove a computer from a docking station.	No

Table 2.1: Explanation of the NetworkService account's privileges.

What Can the Local System Account Not Do?

Everyone complains that the Local System account has too many privileges to be used to run any service. It is true that this account essentially operates as the OS and has more privileges than the Administrator account. But is it all-powerful? Let's consider what it can't do. It can't shut down remote systems. This inability makes sense—what business would it have shutting down remote systems? It also can't create a new computer account, synchronize directory service data, or

enable computer or user accounts for delegation. If these limitations seem strange to you, look closely. In every case, these privileges have a more global effect. In other words, they allow management of systems in the domain, and this ability exceeds the security boundary of the local computer. In its own realm, Local System is still the most powerful account.

 Finding the privileges of a running process is not something you will find listed in the application documentation. The reason is that the process runs in the security context of the user account that is running it. Services, of course, run in the security context of the account used to log on the service during startup. If these accounts are standard OS accounts, you might be able to find a list of privileges assigned to them in Microsoft's documentation. If you would like to find out the privileges in real time, you can compile and run the Token Monitor application provided in *Writing Secure Code*.

Chapter 3: Understanding Active Directory Foundations

Q 3.2: Help! We have a large, multidomain forest that contains thousands of users, and we use universal groups extensively to provide access to resources. We also have many small branch offices that have only a handful of users each. Because the Global Catalog server must be accessed at logon to determine membership in universal groups, all users at branch offices access Global Catalogs across the WAN even though they have a domain controller locally. If I make the domain controllers at these branch offices Global Catalog servers, I've increased the replication traffic by an unacceptable amount. Does Windows .NET Server have an answer?

A: Windows .NET Server does provide a better way to handle this problem. To understand why Microsoft developed .NET's solution and when it should be used, you must understand the relationship between authentication, universal groups, native mode, and Global Catalog (GC) servers in Windows 2000. Win2K universal groups can exist only in native-mode domains. Universal groups can have as members any user from any domain in the forest, from any other universal group, and from any global group from any domain in the forest. Universal groups can be granted access to any resource in the forest. They often serve as collections of global groups from multiple domains. Thus, any resource can, with the inclusion of a single access control entry (ACE), provide access to large numbers of users. This functionality simplifies administration of resources and reduces the size of the discretionary access control list (DACL) on any one object, as Figure 3.7 illustrates. In the diagram, the ForestPrinters universal group is granted the right to print to printers located in domains A, B, and C. The ForestPrinters group has as members three global groups: Aprinters, Bprinters, and Cprinters. Each of the three groups consists of users from the respective domains.

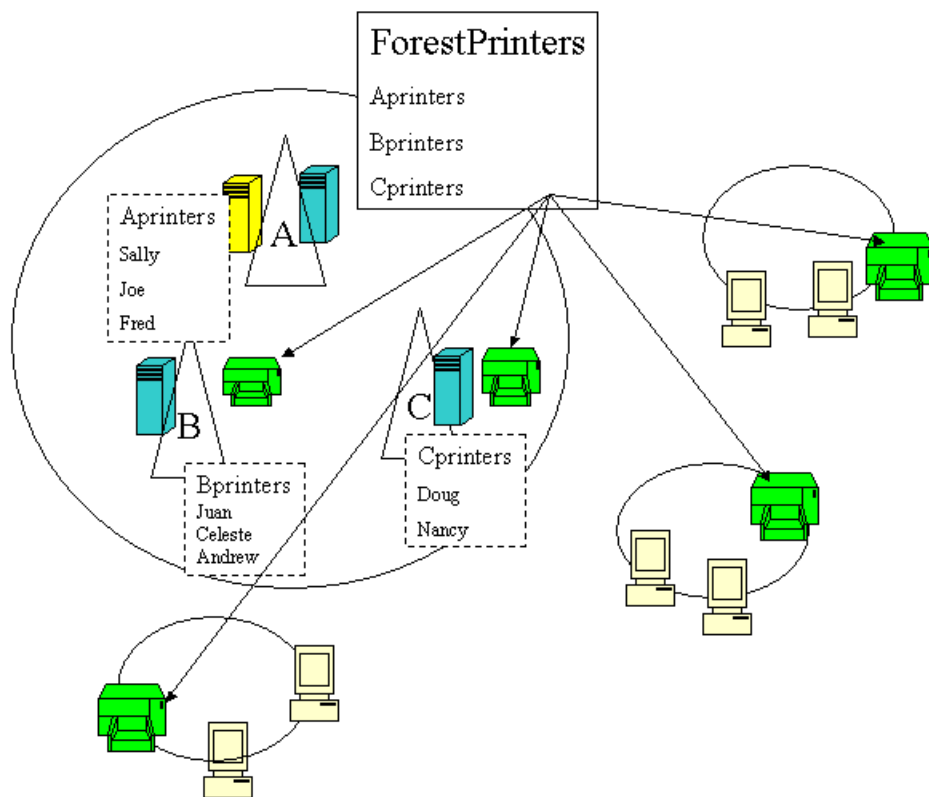


Figure 3.7: Use of the universal group ForestPrinters reduces the number of ACEs in a DACL.

When Nancy, a user in domain C, logs on to the domain controller in domain C, her membership in any universal groups must be determined. This information is determined by querying the GC server. In this forest, the available GC server is a domain controller in domain A. The group security IDs (SIDs) of the groups Nancy is a member of, including the ForestPrinters group, become part of the information in the Kerberos tickets returned to her computer and can be used to create access tokens. This activity is transparent to Nancy. If everything works as designed, Nancy gets her desktop and is able to compose documents, print them, and do other jobs.

A problem can occur, however, if Nancy works in a branch office and there is no domain controller from her domain present in her branch office. If such is the case, to complete her log on, her computer must be able to access a domain controller and a GC across the WAN. Figure 3.8 illustrates this process. In step 1, Nancy's computer will contact the domain controller located at headquarters. In step 2, the domain controller contacts the GC to determine Nancy's universal group membership. Finally, in step 3, the information is returned to Nancy's computer as part of a Kerberos ticket. If the WAN link is down, Nancy's computer will not be able to reach the domain controller or the GC server. She will not be able, therefore, to be authenticated by the domain controller or obtain current information about her membership in and groups, including universal groups. She can, of course, log on with cached credentials, but might then be limited in her ability to access network resources. Cached credentials may be out of date and, thus, not provide correct information to provide her with the rights and privileges she needs to do her job. She also may have trouble connecting to and using network resources as neither her system nor the network resource can complete the processing necessary.

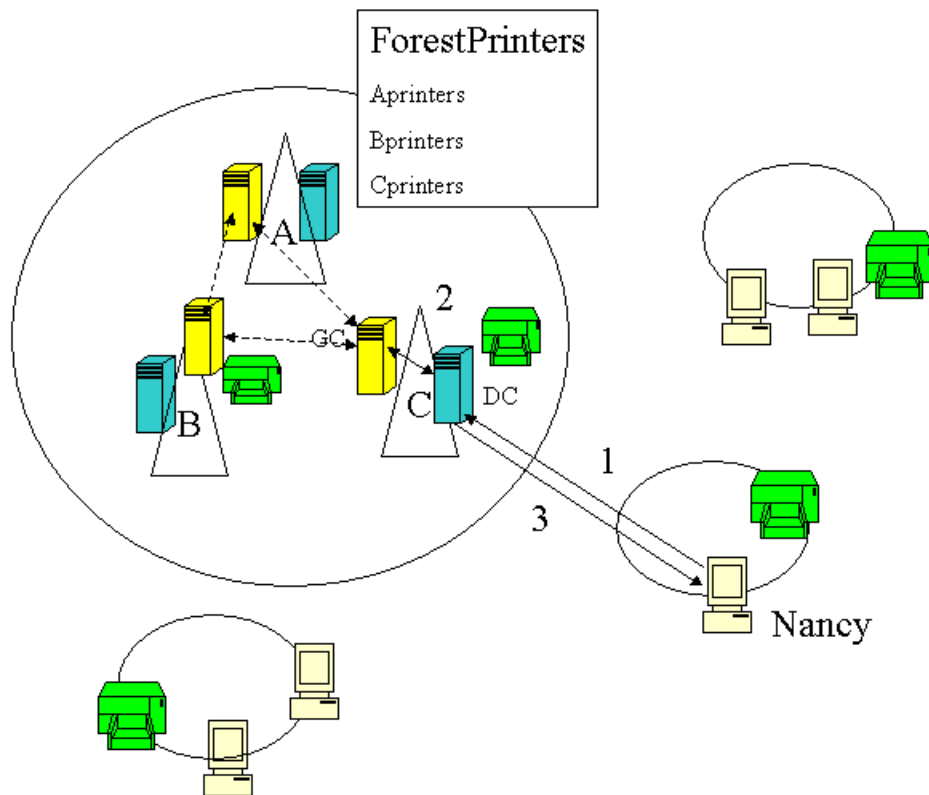


Figure 3.8: Logging on from a branch office requires WAN access to a domain controller and GC server.

Win2K Branch Office Solutions

The intuitive solution to branch office logon for mixed-mode Win2K environments is to place a domain controller in all but the very smallest branch offices. By very small, I mean less than 10 employees. Microsoft guidelines also discuss the use of a single domain controller in a branch office that houses between 10 and 50 users, or calculating the number of domain controllers required based on Group Policy requirements and the use of applications that rely heavily on Lightweight Directory Access Protocol (LDAP) queries against Active Directory (AD).

These guidelines work well while the domain is in mixed mode. However, when domains are moved to native mode, you can create universal groups, which means that logons from branch offices will seek a GC server across the WAN if necessary regardless of whether a domain controller is present. The intuitive solution no longer works. Figure 3.9 illustrates this scenario. When Nancy attempts to log on, her computer is able to communicate with the local domain controller. However, to find a GC, the domain controller must communicate across the WAN to a GC at headquarters. The figure also represents, with the use of dotted lines, replication of data between GCs at headquarters. (Actual replication patterns will vary, but the concept is correctly displayed—information from all domains is replicated to all GCs.) Locating a GC only at headquarters may be an acceptable practice if the number of users in Nancy's branch office is small and WAN connections are stable and meet bandwidth requirements. However, if the WAN link is down, the GC cannot be reached and normal logon will not be possible. If, however, cached credentials exist for Nancy on the computer she is using, then logon will occur.

