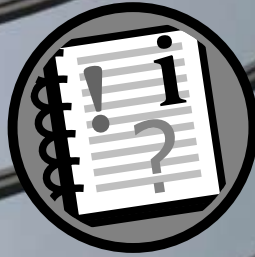


Unlocking Microsoft's® most arcane security tool.



THE WINDOWS SERVER 2003 SECURITY LOG REVEALED

By Security Expert

RANDY FRANKLIN SMITH

Founder of the *Security Log Secrets* Seminars

The Windows Server 2003 Security Log Revealed

The Windows Server 2003 Security Log Revealed

by Randy Franklin Smith, CISA, SSCP, Security MVP

Copyright © 2005-2006 by Randy Franklin Smith and Monterey Technology Group, Inc.

All rights reserved, including the right to reproduce this book or portions thereof in any form whatsoever.

For information address Monterey Technology Group, Inc.,
www.montereytechgroup.com

No part of this book is guaranteed to prevent intrusions. We make no guarantee as to the fitness, safety or efficacy of any part of this book. Use your knowledge and professional judgment with guidance in this book as with all other information you read. We recommend testing new technology and system changes before applying them to production environments.

The Windows Server 2003 Security Log Revealed

The Windows Server 2003 Security Log Revealed

In loving memory of
Keo
the friend who made me a dog lover

The Windows Server 2003 Security Log Revealed

Table of Contents

Chapter 1 Introduction	2
Chapter 2 Audit Policies and Event Viewer	4
Chapter 3 Understanding Authentication and Logon	16
Chapter 4 Account Logon Events	23
Chapter 5 Logon/Logoff Events	40
Chapter 6 Detailed Tracking	47
Chapter 7 Object Access Events	52
Chapter 8 Account Management	63
Chapter 9 Directory Service Access Events	68
Chapter 10 Privilege Use Events	80
Chapter 11 Policy Change Events.....	85
Chapter 12 System Events	91
Chapter 13 Getting the Most from the Security Log	93

Chapter 1 Introduction

First, the Bad News...

You can glean a wealth of information from the Windows Security log, but the mechanism isn't without problems. Each Windows computer—including domain controllers (DCs)—has a discrete Security log. Each DC logs security events according to the activity that it sees; this information doesn't replicate to the other DCs in the domain. Windows has no native capability to centrally collect, analyze, monitor, report, and archive the many Security logs that exist throughout your network.

Another problem is that the log's event descriptions and codes are cryptic and poorly documented by Microsoft. If that weren't bad enough, Microsoft eliminates, merges, and changes the meaning of event IDs from one version of Windows to the next. In addition, the order of strings in a given event's description sometimes changes between Windows versions. (I'll go into more detail about description strings later.) These changes can really throw a wrench in the works when you upgrade one or more systems after having set up reports or rules based on event ID or the position of a string. In this book (as well as in my free Security Log Encyclopedia at <http://www.ultimatewindowssecurity.com/encyclopedia.html>), I endeavor to document such changes.

This Book

In Chapter 2, I'll introduce you to the Windows audit policy (including the relationship between audit policies and audit categories), the Microsoft Management Console (MMC) Event Viewer console, and the format of security events. Even if you're an experienced Windows Server administrator, I recommend at least scanning this chapter. I've included a few valuable nuggets that might well be new to you.

Chapter 3 introduces you to the concepts of Windows authentication and logon (which serves as the foundation for subsequent chapters), then delves into the closely related Account Logon and Logon/Logoff audit categories. Chapter 4 discusses how Windows logs authentication activity by using Account Logon events, and Chapter 5 deals with logon events in the Logon/Logoff category.

In Chapter 6, we examine the Detailed Tracking category, and I show you how to track programs that users execute. In Chapter 7, you'll find out how to monitor file-system activity and access attempts on other types of objects by using the Object Access category. Chapter 8 shows you how to audit changes to users, groups, and computer accounts by tracking Account Management events,

and Chapter 9 reveals how to use Directory Access events to track changes to Active Directory (AD) objects such as organizational units (OUs) and Group Policy objects (GPOs). Chapters 10, 11, and 12 deal with the Privilege Use, Policy Change, and System Event categories, respectively.

At the End of the Day...

Windows has the ability to generate a detailed audit record of security events on each system, but exploiting that information is a lot like mining low-grade ore, which has to be subjected to a laborious refining process before you can get to the gold. Unless your needs are limited to occasional investigations, you'll want some type of automated solution for collecting, monitoring, reporting, and archiving the Security logs that are scattered throughout your network. There are many such tools on the market. The two most important criteria in choosing which product to use are whether the product can meet your scalability needs and whether it provides the ability to build sophisticated alerts and rules based on specific string positions within an event's description. My contacts at Microsoft indicate that this capability will become even more important for future versions of Windows.

Chapter 2

Audit Policies and Event Viewer

A Windows system's audit policy determines what type of information you'll find about that system in the Security log. Each Windows system on your network has nine audit policies (Windows NT has only seven), which can be enabled or disabled:

- *Audit account logon events*
- *Audit account management*
- *Audit directory service access*
- *Audit logon events*
- *Audit object access*
- *Audit policy change*
- *Audit privilege use*
- *Audit process tracking*
- *Audit system events*

An event in the Windows Security log is either type Success or type Failure. When you enable an audit policy you have the choice of enabling it for success events, failure events, or both, depending on the policy. All nine audit policies generate success events but only some of the policies generate failure events. Don't fall for the recommendation to enable failure events only for each category. A common misconception is that a failure-only audit policy will alert you to all suspicious events. In reality, many of the most important events in the Security log are success events such as changes to critical user accounts and groups, account lockouts, and changes to security settings. To view a system's audit policy settings, you can open the Local Security Policy console on the computer and maneuver to Security Settings\Local Policies\Audit Policy, as Figure 2-1.



Figure 2-1 The nine audit policies in Local Security Settings

When you open one of the audit policies, you may or may not be able to modify it, depending on whether the policy has been defined in a GPO that has been applied to the local system. If the computer is a member of an AD domain, the computer automatically applies appropriate GPOs from the domain. Any policy defined in a GPO overrides policies defined in the system's local policy object and becomes the effective setting.

Windows Server 2003 and Windows XP always display the effective settings when in the Local Security Settings dialog box. (If you can modify the setting, the policy isn't defined in any applicable GPOs defined in AD.) Windows 2000's Local Security Policy console puts the system's local setting from effective setting in separate columns.

I recommend that you use the Local Security Policy console only for viewing a system's audit policy—not for configuring it. As with all security settings, the best practice is to use Group Policy to centrally manage your audit policy.

Versions earlier than Windows 2003 disable all nine audit policies by default. Windows 2003 enables a few categories by default but the options selected make no sense from a security standpoint. Use Group Policy to push out the audit policy you want rather than relying on default settings.

Let's drill down just a bit into each of the audit policies.

Audit Account Logon Events

Microsoft should have named the *Audit account logon events* policy *Audit authentication events*. On DCs, the policy tracks all attempts to log on with a domain user account, regardless of where the attempt originates. If you enable

this policy on a workstation or member server, it will record any attempts to log on by using a local account stored in that computer's SAM.

Audit Logon Events

The *Audit logon events* policy records all attempts to log on to the local computer, whether by using a domain account or a local account. On DCs, this policy records attempts to access the DC only. The policy does not, for instance, track a user who uses a domain account to log on at a workstation. (In that case, the user isn't logging on to the DC; the DC is simply authenticating the user.) To track *all* domain account authentication, you should use *Audit account logon events*.

Audit Account Management Events

The *Audit account management events* policy, which you can use to monitor changes to user accounts and groups, is valuable for auditing the activity of administrators and Help desk staff. This policy logs password resets, newly created accounts, and changes to group membership. On DCs, the policy logs changes to domain users, domain groups, and computer accounts. On member servers, it logs changes to local users and groups.

Audit Directory Service Access

The *Audit directory service access* policy provides a low-level audit trail of changes to objects in AD. The policy tracks the same activity as *Audit account management events*, but at a much lower level. By using this policy, you can identify exactly which fields of a user account or any other AD object were accessed. *Audit account management events* provides better information for monitoring maintenance to user accounts and groups, but *Audit directory service access* is the only way to track changes to OUs and GPOs, which can be important for change-control purposes.

Audit Object Access

The *Audit object access* policy handles auditing access to all objects outside AD. The first use you might think of for the policy is file and folder auditing, but you can use it to audit access to any type of Windows object including registry keys, printers, and services. Furthermore, auditing access to an object such as a crucial file requires you to enable more than just this category; you must also enable auditing for the specific objects you want to track. To configure an object's audit policy, open the object's Properties, select the Security tab, click Advanced, and then select the Auditing tab, which Figure 2-2 shows. **Be warned: This policy can really bog down your server if you enable it on too many objects.**

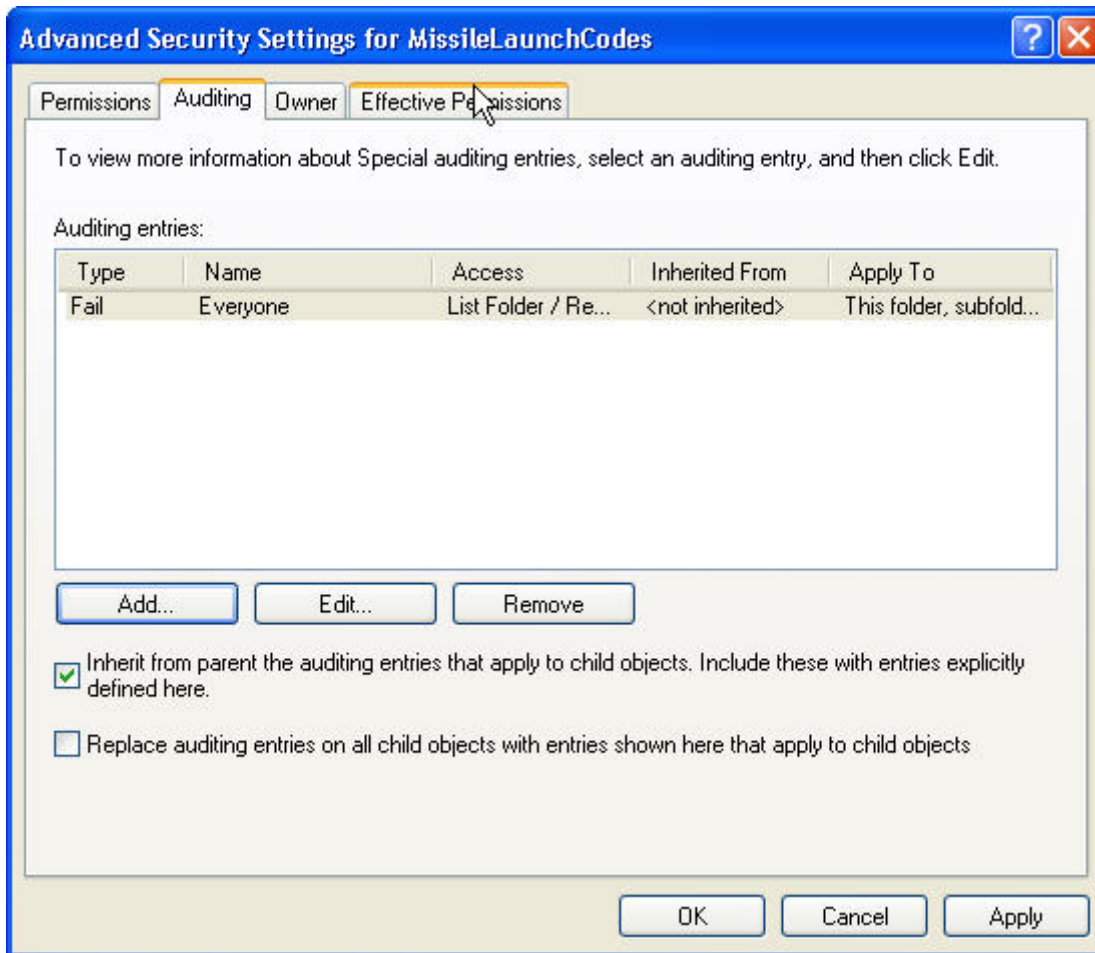


Figure 2-2 Audit policy for a folder

Audit Policy Change

The *Audit policy change* policy provides notification of changes to important security policies on the local system, such as changes to the system's audit policy or, when the local system is a DC, changes to trust relationships.