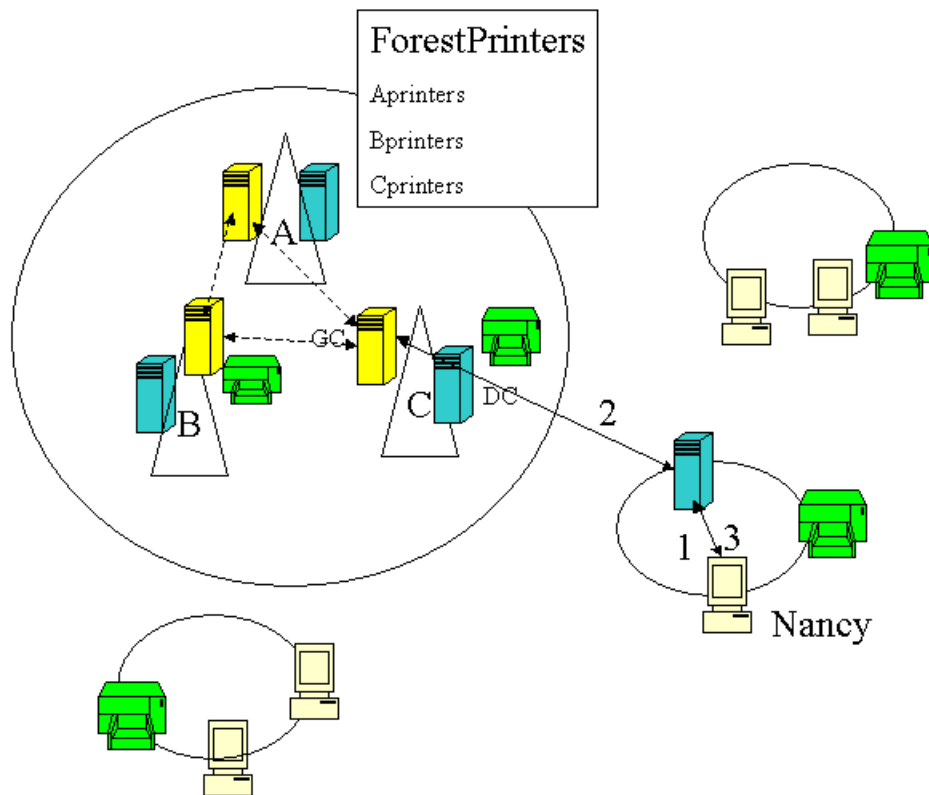




Figure 3.10: A domain controller at a branch office doesn't eliminate the need for WAN access to a GC.

 Is it possible that the lack of a GC server can mean that clients will not be able to log on? Yes, if no cached credentials exist for a user in a native-mode domain and the GC cannot be reached, the user cannot be logged on. This behavior will not affect an administrator. For more information please see <http://support.microsoft.com/default.aspx?scid=kb;EN-US;q216970>.

A solution to this is to place a GC server at each branch office that has a domain controller. However, although this setup means localization of authentication traffic, it means increased replication traffic over the WAN. The GCs must replicate changes in forest data between themselves. Branch offices with low bandwidth connections might find replication demands excessive. In addition, branch office domain controllers might not meet the increased memory and processor requirements of a GC. Figure 3.10 illustrates the authentication and GC replication patterns in this scenario.

 Many forests consist of domains that are in mixed mode and domains that are in native mode. Nancy might be a member of universal groups located in the native-mode domains while her domain is still in mixed mode. What will the logon behavior be? In this scenario, Nancy's logon does not have to access a GC. Instead, should Nancy attempt to access a resource that has universal group access assigned, the domain controller for that domain will be contacted, then the domain controller will contact the GC to determine whether any universal group membership SIDs need to be added to Nancy's access token.

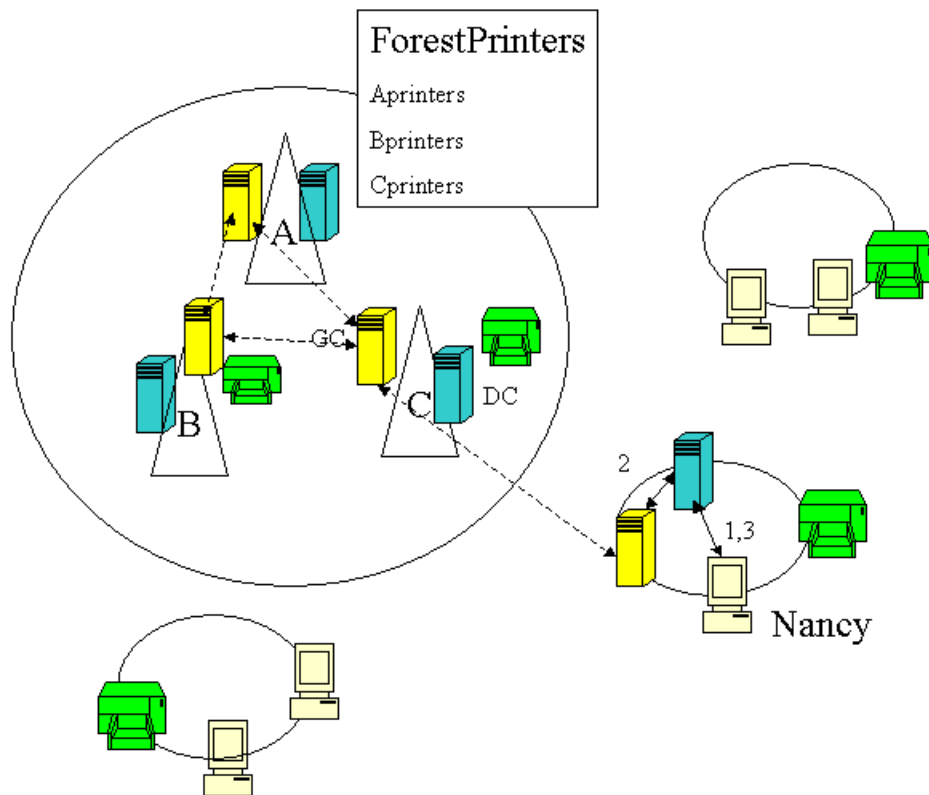


Figure 3.10: Authentication traffic across the WAN can be decreased by placing a GC at a branch office, but this setup means increased replication traffic across the WAN.

.NET Universal Group Membership Caching

.NET provides a simple solution to the problem. Because the major reason that branch offices need access to a GC is to determine group membership in universal groups, .NET lets you configure domain controllers to provide universal group membership caching. If this option is enabled, the first time a user logs on, universal group membership must be determined by contacting the GC server across the WAN. Membership information is then cached indefinitely on the designated domain controller. The following rules govern this behavior:

- Cached data is periodically refreshed.
- The default refresh time is 8 hours
- As many as 500 users' cached universal group membership data can be refreshed at a single time.
- The ability to cache membership data is site specific; thus, all domain controllers in that site must be .NET Server machines.

Universal group membership caching allows faster logons and reduces the need for locating GC servers at branch offices; thus, reducing replication traffic on the WAN. GC servers may still be required at branch offices that use applications that rely heavily on LDAP searches of AD. To configure a .NET domain controller for universal group membership caching:

1. Open Active Directory Sites and Services.
2. Expand the Site object for the site at which universal group membership caching is desired.
3. Double-click the NTDS Site Setting object.
4. On the Site Setting tab of the NTDS Site Setting Properties page, which Figure 3.11 shows, select the *Enable Universal Group Membership Caching* check box.
5. In the *Refresh cache from* drop-down box, select the site from which the cache should be refreshed.

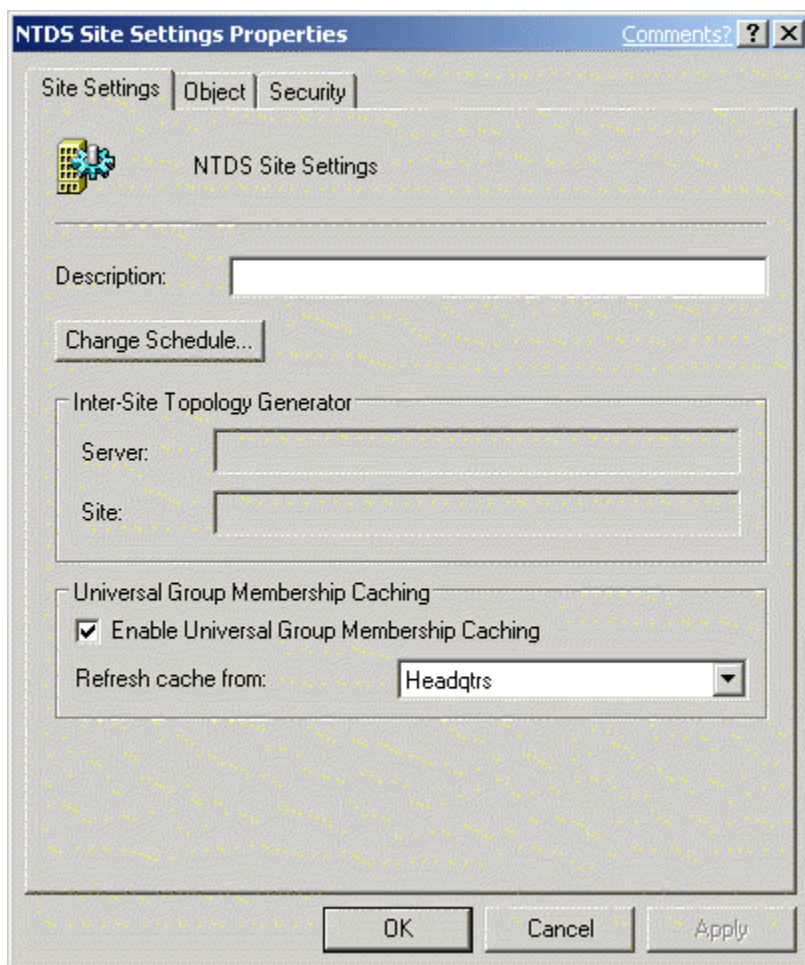


Figure 3.12: Configuring universal group membership caching.

Chapter 4: Fulfilling the Promises of Group Policy

Q 4.2: How can I prevent users from running tools that they should not run?

A: Although you didn't mention which tools you don't want users to run, I can think of numerous tools that should not be used by most ordinary users, such as registry editing tools, networking utilities, and resource kit utilities. There are three ways that you can prevent users from running these tools:

- Protect executable files by using file permissions
- Remove executable files from users' machines
- Use Software Restriction Policies

Setting Restrictive File Permissions

Setting restrictive file permission is the time-honored way of restricting the use of files. Recommended security practices include making sure that all drives on all Windows NT, Windows 2000 (Win2K), Windows XP, and Windows .NET computers are NTFS formatted. When Windows is installed on NTFS drives, standard file permission settings on system folders are fairly secure; the administrator's responsibility is to lock down other folders and files and consistently investigate, test, and deploy more restrictive settings on system files and administrative tools. However, this method doesn't prevent the savvy user from placing a copy of a restricted tool in a folder in which the user has the right to run the program. And there is also no guarantee that for every machine, every permission setting is correct.

You can write Group Policies that set file and folder permissions to a corporate standard, and push these settings across an enterprise. However, a sophisticated user or an attacker has only to provide their own copies of the restricted tools to foil these administrative tactics.

Removing Executables

Another sound practice is to remove the executables. There are two problems with this approach. First, a user can still provide their own copy of a tool. Second, Windows File Protection (WFP) might make it difficult to remove tools that are considered part of the OS and thus protected. Another concern is that patches, system updates, and additional Windows components might reinstall the removed executable without warning.

Using Software Restriction Policies

The ability to use Software Restriction Policies is new with Windows XP and .NET. Software Restriction Policies offer a new way to prevent unauthorized use of system tools, restrict users to only approved software, and prevent attackers from using system tools in an attack on the system. Software Restriction Policies let you, the administrator, either create a policy that disallows any software, then create unrestricted rules that let only approved software run, or create a policy that lets all software run, then create disallowed rules that prevent identified software from running. Software is either disallowed or unrestricted based on the following rule types:

- Path—Explicitly identify the program path.
- Hash—When a program is selected, a cryptographic hash is built. Any attempt to run the program will result in a check of the hash, and the program will be allowed to run (or not allowed to run) based on the type of policy. The program can reside anywhere, and the action will be the same. This option is useful to prevent software from running no matter the location.
- Certificate—Rules are built based on the presence of a code publishers' software signing certificate.
- Internet zone—You can prevent software from running from a particular Internet Explorer (IE) software zone. You cannot prevent software that has been downloaded from that zone from running.

Enforcement properties include or exclude DLLs and identify whether the policy applies to members of the Administrators group. Additional settings allow addition or deletion of file types (designated file types) that can be managed and determine whether users, computer administrators, or enterprise administrators are affected by trusted publishers. By default, policies apply to all users and program files except library files such as DLLs.



Software Restriction Policies might not appear where you would expect them in a Resultant Set of Policies (RSOP) report. When you run RSOP to determine the effective policy applied for a user, you would expect Software Restriction Policies to appear in the location in which they are configured (User Configuration, Windows Settings, Security Settings, Software Restrictions). However, Software Restriction Policies might appear under Extra Registry Settings instead.

To create, examine, or manage local Software Restriction Policies:

13. Click Start, select Run, and type
`secpol.msc`
in the Run text box.
14. Click OK or open the Administrative Tools, Local Security Policy console.
15. Select the Software Restriction Policies container.
16. If no policy exists, right-click the container, and select Create Software Restriction Policy.

To create or manage Software Restriction Policies for a site, domain, or organizational unit (OU), open the Group Policy Object (GPO) for the appropriate container. The Software Restriction Policy container is located under Computer Settings, Security Settings.

A Simple Policy to Restrict Tool Use

You can easily create a policy that restricts the use of tools. You will have to determine the list of tools that need to be restricted and the type of rule that will be used. Unfortunately, you'll have to come up with your own list of tools, but I can help you understand how to make choices for the type of Software Restriction rule to use, and provide you with a step-by-step policy-creation exercise.



It is possible to create a Software Restriction Policy that can prevent systems from running. Therefore, the best procedure to follow is to create a policy in the Local Security Policy of a single Windows XP system. Then test the policy by applying it to a single OU that represents a test computer or computers. When you are sure that this policy meets your requirements for software restriction without breaking other software, you can use the policy to restrict multiple systems.

A good first policy will start with all software allowed to run and will contain rules that prevent individual programs from running. This process will give you the opportunity to learn how the policy works without taking as large a risk of making the system inoperable. The following Local Security Policy example meets these requirements.

Again, always test Software Restriction Policies on a single machine first! It is possible to crash the operating system (OS) by being overly restrictive.

Creating a Simple Software Restriction Policy

This example policy will prevent the following tools from running:

- All software located in the C:\alpha geek tools folder
- Regedit32.exe (no matter where it is located)
- Software on sites included in the IE Restricted Sites security zone
- Solitaire game (sol.exe—no matter where it is located)

Figure 4.3 shows the completed policy.

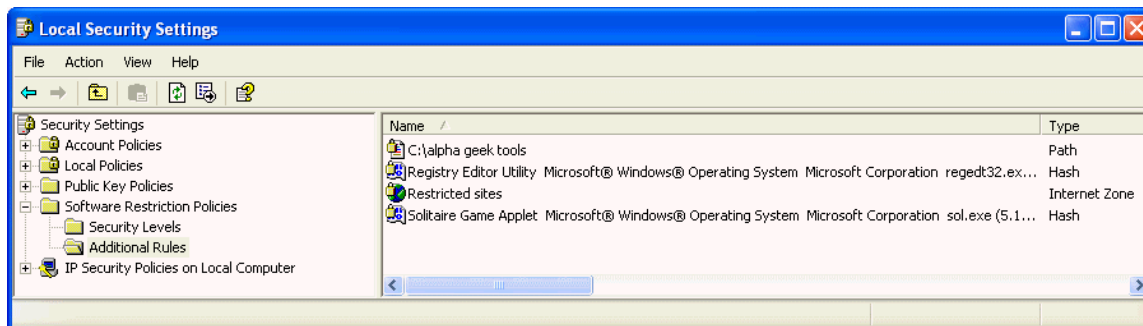


Figure 4.3: The *Additional Rules* section of a policy lists the policy type and basic information.

This policy should not restrict administrators, so the first step to take is to set the policy so that it will only affect ordinary users:

1. Double-click the Enforcement object at the root of the Software Restriction Policy container.
2. In the resulting window, which Figure 4.4 shows, change the setting from the default by selecting the *All users except local administrators* check box, then click OK.

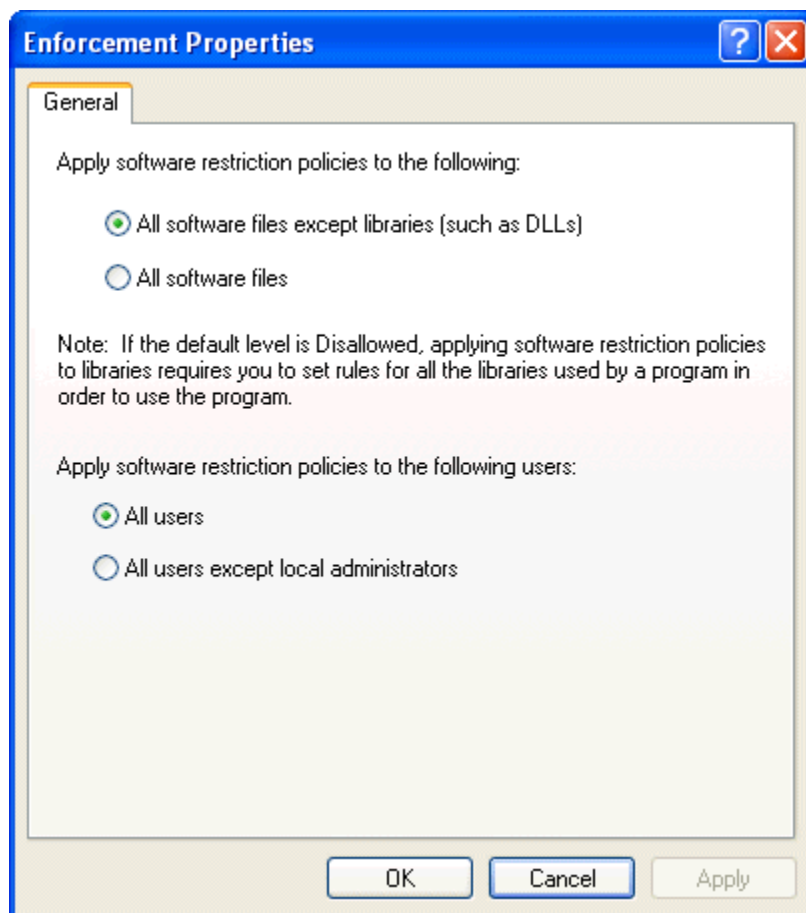


Figure 4.4: Be sure to change the policy from the default setting to let local administrators use the restricted tools.

Next, a rule for each tool, folder, or certificate should be restricted:

1. Right click the Additional Rules container, and select *Create new path rule*.
2. Enter the path C:\alpha geek tools (or browse to select it).
3. Leave the security level at Disallowed, and click OK.
4. Right click the Additional Rules container, and select New Internet Zone Rule. (Differences in options may exist due to different versions of beta software.)
5. Select the Internet zone Restricted Sites.
6. Leave the security level at Disallowed.
7. Change the description to comment that these sites are ones you do not trust, and click OK.
8. Right-click the Additional Rules container, and select New Hash Rule.
9. Browse to a copy of the sol.exe file. The hash appears in the File Hash box, and information about the file will appear in the File Information box. Leave the security level at Disallowed, and click OK.
10. Repeat these steps, except browse to a copy of the regedt32.exe file in step 9.

Testing the Policy

The first time you create a rule of a particular type, test the rule. You can do so by logging off and logging on as an ordinary user, then attempting to run the tool. You should be refused and receive the message that Figure 4.5 shows.

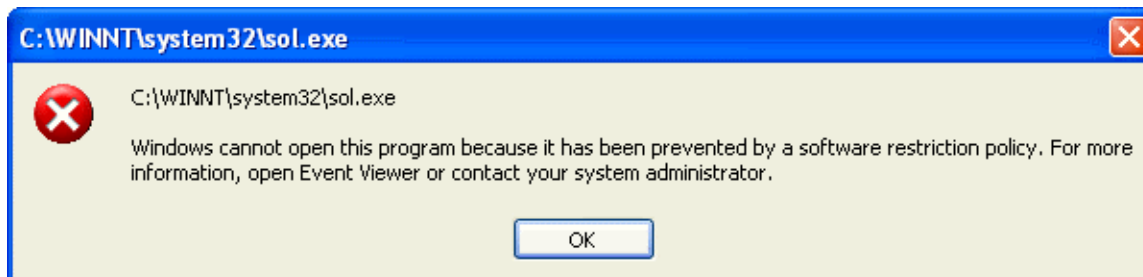



Figure 4.5: If a user attempts to run a restricted program, an error message will appear.

Next, log on as an administrator and attempt to run the tool. You should be able to. The instructions I previously provided don't include these testing instructions for brevity sake. Finally, when the policy is complete, test all rules to ensure that they operate as you expect. Any changes to the rules should require a retest.