Audit Privilege Use

The *Audit privilege use* policy tracks the exercise of user rights. Microsoft uses the terms *privilege*, *right*, and *permission* inconsistently. In this policy's case, *privilege* refers to the user rights you find in the Local Security Policy under Security Settings\Local Policies\User Right Assignment. This category generates a lot of noise and I usually recommend leaving it disabled.

Audit Process Tracking

The *Audit process tracking* policy tracks each program that is executed, either by the system or by end users. You can even determine how long the program was open. You can tie this policy, *Audit logon events*, and *Audit object access* events

together by using the Logon ID, Process ID, and Handle ID fields within the various event descriptions and thereby paint a detailed picture of a user's activities.

Audit system events

The *Audit system events* policy logs several miscellaneous security events.

Event Viewer

The preceding 9 audit policies allow you to fire up the Windows auditing function. Once Windows starts sending events to the Security log you need a way to view them. The only built-in tool for accessing the Security log is the MMC Event Viewer snap-in, which Figure 2-3 shows. By default, Event Viewer displays the local computer's event logs, but you can easily use the console to view the logs of other computers on the network. To view another computer's logs, right-click the root in the details pane and select *Connect to another computer*. You will need to have the *manage auditing and security log* and *access this computer from the network* user rights on the target system.

Event Viewer shows only basic information about each event. You usually need to open an event's Properties, as Figure 2-4 shows, to see the whole story. Each event has a number of standard fields (I call them *header fields*) and a description field.

The header fields tell you the event ID, the date and time the event occurred, whether the event is a Success or Failure, and the event's source and category. All events in the Security log list the source as Security, so the Source field is pretty useless. Security events fall into nine categories, which correspond to the nine audit policies, as Figure 2-5 shows. The Category field specifies the category into which the event falls. The User field isn't usually of much value because so many events simply list SYSTEM as the user. This is especially true of events in the Logon/Logoff and Account Logon categories.

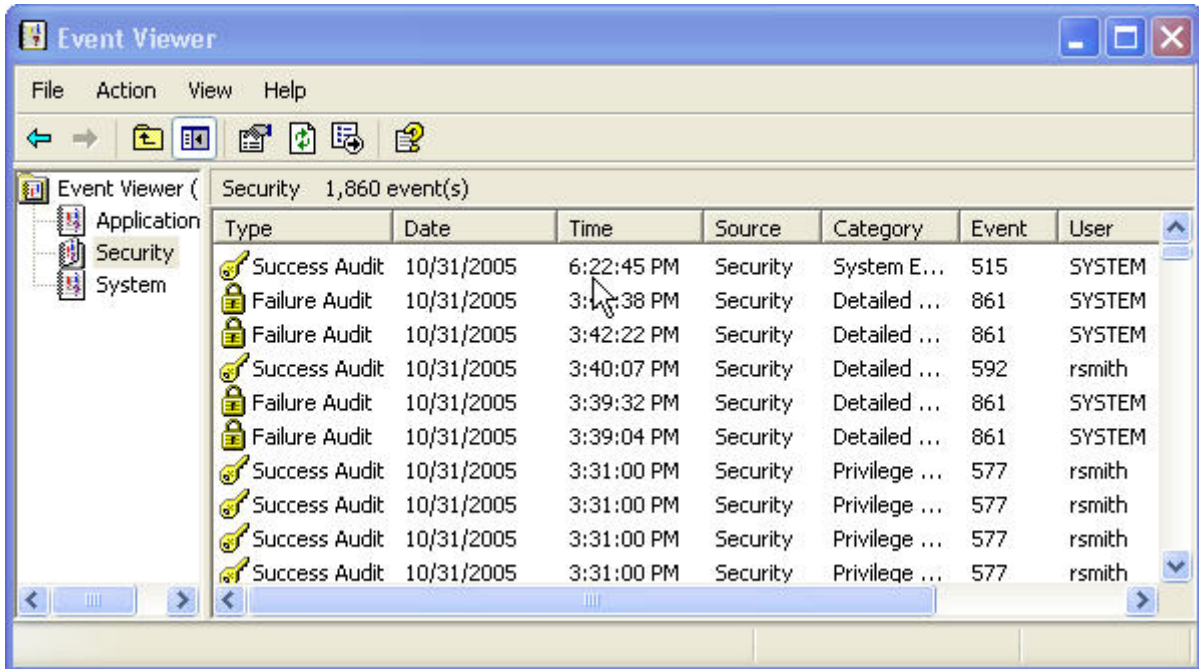


Figure 2-3 Event Viewer

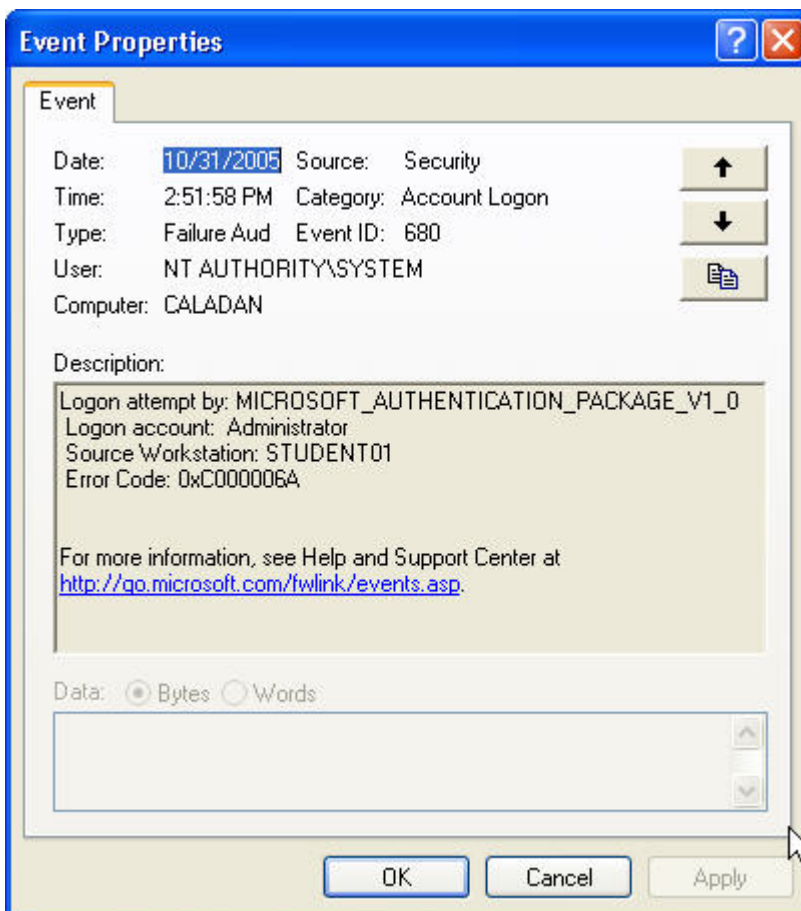


Figure 2-4 Event Properties

Audit Policies	Security Log Categories
Audit account logon events	Account Logon
Audit account management	Account Management
Audit directory service access	Directory Service Access
Audit logon events	Logon/Logoff
Audit object access	Object Access
Audit policy change	Policy Change
Audit privilege use	Privilege Use
Audit process tracking	Process Tracking
Audit system events	System Events

Figure 2-5 Audit policies and corresponding categories

You'll find the real details about an event in the Description field. When you view an event's description you are actually seeing two types of information that have been merged. Each event ID has a static description that contains defined placeholders into which dynamic strings of information connected with a particular instance of the event are merged. The easiest way to explain the static and dynamic elements of an event description is with an example. Figure 2-6 illustrates the merger of static and dynamic information for an instance of event ID 529.

Static Description	Dynamic Strings	Merged Description
Logon Failure Reason: Unknown user name or bad password User Name: %1 Domain: %2 Logon Type: %3 Logon Process: %4 Authentication Package: %5 Workstation Name: %6	Administrator STUDENT01 3 NtLmSsp MICROSOFT_AUTHENTICATION_PACKAGE_V1_0 STUDENT01	Event Type: Failure Audit Event Source: Security Event Category: Logon/Logoff Event ID: 529 Date: 10/31/2005 Time: 2:51:58 PM User: NT AUTHORITY\SYSTEM Computer: CALADAN Description: Logon Failure: Reason: Unknown user name or bad password User Name: Administrator Domain: STUDENT01 Logon Type: 3 Logon Process: NtLmSsp Authentication Package: MICROSOFT_AUTHENT... Workstation Name: STUDENT01

Figure 2-6 Event description

It's important that you understand how event descriptions work because this is where the important details about the event reside. In the case of event ID 529, you need to look at string 1 to determine the name of the user who failed to log on. String 3 tells you the type of logon that was attempted (a network logon in this example). These dynamic strings are also important when you have a Security log-management solution or are trying to analyze the log by using a utility such as Microsoft LogParser. Many of the typical alerts that you can define by using Security log-management solutions require criteria based in part on one or more strings from an event's description. In most cases, filtering based on event ID alone isn't sufficient. When designing a report based on the Security log, you'll often find that you need to parse one of the report columns from a string in the event's description. If you are shopping for a Security log-management product, make sure it provides the flexibility to create alert criteria and reports that are based on specific string numbers within the description.

Event Viewer provides basic filtering and search capabilities. To filter the displayed events, right-click Security in the details pane, select View, then select Filter to open the dialog box that Figure 2-7 shows. In this example, I've selected Security as the event source and Account Management as the category. (You must specify the source before you can select a category.) I've also filtered out Success events. If your logs are very large and filter updates are taking a long

time, try further limiting the filter by specifying a From and To date and time range.