 Are these tests sufficient? You should always seek to write exhaustive tests for your policy. No one list of testing scenarios will work for every rule set. Do not rely on these suggestions alone to work for your policy.


To test this policy, make sure that you have designated in the Restricted Sites zone in IE a Web site that has software that can be run from its Web pages. To test the policy that we created in our example, log on as an ordinary user and attempt to run files in the alpha geek folder. Attempt to play solitaire. Attempt to run regedt32. Copy the sol.exe file to a temporary folder. Attempt to play solitaire. Open IE and navigate to the previously designated Web site. Attempt to run software. All of these tests should fail. A pop-up message similar to the one that Figure 4.5 shows should be the result of each attempt. Next, try to download software from the restricted Web site and run it. This attempt should be successful. The Software Restriction Internet zone rule only applies to programs that are run from within the zone. Next, log on as Administrator, and repeat the tests. You should be successful each time.

Examining the Policy for Holes

It might be possible, even desirable, to create multiple rules to handle some areas. For example, if your policy seeks to absolutely prevent users from running certain tools, you should create hash rules for each tool. Path rules, such as the one written for C:/alpha geek tools, will only prevent the running of tools from within this folder and its subfolders. If the user can copy a tool from the folder to another, that user can run the tool. If the user can obtain a copy of the tool from another source, the user can run the tool. A hash rule, however, will prevent the software from running from any location on the computer.

The rationale behind path rules is twofold. First, it quickly blocks any new tool that is added to the folder. This affords some protection while the Group Policy is adjusted and a hash rule is written for the specific program. Second, if the Software Restriction Policy is written to disallow

all software, a large number of system programs must be allowed to run. Being able to do so by selecting a folder versus creating individual rules for all system files is essential. Hash rules can be used to specifically prevent some tools located within the system folder from running. Also, if system files are being blocked, don't forget to think about the copies that may be in dllcache.

 If you are not using hash rules to prevent software from running, don't forget to make a path rule that includes %windir%\system32\dllcache. A copy of the disallowed program might be at this location, and if a path rule does not cover it, the program will be able to run.

You should also be clear about which program file types are covered by your policies? When a path rule is written, if a program file type is not covered, the program will be allowed to run. You can examine which program file types are covered by opening the Designated File Types object of the Software Restriction Policies container. From this window, you can also add and remove file types.

Another policy hole consideration results from a program that calls another program that calls another program—is this behavior restricted or allowed? The answer is not simple and there is no blanket rule. If a program is disallowed, it cannot call other programs. If a program can run on its own or is called from within another unrestricted program, the program will run. Conversely, if an unrestricted program calls a disallowed program, the disallowed program will not run, possibly resulting in the failure to run of the unrestricted program.

What will happen if multiple policies apply to the same programs? As the following list shows, an order of precedence exists. The first element listed, hash rule, is highest in the order and will take precedence:

1. Hash rule
2. Certificate rule
3. Path rule (if path rules conflict, the most restrictive will take precedence)
4. Internet zone rule

Finally, you should realize that later versions of a restricted program might not be restricted by the rules you have written. A hash rule applied to a known virus file, for example, cannot restrict a later version of the virus.

Certificate Rules

Another type of software restriction rule is the certificate rule type. Certificate rules apply to scripts and .MSI files only. To use them, a code-signing certificate is used to sign the files. Certificate rules are used to identify the code-signing certificates that are valid on this computer or on the computers within the Group Policy Container (GPC) of this GPO. Certificate rules are an excellent way to ensure that only approved scripts can be run. They also solve the dilemma caused by the following scenario: If all software is disallowed, rules might be written that let the OS run. In addition, rules can be written that allow only the software a user needs to run. How then, can logon scripts, new software installations, and maintenance scripts be allowed to run? If a rule is written that indicates that certificate A is approved, any script or .MSI file that is signed with certificate A will run. Those not signed by this certificate will not run.

Trusted Publishers

In a related area, Trusted Publishers Properties, which Figure 4.6 shows, should be configured to determine who can select trusted publishers in IE. Trusted publishers can be designated within the Content tab, Trusted Publishers section of IE. This designation lets software that is signed by approved software publishers be run from IE. The default setting is to allow end users to select trusted publishers.

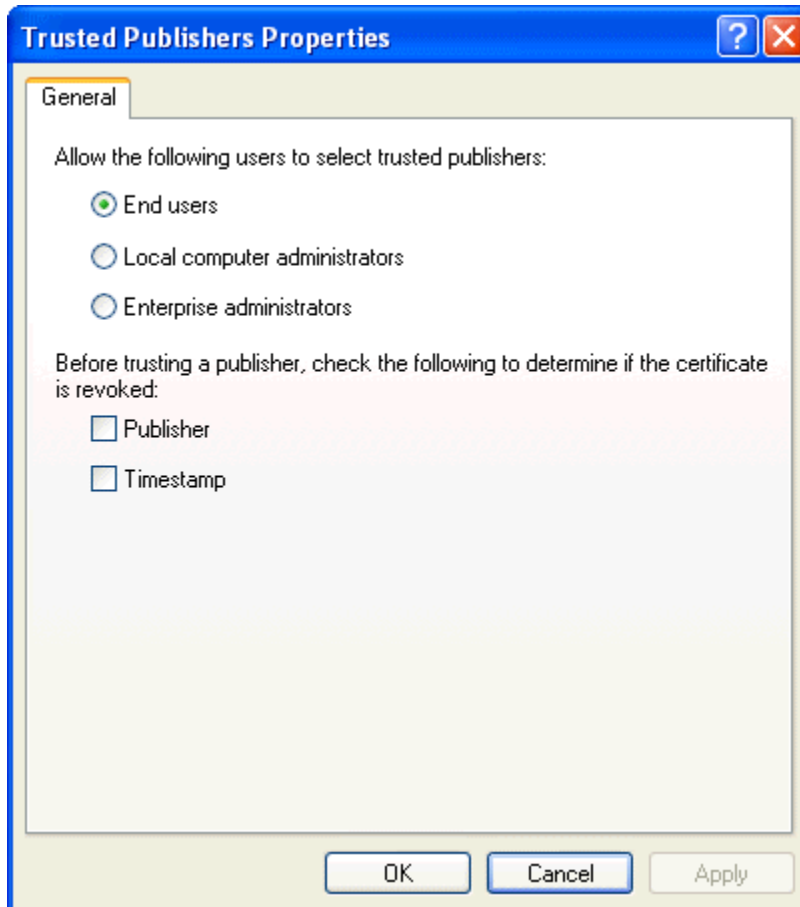


Figure 4.6: The default rule for trusted publisher lets users select them.

Moving On

Now that you can successfully create a small local software restriction policy, you might want to design and execute domain-wide policies. Before you attempt to do so, you should be aware of several considerations. First, be aware how very new this process is. Second, .NET has not been released. Third, you will need to develop sophisticated troubleshooting skills and procedures and detailed testing plans before deploying widespread policies.

There is a lot more to software restriction policies than you first might believe. This area is new and will take a while before all the bugs and best techniques are worked out. In the time it took to write the first draft of this answer and do my first review, a bug was discovered in the processing of rules that restrict 16-bit programs. The Microsoft article “Software Restriction Policies Do Not Recognize 16-Bit Programs” at

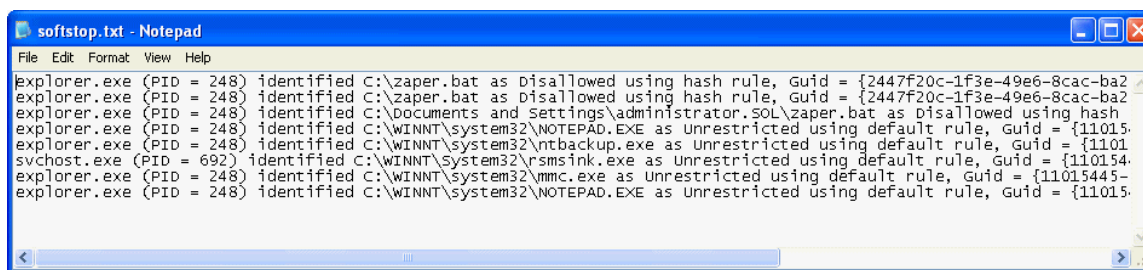
<http://support.microsoft.com/default.aspx?scid=kb;EN-US;q319458> states that 16-bit software

that is run in the virtual machine (ntvdm.exe) is not recognized by software restriction policies. Thus, the rule written to prevent the running of command.com or edit.exe doesn't work. Users can run these programs even if the programs were specifically denied by software restriction rules. Microsoft already has a fix to correct this problem. It can be obtained from Microsoft support services. The company advises you to obtain and use the fix only if you are attempting to restrict such software.

Win2K cannot process software restriction policies. You will have to wait for .NET Server and your migration to .NET domain controllers if you want to fully benefit from these practices. You will only be able to impact users with Windows XP on the desktop. Still, software restriction policies will be useful policies as you move forward.

Even simple software restriction policies can easily frustrate. A simple troubleshooting technique is to allow the collection of processing into a log file. You can do so by adding a value to the registry and entering the path and file name of the log. To do so, add the string value LogFileName to the

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Safer\CodeIdentifiers registry key. For my test, I then entered the value C:\temp2\softstop.txt (see Figure 4.7), then ran some tests. Files that I didn't explicitly run showed up on the list.



```

softstop.txt - Notepad
File Edit Format View Help
explorer.exe (PID = 248) identified C:\zaper.bat as Disallowed using hash rule, Guid = {2447f20c-1f3e-49e6-8cac-ba2
explorer.exe (PID = 248) identified C:\zaper.bat as Disallowed using hash rule, Guid = {2447f20c-1f3e-49e6-8cac-ba2
explorer.exe (PID = 248) identified C:\Documents and Settings\administrator.SOL\zaper.bat as Disallowed using hash
explorer.exe (PID = 248) identified C:\WINNT\system32\notepad.exe as Unrestricted using default rule, Guid = {11015
explorer.exe (PID = 248) identified C:\WINNT\system32\ntbackup.exe as Unrestricted using default rule, Guid = {1101
svchost.exe (PID = 692) identified C:\WINNT\system32\rsmssink.exe as Unrestricted using default rule, Guid = {110154
explorer.exe (PID = 248) identified C:\WINNT\system32\mmc.exe as Unrestricted using default rule, Guid = {11015445-
explorer.exe (PID = 248) identified C:\WINNT\system32\notepad.exe as Unrestricted using default rule, Guid = {11015

```

Figure 4.7: You can troubleshoot software restriction rule problems by creating a log through registry edits.

Chapter 5: Administrative Authority

Q 5.2: I seem to have locked out the Administrator account in the domain. I can no longer log on to the domain using this account. What should I do?