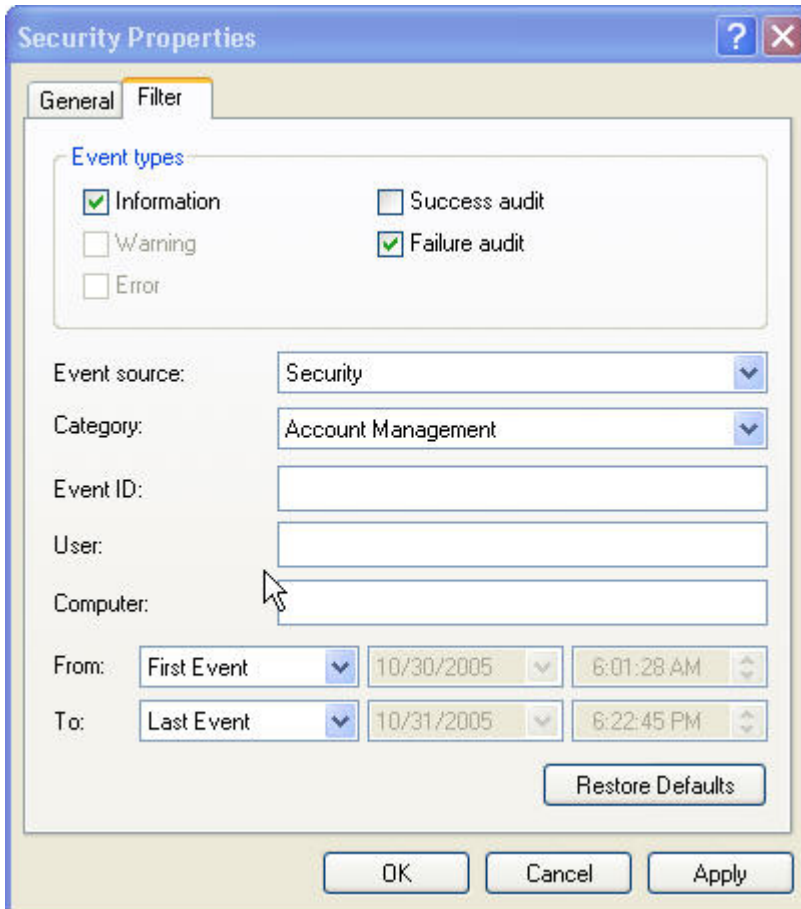


Figure 2-7 Event Viewer Filter

The only other useful analysis feature in Event Viewer is the Find option. Right-click Security, select View, then select Find. Figure 2-8 shows a search for instances of event ID 590 (which identifies executed programs) that contain “excel” in the description. These Find criteria will locate the next occurrence of Microsoft Excel being started. Unfortunately, you can’t specify precise string numbers in the Find criteria. You can look only for the occurrence of a sequence of characters. You’ll find the same limitation in many Security log-management solutions. For instance, finding all occurrences of event ID 529 with a logon type of 2 can be problematic unless you search for “Logon Type: 2”.

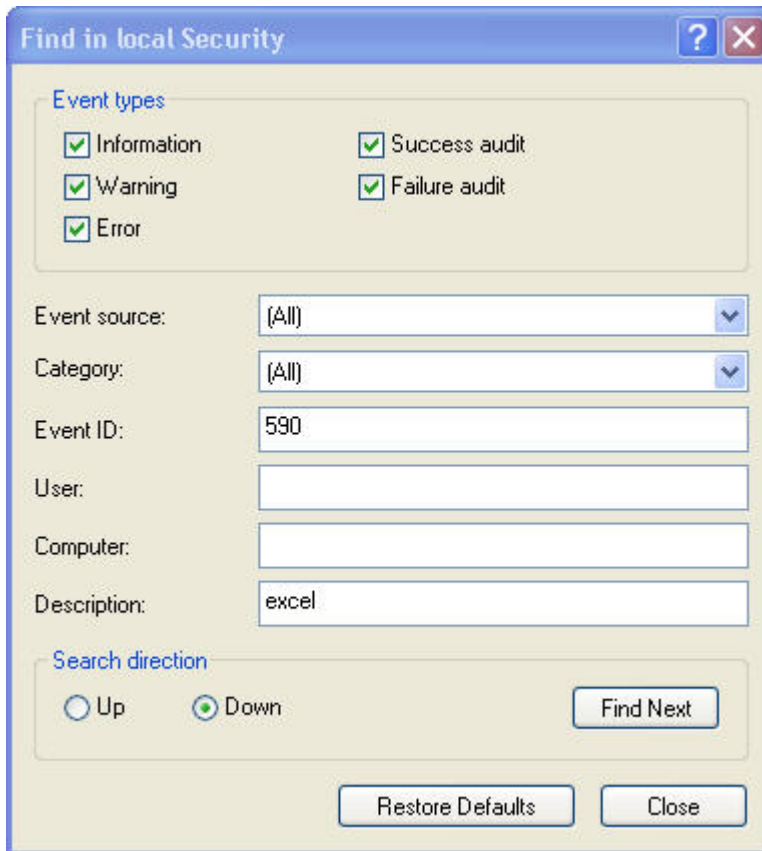


Figure 2-8 Event Viewer's Find option

Aside from using Event Viewer to view security events, you use it to configure the maximum size of the Security log. Right-click Security in the details pane, then select Properties to open the log's Properties dialog box, which Figure 2-9 shows. Windows will not grow the log beyond the size you specify. I recommend that you set the Maximum log size to no larger than 199 megabytes; 200MB seems to be the point at which Windows gets a bit flakey in terms of stability and performance.

No matter what maximum size you configure, the log will eventually reach it. You can configure Windows to do one of three things at that point. I recommend that you choose the *Do not overwrite events (clear the log manually)* option because if you do, Windows will just stop logging events when the log reaches its maximum size. (You can actually use the `CrashOnAuditFail` option to configure Windows to crash if this happens, but I don't recommend it for most commercial scenarios. See <http://support.microsoft.com/kb/823659> for more details on `CrashOnAuditFail`.) You run a similar risk by using the *Overwrite events older than X days* option. If events pour in so fast that the log reaches maximum size before any events expire, Windows stops logging events until some do expire. That leaves the *Overwrite events as needed* option, which I select for nearly every project.

Note that the Security Properties dialog box also displays the log file's location and current size, as well as various dates and times associated with the log. From this dialog box, you can also clear the log. When you clear the Security log, Windows immediately logs event ID 517. Although event ID 517 is part of the System Events category, Windows always logs the event, regardless of your audit policy. When you clear the log, Event Viewer gives you the option of saving a copy first.

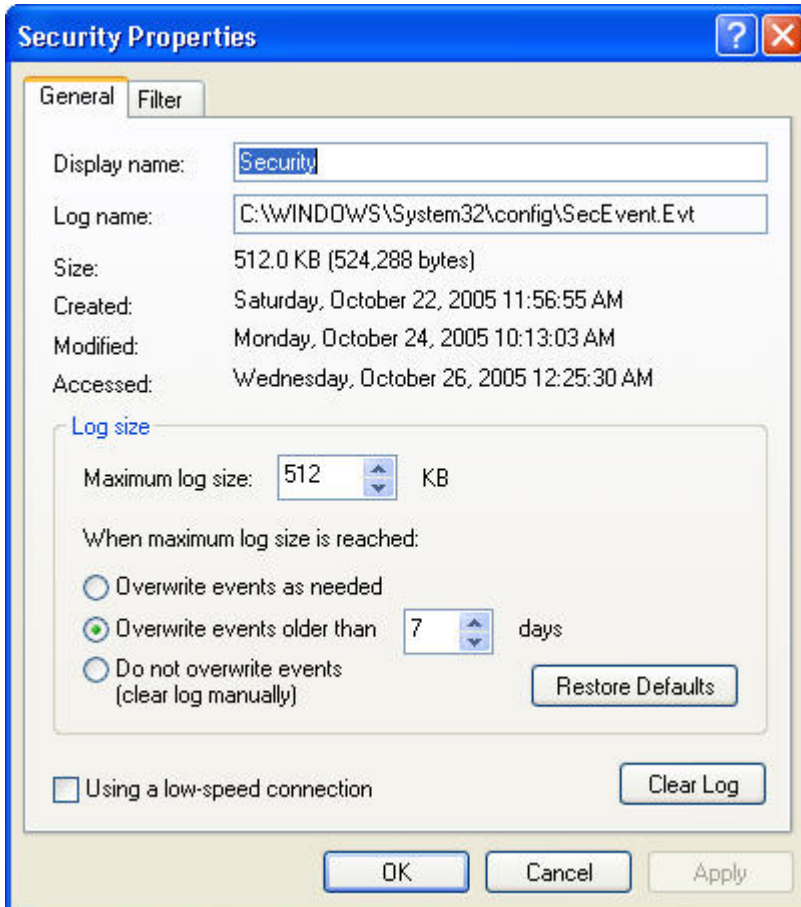


Figure 2-9 Security Log Properties

You can use Event Viewer to dump the Security log to a file, either in the process of clearing the log or independently. When you right-click Security and select **Save As**, you have the option to choose the format in which to save the log. If you specify event file (EVT) format, Event Viewer saves a copy of the log in the same EVT format that the live log uses. You can also specify comma-separated value (CSV) or tab-delimited (TXT) format. If you use the EVT format, you'll need to use a tool that supports EVT files, such as LogParser. You can also use Event Viewer to view logs saved in EVT format.

Note that when you save the Security log, Windows requires you to save it to a local volume of the server. You can subsequently copy the file elsewhere on the network, but the dump API that Event Viewer uses can save the log only to a

local volume. By default, Event Viewer stores the Security log as %systemroot%\system32\config\SecEvent.evt, but you can change the location by editing the registry. In a registry editor, maneuver to the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Eventlog\Security subkey and set the File value to any path name on the local system. Because the File value is data type REG_EXPAND_SZ, you can include environment variables in the path name. Why would you want to change the location of the log file? Perhaps you want to offload to a different volume the file I/O associated with the Security log.

Event Viewer occasionally will report that the Security log is corrupt and will refuse to display it. Usually, all you need to do to make the problem go away is close and re-open Event Viewer. (If you ever run into a truly corrupt event log, <http://support.microsoft.com/kb/172156/> explains how to fix the problem.) The only time I've encountered a corrupt Security log was when a rogue administrator tampered with the log. This brings us to the subject of Security-log integrity.

The Security log is fairly secure. To erase events or otherwise tamper with the Security log or audit policy, you need physical access to the target system, administrator authority to that system, or Write access to a GPO applied to the system. The only shrink-wrapped tool that you can use to selectively delete events from the Security log is WinZapper, written by Peter Nordahl and found at <http://www.ntsecurity.nu/toolbox/winzapper/>. (For more information about WinZapper, see my article "Avoiding WinZapper's Sting" at <http://www.windowsitpro.com/Article/ArticleID/15674/15674.html>.) Larger IT departments should implement separation of duty between operations and security-monitoring staff. To protect the integrity of Security logs from rogue administrators, you must export events from the Security log, as frequently as possible, to a separate server that is out reach (both physically and logically) of the local server's administrator and other operations staff. Security-monitoring staff then can monitor the security activity reported by the servers and review the activity of operations staff, as needed.

Chapter 3

Understanding Authentication and Logon

You might have noticed that Windows 2000 and later has two audit policies that mention logon events: *Audit logon events* and *Audit account logon events*. (NT has only *Audit logon events*.) By itself, *Audit logon events* has limited value because of the way that Windows handles logon sessions. When you log on to a computer by using a domain account, you might expect the event to be logged on the DC. In reality, the logon occurs on the workstation or server you are accessing, and therefore the event is logged in that computer's Security log—if *Audit logon events* is enabled on that system.

Not having DCs logging authentication activity made it impossible to get a complete audit trail of domain-user logon activity. You would have had to collect the Security logs from every workstation and server on your network! Therefore, Microsoft added the *Audit account logon events* policy to Windows 2000. Leave it up to Microsoft to add an important new feature and give it a confusing name. *Audit account logon events* would be better named *Audit authentication events*. In Windows, authentication and logon are related but ultimately separate activities that can, and often do, take place on separate systems. To effectively use these two audit policies, you need to have a complete understanding of how the authentication and the logon processes work in Windows. Another issue that complicates things is the difference between local and domain accounts and how they affect the audit process. In this chapter we'll cover both issues in detail before looking at *Audit logon events* and *Audit account logon events* in subsequent chapters.