A: You probably have introduced an empty Restricted Groups policy for the Administrators group at the domain controller Group Policy Object (GPO) level. You will need to add the Administrator account to this policy.

Windows 2000 (Win2K) introduced the concept of Restricted Groups and it is present in the same form in Windows .NET. This Group Policy was developed to allow domain administrators to control the membership of local groups on computers within an organizational unit (OU) or within the domain. Let me explain further.


Each Win2K, Windows XP, and .NET computer has a local Administrators group and a local Security Accounts Manager (SAM) database of accounts even if the computer has been joined in a domain. As you know, account membership in the local Administrators group gives broad powers on the local machine. This access is a very good reason for restricting membership in this group to only those individuals who require such power. Unfortunately, it is difficult to restrict those who are given membership in the Administrators group from adding other members. It is also an attack technique to add users to administrative groups after successfully obtaining administrative access. The attackers can then use those user accounts in the future.

However, if a group is added to the Restricted Groups policy, and users are added to this group, only these group members will be allowed. If members are added to the group on the local machine, when the policy is refreshed, they will be removed. I've illustrated this confusing concept in the following example scenario:

Sally's account is given membership in the Administrator's group of her Windows XP machine. A Group Policy exists for the OU in which Sally's computer account resides. The Administrator's group is added to the Restricted Groups policy in this Group Policy. Sally's account is added to the Administrators group within the policy. Sally is tricked into adding Tom's account to the local Administrators group on her computer. Tom has no domain administrative privileges. During policy refresh, the discrepancy is found. Tom's account is not in the Restricted Groups Administrators group. Tom's account is removed from the local Administrators group on Sally's computer. That afternoon, Tom attempts to remotely edit the registry on Sally's machine to play a trick on her. He cannot do so. At the end of the week, John reviews the audit logs on Sally's machine and finds the records in which Sally added Tom to the Administrators group and Tom attempted to remotely access the registry. Tom and Sally are in a little bit of trouble.

You can use Restricted Groups to manage membership in any group, whether it is default or created. However, problems might be caused by people who do not read instructions or because it is easy to select the default groups that exist on the domain controller instead of the desired groups on the local machines. Because many people do not study the documentation and insist on trying configuration on production machines without proper testing, it is possible to prevent authorized individuals from doing their administrative job, and indeed, it is possible to lock out the domain Administrator account. It happens because two steps must be taken to create a proper

policy. First, the group must be added to the Restricted Groups policy. Second, authorized members must be added to the group. Any authorized member that is not added to the group inside the Restricted Groups policy will be removed from the group. The two-step process is not intuitive and doesn't produce a warning if not carried out properly. The administrator or consultant who likes to break things to learn about them will be rewarded here.

 What happens if an Administrator for a child domain removes the Enterprise Admins group from the local Administrators group in his or her domain? Members of the Enterprise Admins group have limited access to child domains. (To totally prevent the Enterprise Admins group from having any access in a child domain requires a little bit more work.) Can the Enterprise Admins get their access back? Yes. They can do so by creating a SITE policy that adds the Administrator group of the child domain, then adds the Enterprise Admins group to the Administrators group in the policy. At the next policy refresh, the Enterprise Admins will have access to the child domain.

Unfortunately, it is also possible for the careful reader of documentation to create a Restricted Groups policy with unexpected results. For example, suppose your domain, east.newyork.mpptp.local, is a Win2K domain and all your clients are Windows XP. You want to create a policy to restrict membership in the local Administrators group of all Windows XP computers. To do so, you open the Domain Security Policy, right-click Restricted Groups, click Add Group, click Browse, select Administrators, and click OK. Because this policy is for the Windows XP computers and you do not want the default Administrator account to be part of this group, you do not add any members. You close the Default Security Policy.

Sound correct? Yes, except for one thing. Because the domain policy will be applied to every domain computer, it is also applied to the domain controllers. Browsing in the Add User windows located in the domain Administrators group, you will find that the Administrator account cannot log on to the domain. To correct this situation, an Enterprise Admin can logon from another domain and open Active Directory Users and Computers. By changing the focus of the console to the east.newyork.mpptp.local domain, the Enterprise Admin can edit the Restricted Groups policy to correct the situation. To do so, the Enterprise Admin member needs to remove the domain\Administrators group from the Restricted Groups policy, then add a group by typing

Administrators

This policy will then apply to the local Administrators group on the computers in the domain instead of the built-in Administrators group on the domain controllers.

Chapter 6: Triple A's—Authentication, Authorization, and Audit

Q 6.2: I am tasked with tracing logon behavior across our Windows .NET forest and would like some help understanding the difference between the Audit categories Audit Account Logon and Audit Logon. What is the difference?

A: Here's a simple explanation of the difference between the two:

- Setting Audit Account Logon Events records events where the account resides. If Nancy sits at her desktop computer and logs on to the domain, Account Logon Events will be recorded in the Security event log of the domain controller. Many Account Logon events are Kerberos events.
- Setting Audit Logon Events records events where the logon event occurs. These events are recorded in the Security event log of Nancy's desktop computer.

Thus, to have a clear picture of a user's domain log on and log off behavior, you must enable auditing of Audit Account Logon Events on the domain controllers and Audit Logon Events on every computer that the user might use to log on to the domain. You will then need to consolidate event logs from multiple computers and query for the different logon events associated with that user. You must also keep in mind the many logon events that might be generated during this user's normal activities:

- Account Logon Events might be recorded when the user attempts to access resources, not just when the user logs into the domain.
- Multiple logons by the user from different computers.
- Logons by the same user using another account.
- Unlock workstation events.
- The user's use of the *run as* command to perform secondary logons
- The user's employment of alternative credentials when mapping network drives.

Tracking logons is not a simple task, but a user's activity can be tracked with a little practice and the knowledge of typical events' meanings.

Logon Events

Multiple events might be recorded. Tables 6.1, 6.2, and 6.3 detail Account Logon Events and Logon Events.

Event	Definition	Comments
672	Authentication Server (AS) ticket issued	This ticket is the first Kerberos ticket issued if the user presents valid credentials; it essentially says "OK, this user is who they say they are."
673	Ticket Granting Service (TGS) ticket issued	This ticket is issued for a specific server or service. When the user, for example, wants to access files on a file server, a TGS ticket for the file server is required.
674	Security principle renewed an AS ticket or TGS ticket	Tickets are renewable (see Kerberos policy definitions in the domain Group Policy).
675	Pre-authentication failure	Authentication failed, many times as a result of a time-synchronization failure.
677	TGS was not granted	Failure perhaps as a result of an invalid AS, unknown server or service, or time skew.
678	Account mapped to domain account	A confirmation that the account has been located.
681	Domain account logon attempt	If an account exists within the domain, logon can fail for many reasons (see error codes listed in Table 6.2).

Table 6.1: Account logon events are recorded where the account resides.

Error Code	Definition	Comments
1326	Unknown user or bad password	The user account or password is invalid.
1328	Time restriction violation	A valid logon time range is defined for this user account and this time is not within that range.
1331	Account disabled	The account is disabled at this time.
1329	Log on not allowed at this computer	The computers from which this account can logon are specified in the account for the user. This computer name is not within that list.
1327	Logon of this type not allowed at this computer	Policy does not allow this type of log on (network, interactive, remote) at this computer.
1330	Account password expired	The password for this account expired.
1792	Netlogon not active	The Netlogon service has stopped.

Table 6.2: Account logon failure codes for Event 681.

Event	Definition	Comment
528	User successfully logged off	Always present if a user logged off in normal fashion.
529	Failure due to bad password or unknown user name	Either an attack or the user forgot his or her password.
530	Attempt to log on outside of allowed time	Allowed time range is set in user account properties.
531	Attempt to log on using a disabled account	A disabled account is not a locked out account. A disabled account is made so by an administrator selecting the check box labeled Disabled. A locked out account is locked out due to unsuccessful logon attempts.
532	Attempt to log on using an expired account	Temporary workers' accounts are often set with expiration dates that can be changed. This event is probably the result of a contractor staying beyond the estimated job tenure.
533	User is not allowed to log on from this computer	Allowed computers for logon can be set in user properties. This computer is not one of those listed for this user.
534	Attempt to log on with a type of logon that is not allowed	Logon locally, network logon, batch logon, service logon, and remote logon can all be denied implicitly or explicitly.
535	Attempt to log on with an expired password	Passwords can be set to expire (there is a difference between expired account and expired password).
536	The Netlogon service is not active	Netlogon stopped or is otherwise malfunctioning.
537	Failed for some other reason	Other reasons not documented.
538	User logged off	User employed Ctrl+Alt+Del to log off.
539	Account is locked out	Bad logon attempts reached the account lockout threshold for this account.
540	Successful log on to a computer	Logon was successful.
541 – 547	Logon events related to IKE	Computer endpoints can be authenticated using IPsec policies. These logon events are reported as well.

Table 6.3: Logon events.

Tracing a User's Logon and Logoff Events

One of the benefits of understanding ordinary events is that you'll know what extraordinary events mean. A good exercise is to attempt to determine which events will be found in the logs of the client computer and the domain controller during an ordinary logon. After compiling your list, log on at a client as an ordinary user. Leave the user logged on while you examine the event logs of the domain controller. Are you able to find what you expected? Remember that you can filter the Security event log by the user's name but that Kerberos logon events will be register as system events (NT AUTHORITY\SYSTEM). Find the user logon event (an event 528 logon type 3 for network logon) using the filter on the user's name, then remove the filter and examine the nearby events. You will find that when opened, the details list the user ID. Return to the client

computer and log off the user. Log on as Administrator to view the security events. What would you expect to find here?

On the local workstation you should find a 528 event—successful logon, as Figure 6.5 shows. In this figure, Nancy, is clearly identified within the body of the event message. Her SID is the only identity in the main event heading because this event was pulled from an exported event log. On the local workstation VMXP1, the user Nancy replaces the SID. If the local workstation log had been exported in comma or tab-delimited format, the name Nancy, not the SID, would appear. Working with user names is easier than working with SIDs, which provides a good reason for exporting logs in comma or tab-delimited format for review.

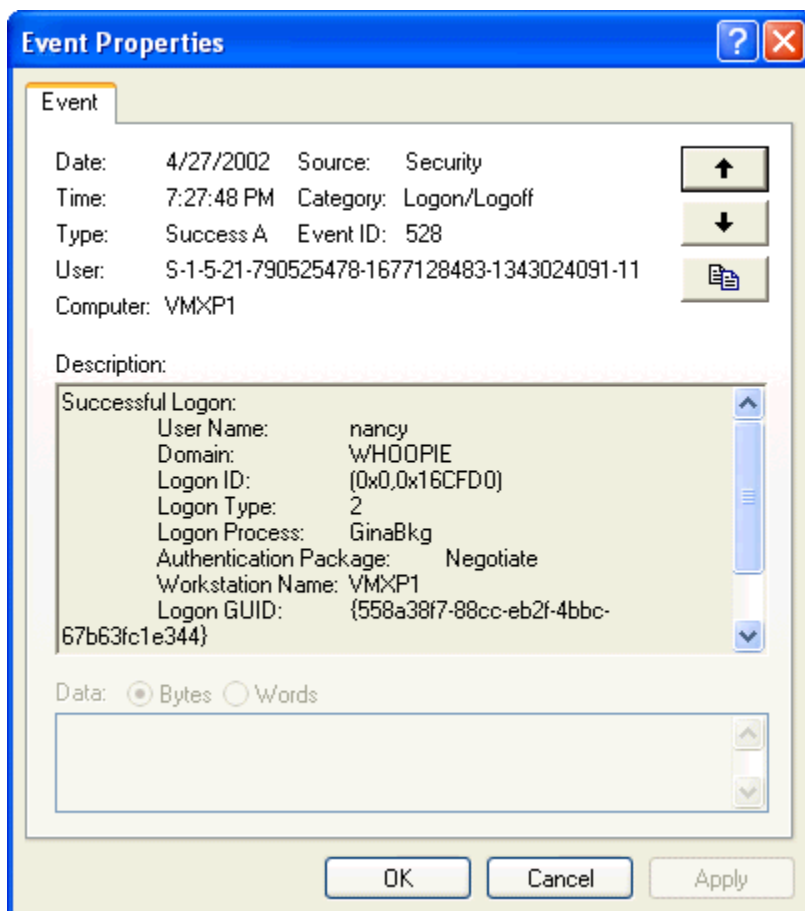


Figure 6.5: Successful interactive logon at the workstation VMXP1 by Nancy.

On the domain controller, we will find events that record the Kerberos ticket requests and a 528 event with a logon type of 3 for network logon. Successful Kerberos events are 672, authentication ticket request, as Figure 6.6 shows.

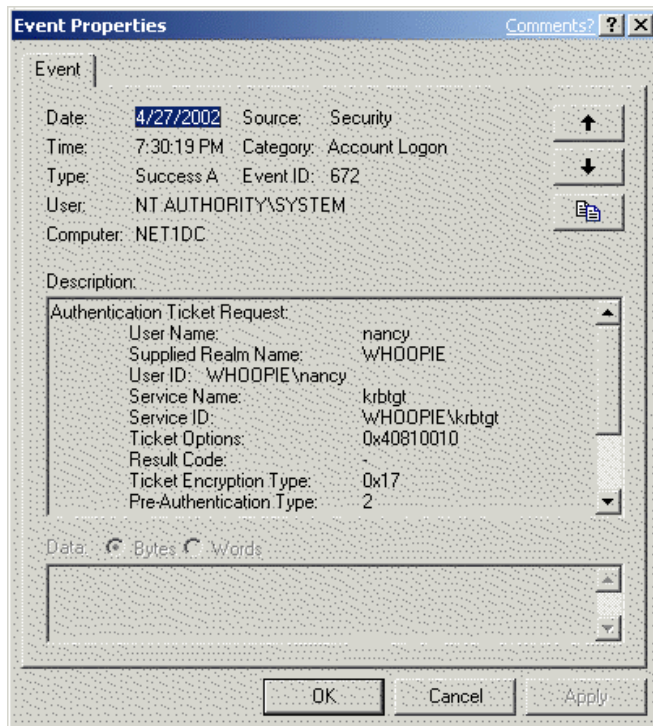


Figure 6.6: A successful authentication ticket requests indicates the user's credentials are in order.

As Figure 6.6 shows, the user is identified by presenting proper credentials (user ID and password). As Figure 6.7 illustrates, event 673 shows a service ticket request in which the user requests a service ticket to use the workstation desktop.

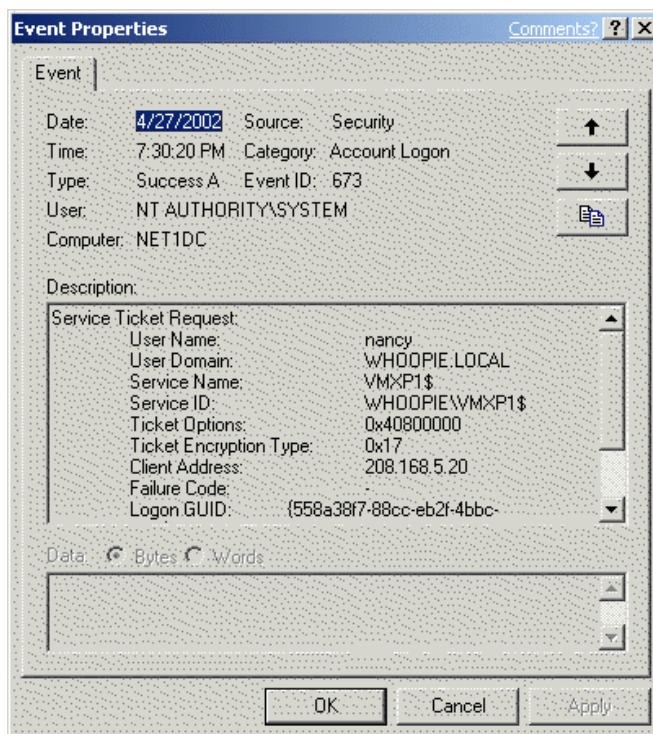


Figure 6.7: A successful service ticket request is required before the user can access his or her desktop.

In this figure, note the logon ID. This number is unique for each logon session. Later in the day when the user logs off, the same logon ID will be used (as Figure 6.8 shows). If a user performs multiple logons and log offs at the same computer, each log on can be matched to its log off event by using the logon ID. When audit logs are consolidated, several logon IDs may be present for the same user. They will represent different logon sessions, either at the same computer or at multiple computers. Connecting logon events by comparing logon IDs will keep things straight. This information might be important if you are trying to determine the timeframes at which a particular user was actually online.

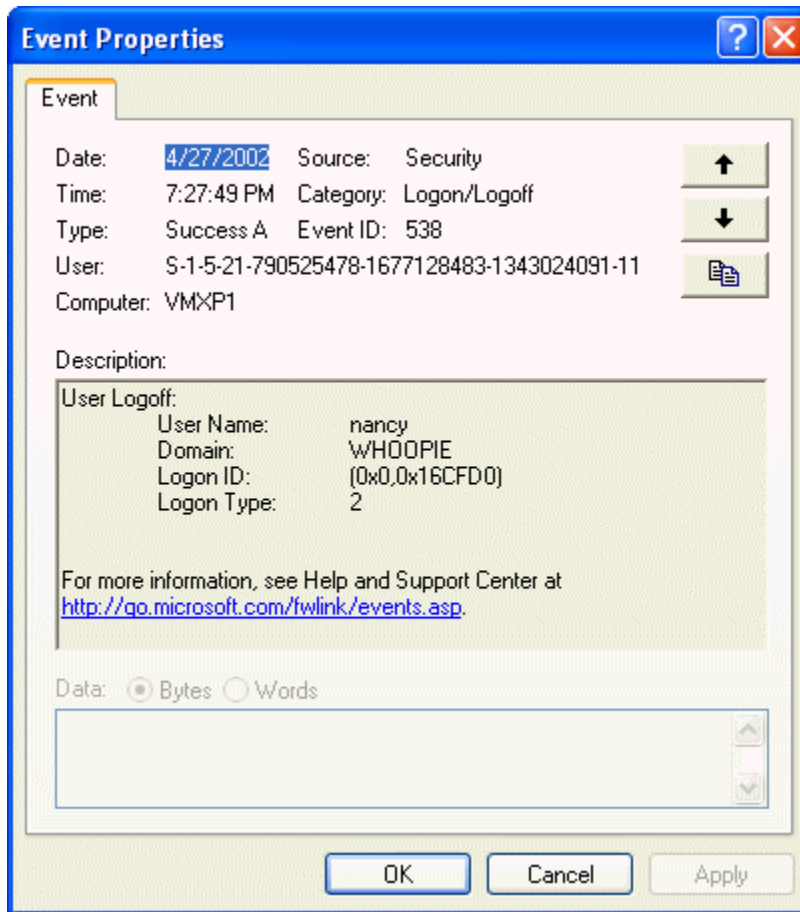


Figure 6.8: Event 538 indicates a successful logoff.

Chapter 7: Remote Access

Q 7:2: What is the Session Initiation Protocol?

A: Session Initiation Protocol (SIP) is an emerging Internet standard described in Request for Comments (RFC) 2543. The SIP signaling protocol is used to establish, create, and manage sessions between communication devices such as PCs, PDAs, and smart phones. It provides the benefits of presence (who is online and what is their status), notification (you've got a call), request (do you want this connection?), and mobility (can find you anywhere on any device). Many of you are familiar with Microsoft's Messenger and other instant messaging (IM) products that provide these features. SIP, however, can be used for much, much more. Because developers are able to depend on devices built to the standard, applications such as conferencing, customer relationship management (CRM), call centers, and others yet unimagined can be built. SIP allows the integration of location via Universal Resource Identifier (URI—defined in RFC 2396) and allows the integration of messaging, multimedia conferences, and telephony. The components of SIP are:

- URI—Uniquely identifies users
- Call_ID—Globally unique identifier (GUID) that identifies a particular conversation. It includes an undue cryptographically random number generated by SIP and the URI of the client.
- Client software (such as IMs)
- Registration servers—Store URIs.
- Proxies—To pass on communications.
- Gateways—Integrate with older phone-based communication systems and with the older h.323 telephony standard.



Server and gateway products are available from Tellme, Webley Systems, WorldCom, and Level3 Communications. Microsoft .NET Server will also provide SIP components.

Although the ability to communicate across the Internet in real time to share files, applications, and video is nice, there are some serious security concerns. How can you ensure that only those invited to the conference can join? Can conversations and shared documents be kept private? How can you be sure that the participants are who they say they are? What kind of information might an attacker gain by tracing how and who is talking to whom? These concerns have answers within the standard.



Internet Engineering Task Force RFC SIP Security

Four security issues are addressed in the SIP specification:

- Message integrity, authorization, and authentication
- Message privacy
- Route privacy
- Identity privacy

Message Integrity, Authorization, and Authentication


In a perfect world, accident or attack would never modify the contents of a message between sender and recipient and invalid attempts to redirect messages or spuriously terminate them would not be an issue. Because the perfect world does not exist contingencies must be made to ensure message integrity and validate users, computers, and requests. Checksums and timestamps can be used to ensure message integrity. The checksum ensures that the message has not changed, and the timestamp helps to prevent successful replay of attacks. Likewise, authentication mechanisms can be used to prove the identity of the proxy, sender, and recipient. In addition to Secure Sockets Layer (SSL), SIP supports HTTP authentication mechanisms such as basic and digest. Pretty good privacy (PGP) can also be used.