Local and Domain Accounts

Windows supports two kinds of user accounts: domain (stored in Active Directory) and local (stored in the SAM). (See Figure 3-1.) Local accounts are stored in the SAM of member servers and workstations and are authenticated by the local system.

Domain accounts are stored in AD and authenticated by DCs. Although it's often best to avoid using local accounts because of their associated maintenance problems, attackers frequently target local accounts so it's important to understand how Windows records logon attempts against local accounts.

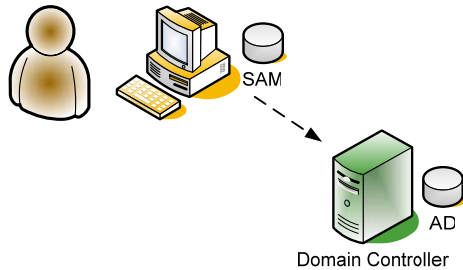
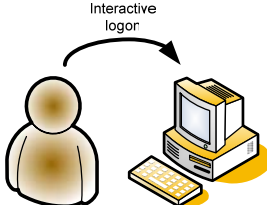
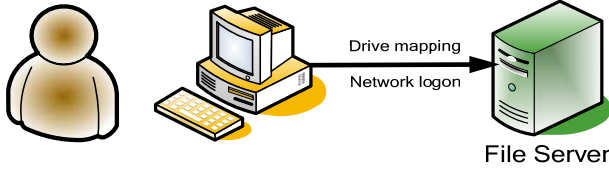
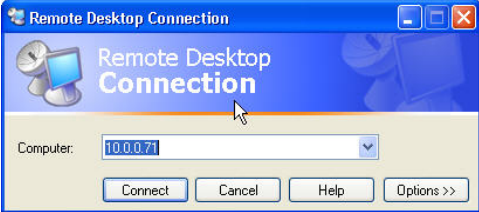
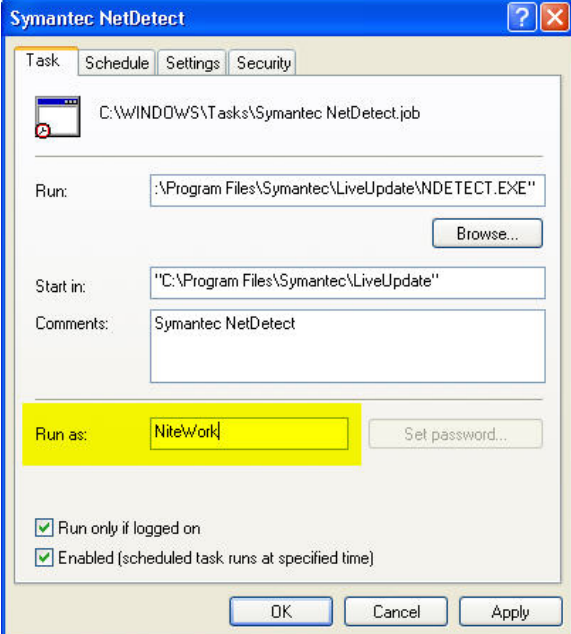


Figure 3-1 Windows supports both local and domain accounts

Logon Types

Windows supports five types of logon sessions. Logon types describe the ways in which users can log on to a system, such as through the system's local console or through a remote desktop session. You can use local or domain accounts with all five logon types. Each logon type has a corresponding logon right which the user must possess to initiate a logon of that type. (To view or modify rights assignments open Local Security Policy\Security Settings\Local Policies\User Rights Assignment). Figure 3-2 lists the five logon types and their corresponding logon rights. The type of user account and logon greatly affect which computer's Security log receives a logon event and which event IDs are logged.

Chapter 3 – Understanding Authentication and Logon

Logon Type	Example	Logon Right
Interactive: Used to log on at the local console		Logon locally
Network: Used to access a Windows resource (e.g., shared folder) from a system on the network		Access this computer from the network
Remote Interactive: Used to log on to a system via Remote Desktop, Terminal Services, or Remote Assistance		Allow logon through Terminal Services
Batch Job: Used to run a scheduled task as a specified account		Logon as a batch job

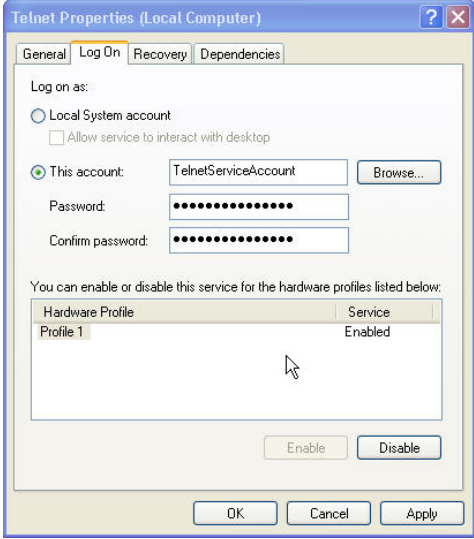
Logon Type	Example	Logon Right
Service: Used to run a service as a specified account		Logon as a service

Figure 3-2 Logon types

When a user logs on interactively at a workstation, Windows uses the *Log on to* field to determine whether the user is trying to log on with a local or domain account, as Figure 3-3 shows.



Figure 3-3 The Windows Logon dialog

When a user attempts a network logon, such as to a shared folder on another computer, the workstation by default reuses the credentials that the user entered when initially logging on. However, the user can specify a different local or domain account. For instance, when mapping a drive to a shared folder, a user can use the *Connect using a different user name* link, as Figure 3-4 shows. To specify a local account, prefix the user name with the computer name, followed by a backslash (\). To log on with a user account from a different domain, use the format of domain\user.

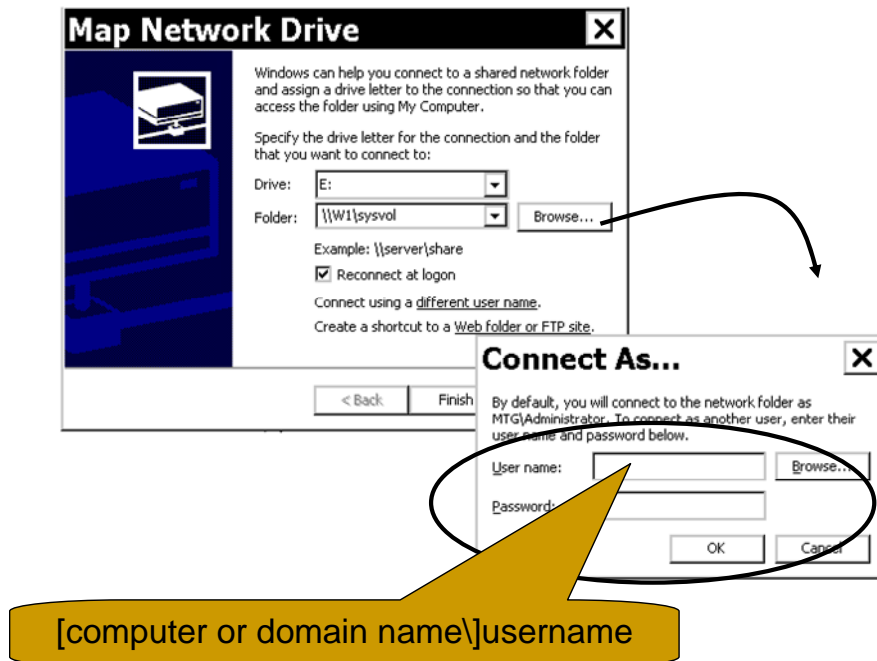


Figure 3-4 Connect using a different user name.

Logon vs. Authentication

As I stated earlier, logon and authentication are separate and distinct functions in Windows. Logon occurs on the system to which a user is gaining access, whereas authentication is performed by the computer on which the user's account resides. When you use a local account to log on to a computer, that computer performs both the logon and authentication. When you use a domain account to access a computer, the logon occurs on the accessed computer but a DC of the domain in which your user account resides performs the authentication.

Figure 3-5 illustrates a situation in which a user logs on interactively by using a domain account, then accesses a shared folder through a network logon. The Interactive logon at the workstation generates a Logon/Logoff event in the workstation's Security log and an Account Logon event on the DC. The Network logon generates a Logon/Logoff event in the file server's Security log and more Account Logon activity in the DC's Security log.

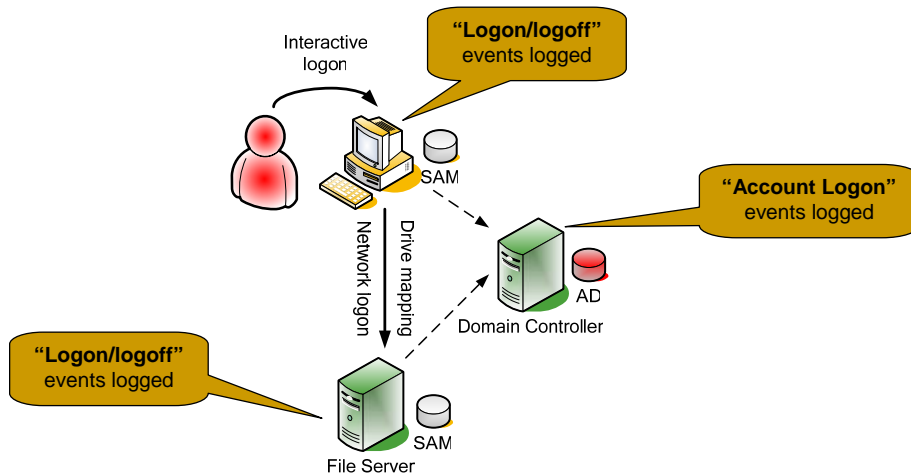


Figure 3-5 User logs on with a domain account

Figure 3-6 shows a user logging on to the same two computers by using local accounts. Each computer logs both categories of events; nothing is logged on the DC.

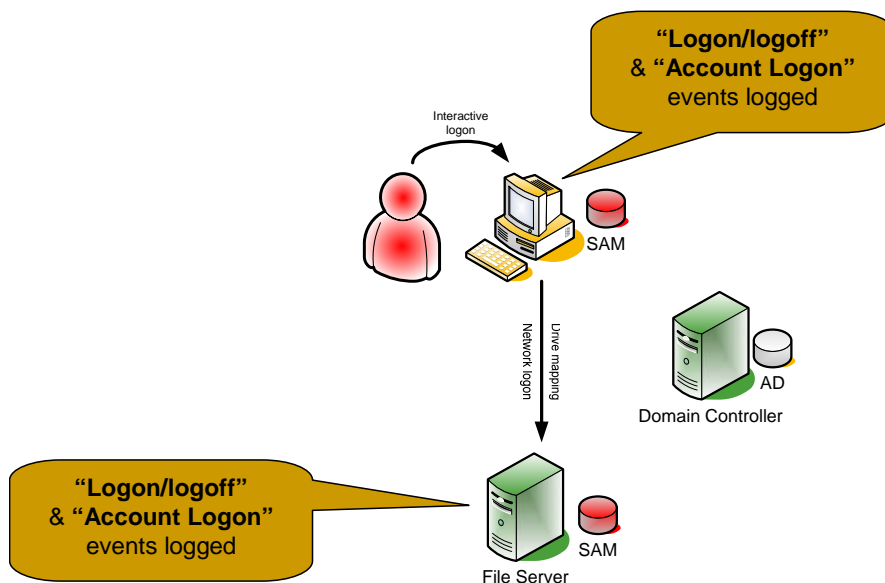


Figure 3-6 User logs on with local accounts

As you can see, *Audit account logon events* is primarily a DC audit policy, but it's also useful on member servers, for identifying attacks against local accounts. Logon attempts that use local SAM accounts should be fairly uncommon if you are following best practice and using domain accounts as much as possible.