After identification of user or proxy, authorization is checked. Authorization can be set to include the client-side rights of the caller (sender) to call, type of call, entry to conference, or use of advanced features such as application sharing. Authorization to use specific proxies for particular purposes can also be set and based on identification of caller. IPSec can be used for authentication between client and proxy and proxy to proxy.

Responses that may redirect calls or terminate them should be authenticated. Such calls, if not validated, can lead to session hijack, premature termination, and denial of service. Unauthenticated responses will be dropped.


Message Privacy

Session requests, responses, and message bodies might include sensitive information. Message bodies might also include copies of encryption keys. SIP systems can encrypt the message body and those parts of the message header that are not necessary for routing or identification. End-to-end encryption is accomplished by using the user keys.

 Although a common explanation of public key cryptography explains that the recipient's public key is used to encrypt the message so that the recipient's private key can be used to read it, such is not usually the case. Instead, to improve performance, the message is usually encrypted using a session key, which is then encrypted with the public key of the recipient. To decrypt the encrypted session key, the private key of the recipient is used. Then the session key can be used to decrypt the message.

Should Alice, for example, send a secure IM message to Bob, the message is first split into two parts—the header fields that are to be encrypted and the message body make one part and the other part is a short header that remains in clear text. A session key may be used to encrypt the header fields and message body. Next, Bob's public key is used to encrypt the session key. The

encryption field of the header is used to specify the type of encryption used, and information about which key to be used in the response is included. Bob's client receives the message and uses Bob's private key to decrypt the session key, which then is used to decrypt the message. If Bob sends a response, it should be encrypted with a session key and Alice's public key used to encrypt the session key.

 Because Alice may have many public key pairs, the secure IM message will let Bob know which public key to use. The public key may be included in the message as well. This inclusion avoids confusion. When Bob's response, encrypted with Alice's public key, returns, Alice's client won't have to guess which of her private keys to use to decrypt it.

Typically, encryption will occur at the client, but implementation may provide encryption by the SIP proxy if clients are not capable or if administrative policy requires enforcement of security policies. Users of end-to-end encryption should note that it only provides privacy—there is no guarantee that messages come from the sender indicated.

Route Privacy

Within the SIP message header, the VIA field includes routing information. VIA is not an acronym, it merely serves to indicate that the field holds information about how the data should go (it should go *via* the route included). Hop by hop, additional VIA statements are added until the end of a message route where the path taken is revealed fully within the message. This information can be used by the response, as each VIA statement can be used to find the way back to the previous routing device. One by one, addresses are followed until the response is delivered back to the sender.

This type of routing information can provide useful information to an attacker, and so hop-by-hop encryption of VIA fields is also possible. During this process, the request Hide field is used to request that VIA fields are hidden from subsequent proxies. Each proxy encrypts the VIA field address that identifies the previous proxy. The proxy can choose the mechanism because only that proxy will decrypt it. The VIA header field is cached along with an association to the header field. The index into the cache is placed into the VIA header field. It then adds its own address to the VIA field, maintains the Hide field (instructing the next proxy to encrypt this new information), and passes the message on. At each proxy, the only information that can be determined is the identification of the previous hop.

When a response must be returned, the client has the address of the last proxy in the chain. This proxy takes the address from its cache, decrypts the VIA field information (because it originally encrypted it) to discover where it must pass the message. The process is repeated at the next proxy and so on until the response has returned to the originator of the request.

Because route hiding of this type may not prevent message looping, additional information is added to the VIA field so that the proxy can tell that it has seen the message before. This ID obviously cannot be the proxy name as that would not hide the route. Instead, a secret key, timestamp, and checksum is used. A checksum can detect that the decryption returned correct information and the timestamp can help to prevent a replay attack.

☞ To ensure that VIA information continues to be encrypted proxy by proxy, messages must be sent to trusted proxies. A trusted proxy is one which is controlled by some party that you trust to configure for the security controlled mandated by your policy. This could be a proxy owned by your organization or that of a business partner or associate with whom mutual trust relationships have been developed.

Identity Privacy

In addition to route and message privacy, there is often reason to hide who is talking to whom. You can do so by using hop-to-hop encryption. IPsec, Transparent LAN service (TLS) or some other transport or network layer encryption may be used to encrypt messages as they travel between proxies. The message may be unencrypted as it travels from sender to proxy and from proxy to recipient. Hop-to-hop privacy may also be used with end-to-end encryption. Figure 7.4 illustrates the difference between end-to-end and hop-to-hop encryption. In the top half of the figure, all communications from the first PC to the last are encrypted (end-to-end encryption). The bottom half of the figure represents hop-to-hop encryption. Data is only encrypted between any two points on the path.

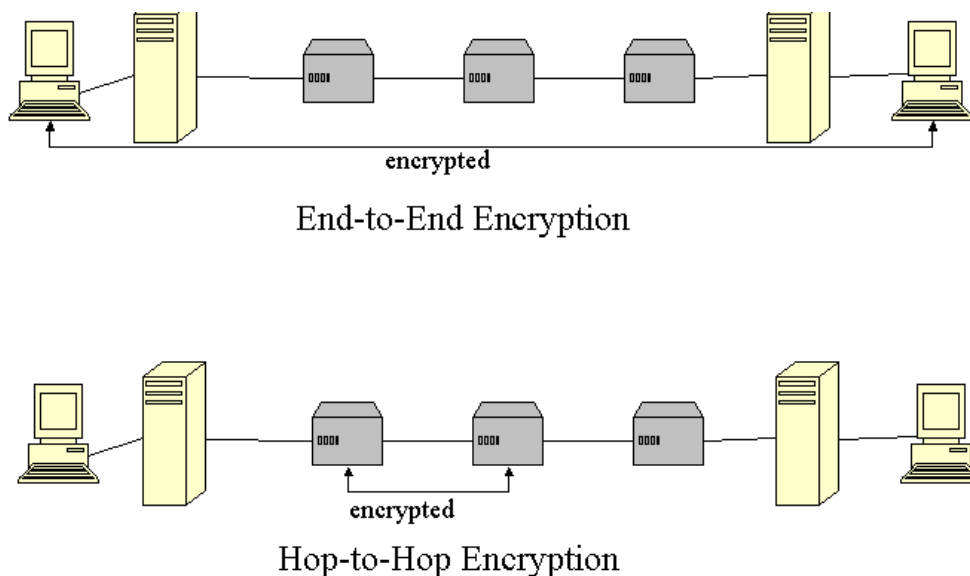


Figure 7.4: End-to-end encryption vs. hop-to-hop encryption.

Miscellaneous Uses of Cryptography

Cryptography is used throughout the specification to ensure privacy or to help aid in mitigating attack. Many users will have multiple devices and yet a single SIP URL (the user address or location registered for the user). A single SIP URL ensures that the user can be located wherever he or she may be and may use any device desired or available. Smart phones, for example, may be more convenient in airport terminals if the message is text, while PCs are more useful for multimedia conferencing. However, the possibility for misdirection by chance or by malicious action is possible. To ensure consistent communication and to mitigate the possibility of a hijacked session, a tag in the *from* section of the message header is used. This tag is cryptographically random and globally unique and identifies the return of responses to the correct device.

Microsoft Implementation

You might have already recognized some of the SIP communication specifications in Messenger and other IM applications. .NET Messenger Service integrates SIP on the Internet, and Microsoft Exchange 2000 Instant Messaging Server utilizes SIP on the intranet. Although security implementation is lacking in current versions of Messenger, SIP is integrated into .NET Server to provide secure real-time communication technology. Real-time communication is based on the Real Time Protocol (RTP) for Voice over IP (VoIP).

Microsoft RTC Server is incorporated in .NET Server as a basic infrastructure service and is fully compliant with the IETF standard. Standard services, a SIP proxy, and registration extension module are available by default, and a rich API is present so that programmers may build even more exotic applications. The registration extension module will register users and track online information. Messaging can be federated across boundaries, and therefore will be available to provide secure communications between forests. Authentication can be configured within the property pages of the RTC Server, as Figure 7.5 shows.

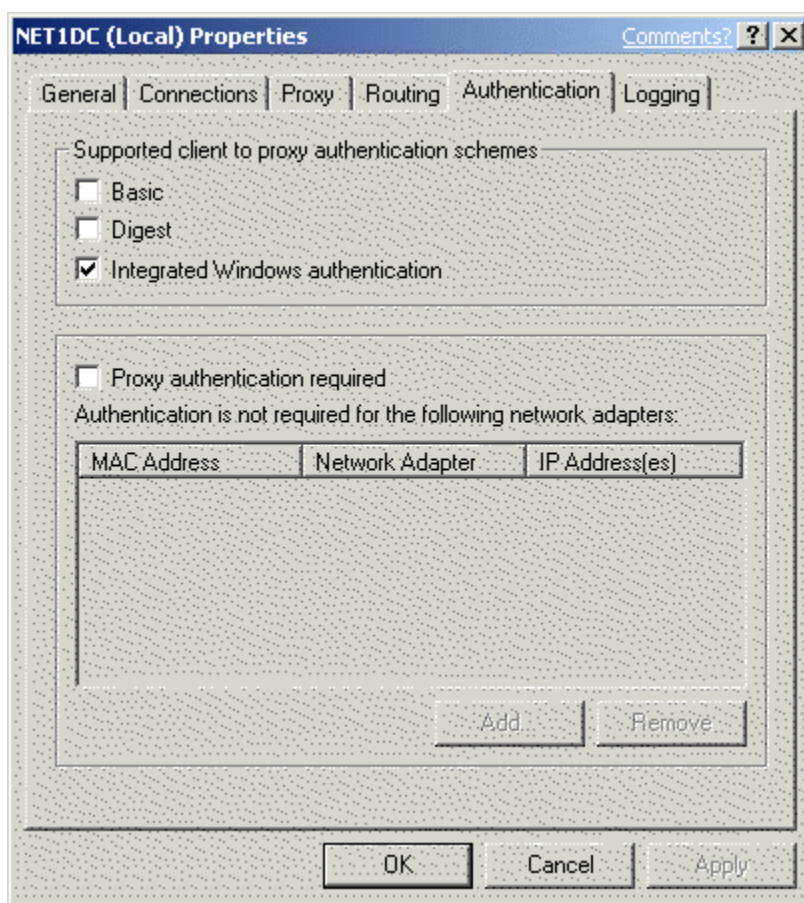



Figure 7.5: Authentication can be configured on the property pages of the RTC Server.



Think of the possibilities! Microsoft mentions real-time help and support via phone available at the click of a Help button, purchasing applications that can access and retrieve order status in real-time and make that information available to the customer, and network messaging to users—say an instant notification that the Exchange server is going down for maintenance. But what can you envision? I think of instant alerts to my pager about my flight status so that I don't run thorough the airport only to discover that the flight has been delayed or cancelled. Or maybe notice that my room at the hotel is available for early check-in and that it's non-smoking and on the quiet side of the hotel as I requested. What are your ideas?

 For more information about SIP, check out the following Web sites:

 <http://www.microsoft.com/windows2000/server/evaluation/business/rtcfaq.asp>

 <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/prodtechnol/winxpro/evaluate/nsid01.asp>

 <http://www.rfc-editor.org/rfcsearch.html>

Chapter 8: Security Tools, Mechanisms, and Emerging Issues

Q 8.2: Managing multiple Windows boxes is like herding cats. It's hard to keep up with which systems have up-to-date patches and which still need them. How can I figure this out?

A: Microsoft has three free tools that may help you: HFNetChk, QChain, and Microsoft Baseline Security Analyzer (MBSA). HFNetChk is a command-line tool that can analyze a Windows NT, Windows XP, or Windows 2000 (Win2K) computer and report missing hotfixes. QChain is a tool to help you apply multiple hotfixes to a computer at a time without excess rebooting between fixes. MBSA is a GUI tool that

- Analyzes basic security on an NT, Windows XP, or Win2K system
- Analyzes security settings for IIS (if present on the system)
- Analyzes security settings for SQL Server (if present on the system)
- Determines which hotfixes are missing
- Provides a way to immediately download and apply missing fixes.
- Scans a Windows .NET system and reports vulnerabilities (although MBSA doesn't officially support .NET)

Although a combination of HFNetChk and QChain can help you keep systems up to date with hotfixes, they do nothing about basic system security analysis and require some sophistication to run. MBSA, however, provides a quick and easy security assessment and an immediate way to update a system. Although MBSA might not answer all your questions and is not exactly the solution for large enterprises, it is a valuable aide for the harried systems administrator.

Analyzing the Analyzer: MBSA Internals

MBSA is at heart a GUI interface for HFNetChk. Rather than learning how to operate a command-line tool, users and new administrators can simply start a GUI product and click their way to security—or so they may think. In reality, of course, they should be thinking about what they are doing.

As with any product, there are three things to understand in order to use it. Before you run MBSA, and certainly before you attempt to act on any of the results, consider the following:

- What is necessary to run it?
- How do you run it?
- How should you interpret and act on the results?

Understanding Prerequisites

When MBSA is installed, you are asked if you want to view a readme file (the file is also available pre-installation at the download site) that explains that MBSA is simple to run but does have basic requirements. MBSA can only be hosted by Win2K and Windows XP systems, but can be used to scan NT and unofficially .NET systems. Table 8.1 lists OS specifics.

OS	Host	Scan Locally	Scan Remotely
Windows 9x, ME	No	No	No
NT Workstation	No	Yes	Yes
Win2K Professional	Yes	Yes	Yes
Win2K Server/Advanced Server	Yes	Yes	Yes
Windows XP Professional	Yes	Yes	Yes
Windows XP Professional using simple file sharing	Yes	Yes	No
Windows XP Home Edition	Yes	Yes	No
.NET Servers	No	No	Yes (but not officially supported in version 1)

Table 8.1: Where to run and what to scan using MBSA.

Before a successful scan can be run, the user running the scan must have local administrative privileges on the system to be scanned. Thus, if you are running a scan, you must be logged on to the scanning machine using an administrative account in the domain of the scanned machine. The application will not prompt you to enter alternative user account names and passwords if your currently logged on account is not valid. If you want to scan a subnet, you must have administrative privileges on every machine in the subnet or it will not be scanned. The scanning system must meet several requirements:

- Win2K or Windows XP system
- Internet Explorer (IE) 5.01 or later or other versions of IE and an XML parser
- XML parser (MSXML version 3.0 SP2)—An XML parser is part of IE 5.01 or later; however, if you're running an earlier version of IE, you can install version 3.0 SP2 of the XML parser during the installation of MBSA or obtain it from the Microsoft Web site.
- If you are going to remotely scan IIS computers, you will need to install IIS Common Files on the computer from which the scan will be made.

The remotely scanned system must meet the following requirements:

- NT 4.0 SP4 and later, Win2K, or Windows XP Professional
- Server service must be running
- Remote registry service must be running on Windows XP and Win2K systems
- .NET can be scanned, but scanning of beta products is not officially supported

Obviously, the SQL Server, IIS, and Microsoft Office scans will only be completed if these products are located on the scanned computer.

Getting Secure the MBSA Way

During the installation, you may choose to install MBSA just for your use or for anyone to run. However, MBSA can only scan computers for which its user has administrative privileges, so installing for anyone to run is the equivalent of installing for any administrator to run, not just anyone. MBSA is started by clicking the desktop icon or running it from Start menu, Programs, Microsoft Security Baseline Analyzer. It can also be run from the command line. Next, you can make scans or review scan reports. Three scanning choices are possible: scanning the local computer, scanning one remote computer via IP address, or scanning a range of computers or an entire domain. The scan reports, one for each computer scanned, are then saved to the local machine in the %userprofile%\SecurityScans folder and can be reviewed at any time.



Scan reports are stored in the user profile of the user who performed the scan. Thus, they are not normally viewable by another user of the computer. However, precaution should be taken to ensure the security of the scan. Information about the status and configuration of any computer should not be made widely available. Best practices include performing the scan remotely for most systems and using Encrypting File System (EFS) to secure the reports. Be aware, however, that extremely sensitive systems do not allow remote administration, and thus, should not be configured (by enabling the remote registry service) for remote scanning. These systems should be reviewed for proper security configuration and patch updates from the console.

Each report consists of the results of reviews in several areas along with details of what was scanned and instructions about how to correct the situation. General security settings are marked with a red x if they are very weak, medium issues with a yellow x, and good settings are marked with a green check mark, as Figure 8.4 shows. Yellow x's may also indicate an area which MBSA cannot conclusively determine. For example, some hotfixes should only be installed in certain circumstances. The basic areas scanned are:

- Service pack and patch updates—MBSA uses a version of HFNetChk to check registry key, file versions, and file checksums against an XML file at the Microsoft download center Web site to determine whether the computer is up-to-date with service packs and updates. Each missing hotfix is reported, and a link to associated Knowledge Base articles and the download location is provided. Administrators may access, download, and apply a missing patch directly from within MBSA. No attempt however is made to fully explain the patch or to link multiple patch requirements into a single install.
- Passwords—MBSA checks the local Security Accounts Manager (SAM) database for weak passwords by scanning for accounts for which the password is blank; is the same as the username; is the same as the machine name; or equals password, admin, or Administrator. This check is not made on domain controllers.



- Account checks—Accounts are checked for password expiration and the number of accounts for which passwords do not expire is reported.
- Administrator group check—The number of members in the Administrators group is reported. MBSA considers more than two administrators on a local machine questionable.
- Guest account—The status of the guest account is checked. Having the guest account disabled is a plus.
- Restrict anonymous—MBSA considers a *restrict anonymous* setting of 2 to be the goal. Fortunately, a brief explanation of these settings and why setting *restrict anonymous* to 2 is not a good idea on domain controllers.
- File System—The file system on all drives is checked to see if it is NTFS.
- Auto Logon—The system is checked to make sure auto logon is not configured.

The screenshot displays the Microsoft Baseline Security Analyzer (MBSA) interface. The main window is titled 'View security report' and shows a summary of a scan performed on 'Sol - Sunshine' on 4/26/2002 at 11:01 PM. The security assessment is 'Severe Risk (One or more critical checks failed)'. The 'Windows Scan Results' section lists several vulnerabilities:

Score	Issue	Result
✗	Windows Hotfixes	8 hotfixes are missing or could not be confirmed. What was scanned Result details How to correct this
✗	Restrict Anonymous	Computer is running with RestrictAnonymous = 0. This level prevents basic enumeration of user accounts, account policies, and system information. Set RestrictAnonymous = 2 to ensure maximum security. What was scanned How to correct this
✗	Administrators	More than 2 Administrators were found on this computer. What was scanned Result details How to correct this
✗	Password Expiration	Some user accounts (7 of 13) have non-expiring passwords. What was scanned Result details How to correct this
✓	File System	All hard drives (1) are using the NTFS file system. What was scanned Result details

The interface also includes a left-hand navigation pane with options like 'Welcome', 'Pick a computer to scan', and 'View a security report'. At the bottom, there are buttons for 'Previous security report' and 'Next security report'.

Figure 8.4: MBSA returns information about common vulnerabilities and how to fix the situation.

Other areas are evaluated, including:

- Auditing—Is it enabled? Are both logon success and failure checked?
- Services—Are any of the especially dangerous services (RAS, Telnet, FTP, WWW, or SMTP) installed?
- Shares—Which shares are present and what are the permissions settings on them?
- OS version

If IIS or SQL Server is present on the machine and the check boxes to scan them have not been cleared, MBSA will also look for common vulnerabilities in those products.

Password checks can take a very long time if multiple accounts exist. In addition, if account lockout is set, it is possible that the multiple password tests—because most should and will fail—may lock out accounts. MBSA documentation suggests that account lockout is changed so that such will not happen. However, early reports of account lockout were reported. Password checks are not run against domain controllers.

Now What?

There is actually quite a lot of useful security information available in a scan, but MBSA has the same problems that any vulnerability checker has. Users of the scanner must realize that there are no hard and fast security rules. When the scanner tags having more than two administrators as a vulnerability, it cannot take into account the number of computers that are managed in a network. Networks with hundreds of computers will require more than two administrators to manage them; networks with thousands of nodes even more. Likewise, hotfixes should be evaluated on their own merit. If a hotfix should only be applied in some circumstances, it is difficult for a vulnerability checker to indicate so in a manner that will make everyone think before applying it. A third issue also involves the application of hotfixes. MBSA provides a click through so that the administrator can locate, download, and apply a hotfix immediately. However, when multiple hotfixes are needed, there is no way to automatically build a chain. Savvy administrators will use the report to determine status of the machine, then use other methods to bring the system up to date.

Nevertheless, MBSA is a useful tool. Home users and network administrators can evaluate basic security against a common norm. They can, and should, make decisions about which parts of the report are valid for themselves.