You might wonder what the benefit of *Audit logon events* is compared to *Audit account logon events*. As I'll explain in a future chapter, you can use *Audit logon events* to link process tracking and object access events to logon sessions. *Audit logon events* also generates more granular information about the type of logon as well as the reason for failed logons. And *Audit logon events* attempts to

record not only the logon event but also the logoff event of each session. (Unfortunately, logoff auditing isn't very dependable because of the idiosyncrasies of Windows networking.)

Now that I've clarified the differences between logon and authentication, let's dive into the nitty-gritty details of Account Logon events and how to decode NT LAN Manager (NTLM) and Kerberos authentication events.

Chapter 4

Account Logon Events

When you enable the Account Logon policy on a Windows 2000 or Windows 2003 DC, the policy records in the DC's Security log all domain account authentication that occurs on that DC. When you analyze the combined Account Logon activity of all your DCs, you have a complete picture of the logon activity of all domain accounts in the domain, regardless of whether the logon attempts are initiated from computers of the local or trusted domain or even unknown computers outside your AD forest and external trusted domains. Windows 2000 and Windows 2003 DCs support NTLM and Kerberos authentication protocols.

NTLM vs. Kerberos

When a Windows 2000 or later computer needs to find out if a domain account is authentic, the computer first tries to contact the DC via Kerberos. (If the system doesn't receive a reply, it falls back to using NTLM. Windows 2000 and later DCs log different event IDs for Kerberos and for NTLM authentication activity, so you can easily distinguish between them.) In an AD forest comprising computers running Windows 2000 and later, all authentication between workstations and servers should be made via Kerberos. Any NTLM authentication events you see on DCs can have only a few explanations.

Windows will fall back to NTLM if routers block Kerberos traffic (UDP port 88). You might see NTLM events on your DCs if the domain trusts another domain outside the forest (as defined in Active Directory Domains and Trusts). This happens because Kerberos doesn't work for most external trust relationships. (Note that realm trusts between a Windows domain and a non-Windows Kerberos realm use Kerberos instead of NTLM. Also, Windows 2003 supports a new type of trust called *cross-forest trusts*. A cross-forest trust is a transitive, two-way trust between two Windows 2003 domains. Cross-forest trusts use Kerberos, not NTLM.)

NTLM events on a DC's Security log can be an indication of rogue computers. Contrary to popular belief, Windows does not prevent a user at a computer from an untrusted domain or stand-alone computer (i.e., a Windows computer that doesn't belong to any domain) from connecting to a server in your domain by using a domain account. To prove this, just use the Net use command to map a drive to a computer in an untrusting domain. For instance, the following sample command connects me to a file server called NYC-FS-1 in the NYC domain, using the domain Administrator account and a password of #dk32HE4:

```
net use \\nyc-fs-1.nyc.acme.local\c$ #dk32HE4 /user:nyc\administrator
```

If you have an application (such as a Microsoft IIS Web application) that uses IIS authentication, you might see NTLM events as a result of users logging on to the application. NTLM events also are generated by Linux, UNIX, or Mac computers. About the only other explanation for NTLM events on your DC Security logs is that you just have some pre-Windows 2000 computers somewhere in the local domain or in the overall forest.

Why does it matter whether your systems use NTLM or Kerberos? For one thing, Kerberos provides mutual authentication between client and server, whereas NTLM authenticates client to server only. For another thing, NTLM is less secure than Kerberos. Although intruders can capture packets from either protocol and attempt to crack the data back to the actual password, NTLM is easier to crack than Kerberos.¹ More important, if an outsider (i.e., someone who is not a member of your organization) is attacking accounts in your domain, you will most likely see those attacks as NTLM authentication events, not as Kerberos. Since the outsider's computer isn't a member of your domain or a trusted domain, his or her logon attempts will use NTLM.

How NTLM Works

NTLM is a challenge/response protocol. Figure 4-1 illustrates the following narrative example: When a user attempts to log on to the workstation, the computer contacts the DC to request authentication of the user. The DC generates a random string of bytes known as the *challenge* and sends it to the workstation. The workstation takes the password that the user has entered, hashes it, then encrypts the challenge, using the password as the key. The encrypted challenge is called the *response*, which the workstation sends back to the DC. The DC encrypts its copy of the original challenge and compares it to the response from the workstation. If the challenge and response match, the encryption keys are the same, which infers that the user correctly entered the password.

¹ You can strengthen NTLM against sniff-and-crack attacks by implementing NTLMv2. See my article "Access Denied: Implementing NTLMv2 on Win2K, NT, and Win9x machines" at <http://www.windowsitpro.com/Article/ArticleID/23068/23068.html> for more information.

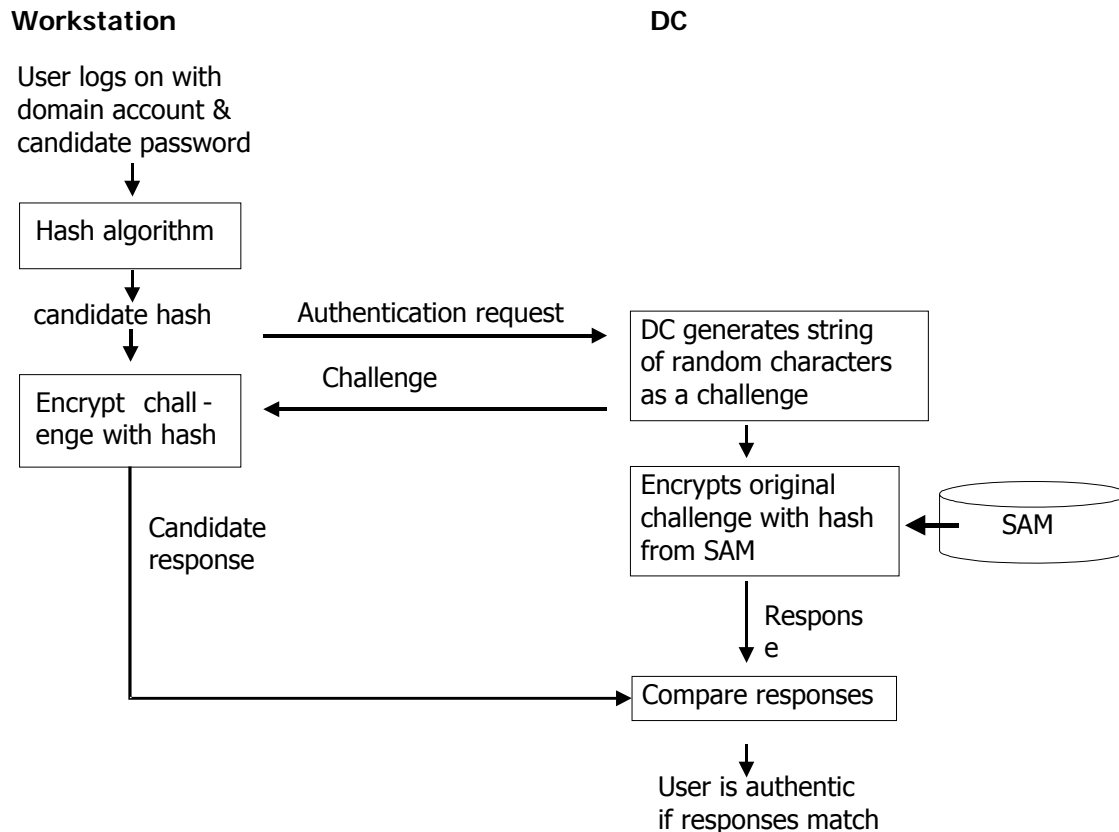


Figure 4-1 How NTLM works

As you can see, NTLM never sends the password or even the hash across the network. However, attackers who can capture the challenge/response packets might be able to crack the information back to the original password because of weaknesses associated with NTLM's backward compatibility support with ancient LanManager technology.²

NTLM Events

Windows 2000 logs just two event IDs—680 and 681—for all types of NTLM authentication activity. A successful NTLM authentication yields event ID 680; a failure produces event ID 681. (Windows 2003 merges these two events into event ID 680, as Figure 4-2 shows.)

² See my article "Why NT Passwords Are Weak" at <http://www.windowsitpro.com/Article/ArticleID/15893/15893.html>.

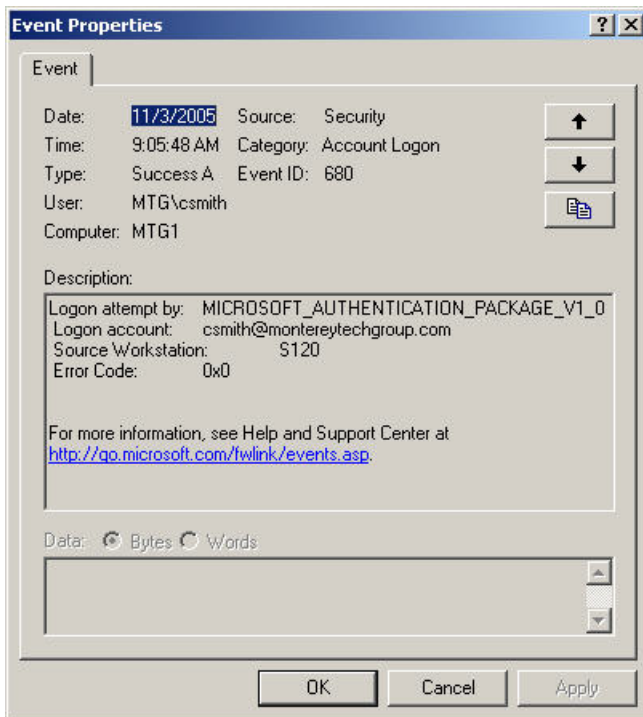


Figure 4-2 Event ID 680

An NTLM event description's *Logon account* field lists the name of the user account that attempted the authentication. The Workstation field provides the name reported by the computer on which the user is present. In the case of non-Windows systems (e.g., Apple computers), the Workstation field might contain a domain name instead of workstation name.

To determine why an authentication attempt failed, look at the description's Error Code field. Figure 4-3 lists the NTLM error codes. As you can see, NTLM provides very specific error codes. You might be surprised when you compare these codes to Kerberos failure, which are far less specific because they reflect the Request for Comments (RFC) 1510 Kerberos standard.

Error Code		Error Description	Comments
Decimal	Hex- adecimal		
3221225572	C0000064	User name does not exist	
3221225578	C000006A	User name is correct but the password is wrong	
3221226036	C0000234	User is currently locked out	
3221225586	C0000072	Account is currently disabled	
3221225583	C000006F	User tried to logon outside his day of week or time of day restrictions	Logged only for domain accounts that are configured with logon-hour restrictions
3221225584	C0000070	User tried to logon from an unauthorized workstation restriction	Logged only for domain accounts that are configured with workstation restrictions
3221225875	C0000193	Account expiration	Logged only for domain accounts that are configured with an account expiration
3221225585	C0000071	Expired password	
3221226020	C0000224	User is required to change password at next logon	Usually logged the first time a user logs on after the account is created or after the password is reset

Figure 4-3 NTLM error codes

NTLM yields an Account Logon event whenever a user logs on to a computer interactively or over the network. For instance, imagine that Bob uses a domain account to log on to his NT workstation, then uses a share folder on server A and one on server B. On whichever DC handles those authentication requests, you'll see a total of three event ID 680 instances: one for the interactive workstation logon and two for the network logons to server A and server B, as Figure 4-4 shows. However, you won't be able to discern that Bob accessed three systems; the events in this case will be identical except for the date and time stamp. An improvement to NTLM events would be the addition of the IP address of the sending computer. With that information, the three events would be much more valuable because you would be able to tell from the DC Security log not only that Bob had authenticated via NTLM three times but that he had accessed his workstation, server A, and server B.

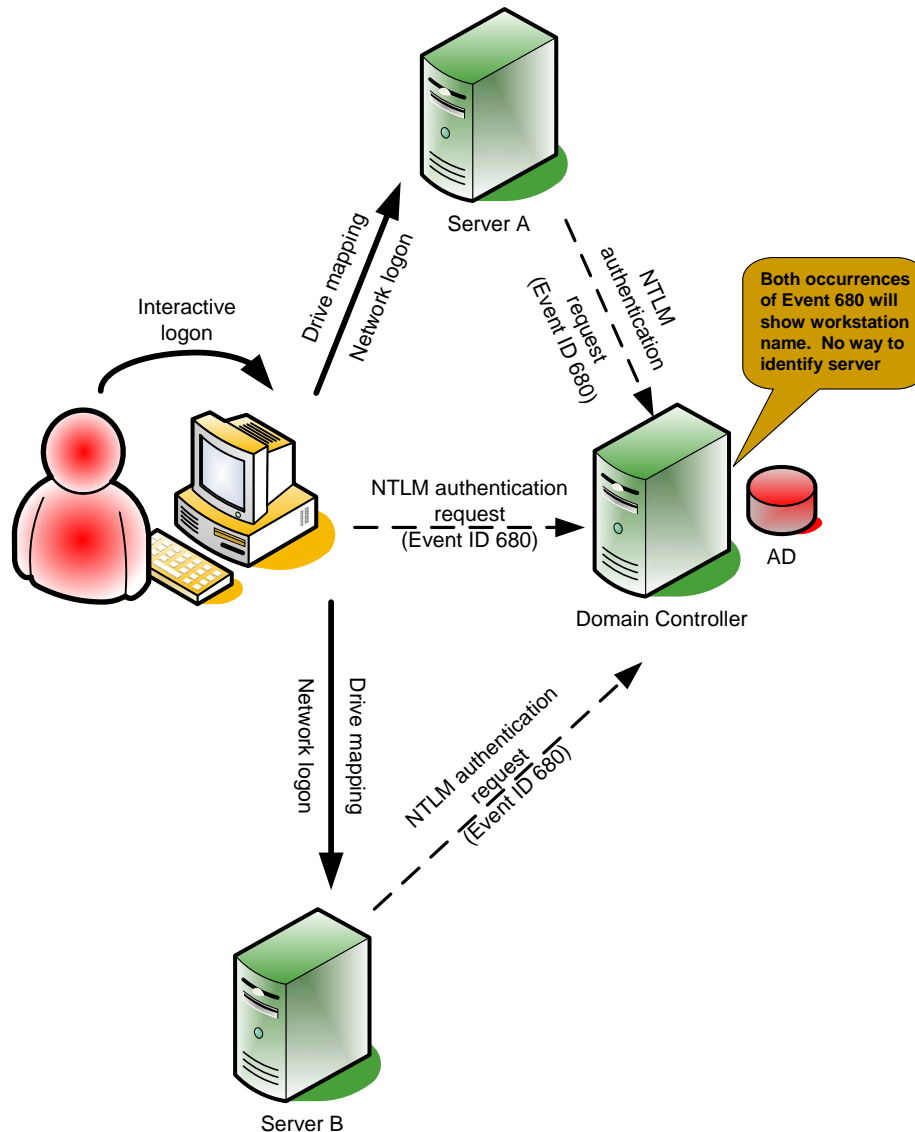


Figure 4-4 NTLM authentication

Things are a little different in Windows 2003. Annoyingly, Microsoft eliminated event ID 681 and instead uses event ID 680 for both successful and failed NTLM authentication attempts, as Figure 4-5 shows. Don't look for event ID 681 on Windows 2003, and be sure to take into account the Success or Failure status of event ID 680 occurrences.

Event ID	Type	Description
680	Success Failure	Account used for logon Logon attempt
681	Failure	The logon to account: %2 by: %1 from workstation: %3 failed

Figure 4-5 NTLM event ID changes in Windows 2003

On DCs, NTLM authentication events give you a record of all logon attempts that used domain accounts and that were serviced by the NTLM authentication protocol. You will also see NTLM events on member servers and workstations. When someone attempts to use a local SAM account to log on to a Windows computer, the authentication is always handled by NTLM. Therefore, successful event ID 680 instances on a workstation or member server is a clear indicator that someone, some service, or some scheduled task successfully logged on by using a local account. You can find out the type of logon from the related Logon/Logoff event, which we'll consider in the next chapter. When you see a failed instance of event ID 680 or event ID 681 on a workstation or member server, you know that someone, some service, or some scheduled task attempted but failed to logon by using a local account. If your environment concentrates on using domain accounts and avoiding the use of local accounts, enabling the Account Logon category on workstations and member servers is an easy way to identify the improper use of local accounts.

While tracking NTLM authentication is important, don't forget about Kerberos authentication, which will likely be the bulk of authentication activity in your DC Security logs.

Kerberos Basics

Kerberos is the default authentication protocol for Windows 2000 and later computers in an AD domain. To understand Kerberos events, you'll find it helpful to understand the basic functioning of the Kerberos protocol. Kerberos uses the concept of *tickets*. A ticket is a small amount of encrypted, session-specific data issued by the DC. When a client needs to access a server on the network, the client first obtains a ticket from the server's DC. The ticket and other data supplied by the client vouch for the client's identity and provide a way for the client to authenticate the server as well. This means that Kerberos provides mutual authentication of both client and server. Using timestamps and other techniques, Kerberos protects tickets from cracking or replay attacks by eavesdroppers on the network. First, let me explain how the overall ticket process works, then I'll walk you through a user's actions and how they relate to Kerberos events.

There are two kinds of tickets: authentication tickets (aka *ticket-granting tickets*) and service tickets. Kerberos issues an authentication ticket when a client first authenticates itself to the DC. The DC sends back the authentication ticket and a session key that's been encrypted with the client's personal key (in this case, the user's password). The client uses its personal key to decrypt the session key. Then the client uses its authentication ticket and session key to obtain a service ticket for each server that it needs to access. Windows generates Security-log events at each step of the Kerberos authentication process, so when you know how to relate general Kerberos events to user activity in the real world, you can closely monitor domain logon activity and pinpoint suspicious events.

Kerberos and the Windows Security Log

Imagine that Fred walks into his office one morning, sits down in front of his XP computer, turns it on, and enters his domain user name and password. Fred's workstation sends an authentication request, via Kerberos, to the DC. In UNIX-based Kerberos implementations, Kerberos simply issues an authentication ticket without checking the user's credentials. If the user's computer can successfully decrypt the session key that comes with the authentication ticket, then the user is authentic. If the client is an impostor, it won't be able to obtain the session key. Therefore, UNIX Kerberos implementations don't immediately detect failed logons that happen because of a bad password. However, Windows takes advantage of an optional feature of Kerberos, called *pre-authentication*. Using pre-authentication, the DC checks the user's credentials before issuing the authentication ticket. If Fred enters a correct username and password, Windows logs a successful event ID 672, *Authentication ticket granted*. Figure 4-6 shows an example.

When you see event ID 672 that lists *Fred* as the user name in the event's description, you can interpret the event as Fred's initial logon at his workstation. In fact, you can even identify his workstation by using the Client Address field in the event's description. All Kerberos events include this field, which identifies the client computer's IP address. The other useful field in event ID 672 is the Supplied Realm Name field, which identifies the user account's domain which in Figure 4-7 is Marketing. Other Kerberos events identify the domain as User Domain or prefix the user name with the domain (e.g., Marketing\Fred). But what if Fred enters a bad password? In this case, Kerberos pre-authentication catches the problem at the DC, and Windows logs event ID 675, *Pre-authentication failed*, with Failure Code 24 in the event's description (as Figure 4-7 shows).

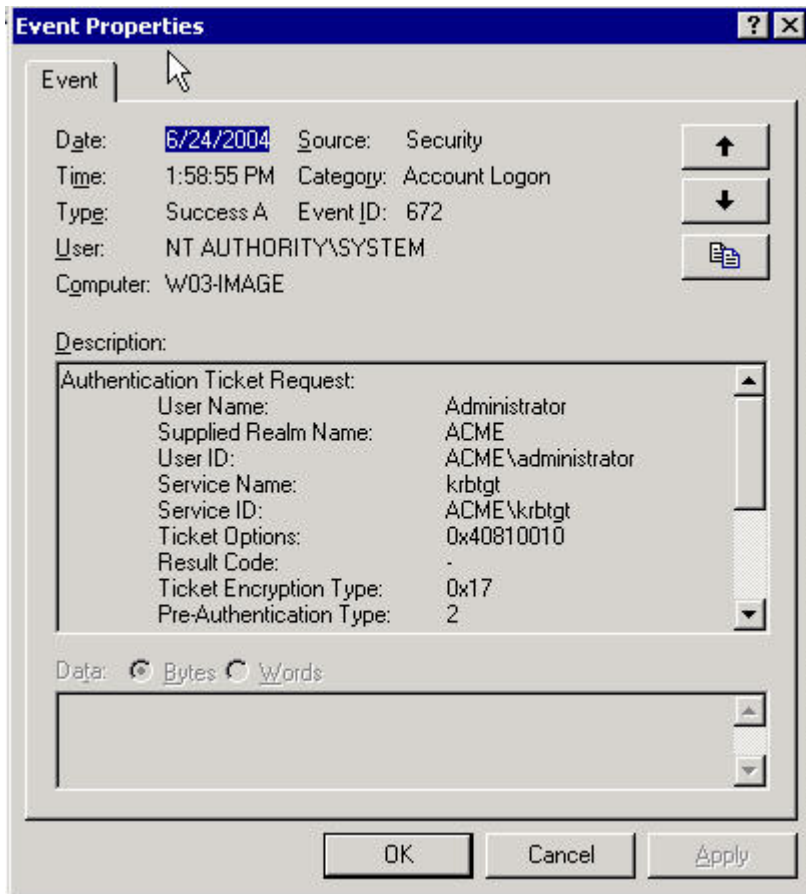


Figure 4-6 Event ID 672

Event Type: Failure Audit
Event Source: Security
Event Category: Account Logon
Event ID: 675
Date: 2/12/2004
Time: 3:22:32 AM
User: NT AUTHORITY\SYSTEM
Computer: DC1
Description:
Pre-authentication failed:
User Name: Fred
User ID: MKTG\Fred
Service Name: krbtgt/MKTG
Pre-Authentication Type: 0x2
Failure Code: 24
Client Address: 10.42.42.10

Figure 4-7 Event ID 675

After obtaining a ticket granting ticket (TGT), the workstation obtains a service ticket from the DC. Windows reports a successful attempt to do so as event ID 673, Service Ticket Granted. The Service Name description field displays *krbtgt*.

You'll always see this instance of event ID 673 after an instance of event ID 672 and you can ignore it.

The workstation next must obtain a service ticket for itself (i.e., a service ticket that authenticates Fred to his workstation and allows him to log on). This event shows up as another instance of event ID 673. The Service Name field in event ID 673 identifies the service for which the ticket was granted—in this case, the workstation's name. This instance of event ID 673 is useful because it gives you the name of the workstation. The earlier instance of event ID 672 provided only the workstation's IP address, not its name.

That isn't the end of event ID 673, though. You'll see additional instances of the event for each server that Fred accesses after logging on to his workstation. Imagine that Fred's logon script or persistent drive mappings initiate connections to the MktgFiles shared folder on the FS2 server. The DC will log event ID 673 when Fred's workstation obtains a service ticket to FS2, as Figure 4-8 shows.

Event Type: Success Audit
Event Source: Security
Event Category: Account Logon
Event ID: 673
Date: 2/12/2004
Time: 3:22:32 AM
User: NT AUTHORITY\SYSTEM
Computer: DC1
Description:
Service Ticket Granted:
User Name: fred
User Domain: MKTG.COM
Service Name: FS2\$
Service ID: MKTG\F\$2\$
Ticket Options: 0x40810010
Ticket Encryption Type: 0x17
Client Address: 10.42.42.10

Figure 4-8 Event ID 673

To recap, when Fred logs on at his workstation for the first time that day, the DC that handles the logon will log event ID 672, closely followed by three instances of event ID 673. One instance lists krbtgt as the service and can be ignored. Another instance lists the DC as the service, and the third instance identifies Fred's workstation name, as illustrated in Figure 4-9.

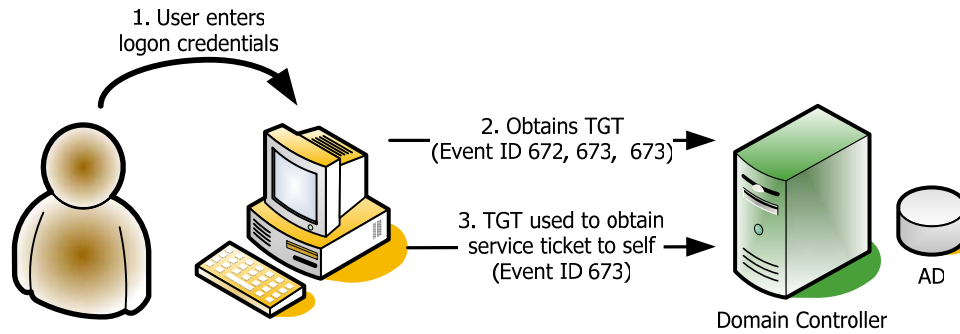


Figure 4-9 Initial logon via Kerberos

Additional event ID 673 instances are logged as Fred accesses other servers, as illustrated in Figure 4-10.³

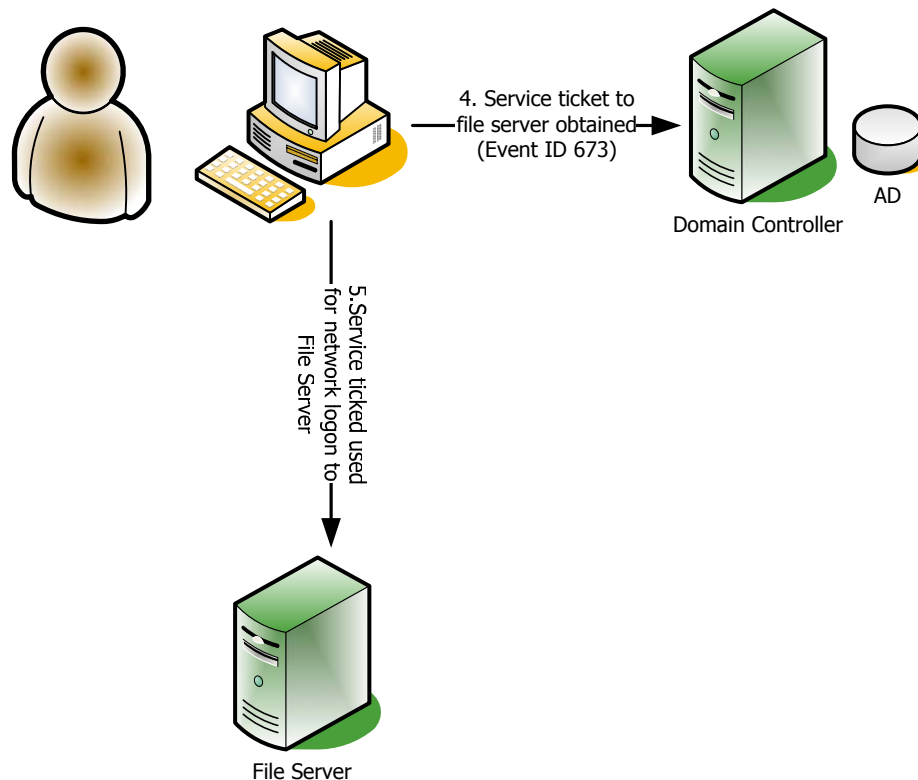


Figure 4-10 Subsequent logon to file server

I've shown you the event that Windows logs when a user enters a bad password (event ID 675 with failure code 24), but what about all the other reasons a logon can fail, such as an expired password or disabled account? Windows 2000 catches all of these logon failures after pre-authentication and logs event ID 676, Authentication Ticket Request Failed. Again, you need to look at the failure code to determine the problem. The most common Kerberos failure codes are listed in

³ Microsoft Knowledge Base article [217098](#), "Basic Overview of Kerberos User Authentication Protocol in Windows 2000" is a good reference for how Windows implements Kerberos.

Figure 4-13. Because Kerberos was developed as a non-operating system-specific protocol, its failure codes do not always directly map to Windows authentication failure reasons. For instance, Kerberos reports the same code for failures that occur because an account is disabled, expired, or locked out. Windows 2003 doesn't log event ID 676. Instead, Windows 2003 reuses event ID 672 but changes the type to Failure Audit instead of Success Audit.

Computer Account–Related Kerberos Events

To support Windows infrastructure features such as AD, Group Policy, and Dynamic DNS updates, workstations and servers must frequently communicate with the DC. This communication begins when a computer boots up. Each system that boots up obtains a TGT from the DC, followed by service tickets to `krbtgt` and the DC resulting in a 672,673,673 event ID pattern.

Windows logs a lot of what most people consider extraneous Kerberos events that you can simply ignore. The events are a result of computers interacting with the domain controller. You can identify such computer-to-computer authentication events because the user name listed in the event is a computer instead of a user. Computer-account names are easily recognizable because they have a \$ appended to the name (e.g., `FS2$` in Figure 4-8).

You will also see plenty of occurrences of event ID 677, resulting from ticket expirations. When a computer remains up for an extended period of time, its service ticket reaches its lifetime limit imposed by the domain's Kerberos policy. When that happens, the computer attempts to renew the ticket. When the renewal succeeds, the DC logs event ID 674. When the ticket has reached its maximum renewal lifetime, the renewal fails and the DC logs event ID 677. This forces the computer to reauthenticate to the DC and obtain a TGT all over again which causes a repeat of the event sequence logged when a computer first starts. Ticket expiration is a natural part of Kerberos activity and can be ignored.

Description Fields

Windows is not very consistent with the description fields you find in Kerberos events. Kerberos events always identify the name and domain attempting authentication, but the information is formatted several different ways, depending on the description fields. Events with a *User name* field generally use the account's pre-Windows 2000 logon name. *User domain* and Supplied Realm Name can log either the domain's pre-Windows 2000 name (e.g., `ACME`) or its DNS name (e.g., `acme.com`). User ID uses the pre-Windows 2000 format of `domain\username` (e.g., `ACME\JSMITH`).

The *Service name* field found in many Kerberos event descriptions is quite unpredictable as well. You might find service names in a variety of formats, including `computername$`, `krbtgt`, and `krbtgt/domain DNS name` or event

HOST/IP address. Events that use the Service ID field generally use the format domain/computername\$.

The Client Address field is valuable because you can use it to determine the source IP address for authentication requests. I've used this field for a variety of purposes, including tracing logon attacks back to their source on the Internet.

Other fields that don't appear to have much (if any) value are Ticket Options, Ticket Encryption Type, and Pre-Authentication Type. These fields always seem list the same undocumented values.

Final Thoughts on Kerberos

For a complete listing of Windows 2000 Kerberos events, see Figure 4-11. Figure 4-12 documents the changes to these events in Windows 2003. (Notice that event ID 676 has been merged into event ID 672 and event ID 677 has been merged into event ID 673.)

Event ID	Type	Description
675	Failure	Pre-authentication Failed
672	Success	Authentication Ticket Granted
676	Failure	Authentication Ticket Request Failed
673	Success	Service Ticket Granted
677	Failure	Service Ticket Request Failed
674	Success	Ticket Granted Renewed

Figure 4-11 Windows 2000 Kerberos events

Event ID	Type	Description
675	Failure	Pre-authentication failed
672	Success Failure	Authentication Ticket Granted Request
676	Failure	Authentication Ticket Request Failed
673	Success Failure	Service Ticket Granted Request
677	Failure	Service Ticket Request Failed
674	Success	Ticket Granted Renewed <u>Service Ticket Renewed</u>

Figure 4-12 Windows Server 2003 Kerberos events

As you can see, you can use Windows Kerberos events, as tracked in event ID 672 and event ID 673, to identify a user's initial logon at the workstation and then track each server the user subsequently accesses. You can use event ID 675 and event ID 676 (or event ID 676 and failed event ID 672 on a Windows 2003 DC) to track failed authentication events.

Keep in mind that authentication events logged on DCs (whether Kerberos or NTLM) don't include logoff events. DCs perform only authentication services. Each workstation and server keeps track of who remains logged on. To track logoff events, you must analyze the local Security log of the desired workstation or server, looking for Logon/Logoff events.

The Windows Server 2003 Security Log Revealed

Failure code		Kerberos RFC Description	Comments
Decimal	Hex- adecimal		
1	0x1	Client's entry in database has expired	
2	0x2	Server's entry in database has expired	
3	0x3	Requested protocol version # not supported	
4	0x4	Client's key encrypted in old master key	
5	0x5	Server's key encrypted in old master key	
6	0x6	Client not found in Kerberos database	Bad user name, or new computer/user account has not replicated to DC yet
7	0x7	Server not found in Kerberos database	New computer account has not replicated yet or computer is pre-Windows 2000
8	0x8	Multiple principal entries in database	
9	0x9	The client or server has a null key	Administrator should reset the password on the account
10	0xA	Ticket not eligible for postdating	
11	0xB	Requested start time is later than end time	
12	0xC	KDC policy rejects request	Workstation/logon time restriction
13	0xD	KDC cannot accommodate requested option	
14	0xE	KDC has no support for encryption type	
15	0xF	KDC has no support for checksum type	

Chapter 4 – Account Logon Events

Failure code		Kerberos RFC Description	Comments
Decimal	Hex- adecimal		
16	0x10	KDC has no support for padata type	
17	0x11	KDC has no support for transited type	
18	0x12	Clients credentials have been revoked	Account disabled, expired, or locked out
19	0x13	Credentials for server have been revoked	
20	0x14	TGT has been revoked	
21	0x15	Client not yet valid - try again later	
22	0x16	Server not yet valid - try again later	
23	0x17	Password has expired	The user's password has expired
24	0x18	Pre-authentication information was invalid	Usually means bad password
25	0x19	Additional pre-authentication required*	
31	0x1F	Integrity check on decrypted field failed	
32	0x20	Ticket expired	Frequently logged by computer accounts
33	0x21	Ticket not yet valid	
33	0x21	Ticket not yet valid	
34	0x22	Request is a replay	
35	0x23	The ticket isn't for us	
36	0x24	Ticket and authenticator don't match	
37	0x25	Clock skew too great	Workstation's clock too far out of sync with the DC's

Failure code		Kerberos RFC Description	Comments
Decimal	Hex- adecimal		
38	0x26	Incorrect net address	IP address change?
39	0x27	Protocol version mismatch	
40	0x28	Invalid msg type	
41	0x29	Message stream modified	
42	0x2A	Message out of order	
44	0x2C	Specified version of key is not available	
45	0x2D	Service key not available	
46	0x2E	Mutual authentication failed	Might be a memory allocation failure
47	0x2F	Incorrect message direction	
48	0x30	Alternative authentication method required*	
49	0x31	Incorrect sequence number in message	
50	0x32	Inappropriate type of checksum in message	
60	0x3C	Generic error (description in e-text)	
61	0x3D	Field is too long for this implementation	

Figure 4-13 Kerberos failure codes

Chapter 5

Logon/Logoff Events

Whether a user logs on by using a local SAM account or a domain account, the Logon/Logoff category records the attempt on the system to which the user tries to log on. When the user logs on to a workstation's console, the workstation records a Logon/Logoff event. When you access a Windows server on the network, the relevant Logon/Logoff events appear in the server's Security log. So, although account logon events associated with domain accounts are centralized on DCs, Logon/Logoff events are found on every system in the domain.

Logon/Logoff events aren't a good option for tracking domain account authentication or for detecting attempts to access computers by using local SAM accounts. However, they do provide some information not available otherwise. First and foremost, Logon/Logoff events on a given system give you a complete record of all attempts to access that computer, regardless of the type of account used. Second, these events reveal the type of logon, which you can't determine from Account Logon events. Ostensibly, this category should also provide the ability to track the logon session itself, identifying not just the logon event but also the logoff. Unfortunately, the value of logoff events is questionable at best. Logon/Logoff events also provide the IP address of the client computer, which is useful information for NTLM-based logons because NTLM Account Logon events doesn't provide the IP address. Finally, the Logon/Logoff category provides two event IDs specific to Terminal Services activity.

Logon/Logoff Event IDs

Windows logs all event IDs in the Logon/Logoff category as either success or failure, never as both. Windows logs one of two event IDs for successful logons. For successful network logons (logon types 3 and 8), Windows logs event ID 540. For all other logon types, Windows logs event ID 528. When Windows detects the end of a logon session, it records event ID 538. The events are shown in Figure 5-1.

Event ID	Type	Description
528	Success	Successful Logon
540	Success	Successful Network Logon
530	Success	User Logoff

Figure 5-1 Successful Logon/Logoff event IDs

Figure 5-2 demonstrates the event IDs recorded when a user initially logs on interactively to a workstation and accesses a shared folder on a file server. The events are the same whether the user logs on with a local account or a domain account.

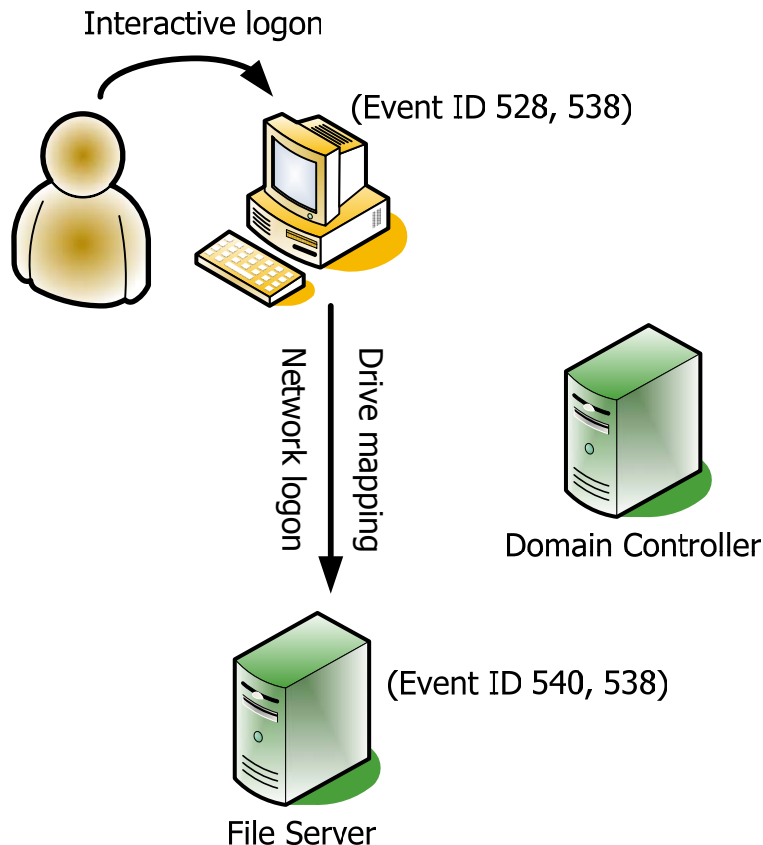


Figure 5-2 Successful logons

How do you link a successful logon event to its corresponding logoff event? You use the Logon ID description field. The logon ID is a unique number (at least I've never seen a duplicate logon ID occur between system restarts) that identifies a particular logon session. By looking for a subsequent instance of event ID 538 that has the same logon ID as an instance of event ID 528 or event ID 540, you can theoretically show when a user logged on and when he or she logged off. However, Windows doesn't log event ID 538 in the way you'd expect, especially for network logons.

Let's discuss interactive logons first. When Bob logs on to his workstation, Windows logs event ID 528 with logon type 2 and the logon ID for the logon session. When the Bob logs off, Windows logs event ID 538 with the same logon ID. But what if the system crashes or is unceremoniously powered down before Bob logs off? When Windows restarts, it might reconstruct the fact that Bob was logged on and record an instance of event ID 538. Taken literally, the event log won't make sense because you'll see a system restart followed by a logoff.

Logging that artificial instance of event ID 538 is a bit of a formality, if you ask me. At any rate, tracking user logoffs in a workstation's Security log is pretty easy.

Tracking network-session logoffs in a server's Security log is another story. When a user maps to a shared folder, the server logs event ID 540 with the logon ID of the logon session. However, if a user remains connected to that share but has no files open and no other activity occurs on the network connection, the server closes the logon session to conserve resources. This server-initiated logoff is invisible to the user because when he or she finally accesses the

The *User name* and Domain description fields in Logon/Logoff events consistently list the pre-Windows 2000 form of the account and domain names. DC Security logs contain many Logon/Logoff events generated by computer accounts, as opposed to user accounts. As with computer-generated Account Logon events, you can recognize these events because the *User name* description field will list a computer name followed by a dollar sign (\$). You can ignore computer-generated Logon/Logoff events.

The Logon Process description field identifies the Windows process that actually submitted the logon request. A logon process collects identification and authentication information and then uses Local Security Authority services to log users on. Because different logon processes handle specific logon types and scenarios, the logon process name can help fill in some gaps in the information provided by Logon/Logoff events. For instance, you might see event ID 540 with logon type 3 with Bob's user name. But does that mean that Bob connected to a windows resource such as a shared folder or that he accessed a Web page on the server? If the logon process is *advapi*, you can determine that the logon was a Web-based logon: IIS processes logon requests through a process called *advapi*. If the logon was to a Windows resource and authenticated via Kerberos, the Logon Process field would list *Kerberos*. Generally, the Logon Process field provides a hint at how the user tried to access the system—at its console, through Server Message Block (SMB) or Common Internet File System (CIFS) for shared-folder access, or via IIS. But as Figure 5-3 shows, some logon processes are authentication-protocol specific.

Logon Process	Explanation
Winlogon, MSGina	Used by interactive logons
KSecDD	Used by the SMB server to authenticate users via Kerberos
DCOMSCM	Used by Microsoft SQL Server and Microsoft SQL Server Desktop Engine (MSDE)
CHAP	Used by Routing and Remote Access Service (RRAS) dial-in or PPTP VPN access. CHAP stands for Challenge Response Authentication Protocol.
LAN Manager Workstation Service	Used in connection with network logons to other computers
Secondary Logon Service	Used by RunAs
Schannel	Used by Transport Layer Security/Secure Sockets Layer (TLS/SSL) authentication usually has to do with IIS Web content that requires SSL and authentication
RASMAN	Used by RRAS
Wdigest	Used by digest authentication, usually through IIS
IAS	Used by Internet Authentication Service (Windows' built-in RADIUS server)

Figure 5-3 Common logon processes

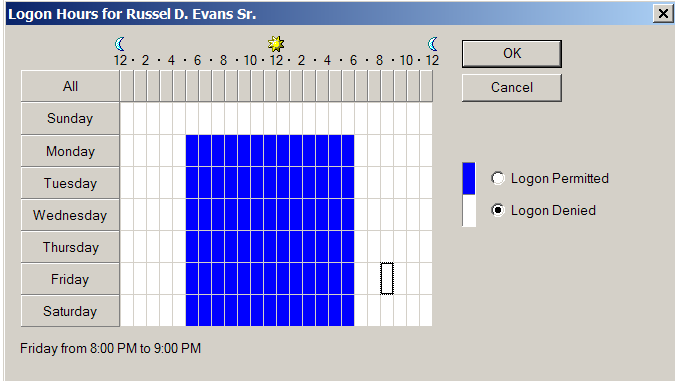
The Authentication Package description field is closely related to Logon Process and identifies the DLL used to handle the actual authentication of the logon attempt. Together, Logon Process and Authentication Package provide valuable information for figuring out where and how a user attempted to log on to the system. This capability is important because there are so many ways to access a Windows computer. Figure 5-4 lists common authentication packages.

Authentication Package	Explanation
MICROSOFT_AUTHENTICATION_PACKAGE_V1_0 or NTLM	NT LanManager
Wdigest	Digest authentication, usually through IIS
Microsoft Unified Security Protocol Provider or Schannel.dll	TLS/SSL authentication usually has to do with IIS Web content that requires SSL and authentication
Kerberos	Kerberos
Negotiate	Kerberos or NTLM, depending on client capability

Figure 5-4 Common authentication packages

Logon/Logoff Failure Events

When a logon attempt fails, Windows logs a distinct event ID that corresponds to the reason for the failure.

Event ID	Type	Description
529	Failure	Logon Failure: Unknown user name or bad password
530	Failure	<p>Logon Failure: Account logon time restriction violation</p>  <p>Applies only to domain accounts</p>
531	Failure	Logon Failure: Account currently disabled
532	Failure	<p>Logon Failure: The specified user account has expired</p> <p>Applies only to domain accounts</p>

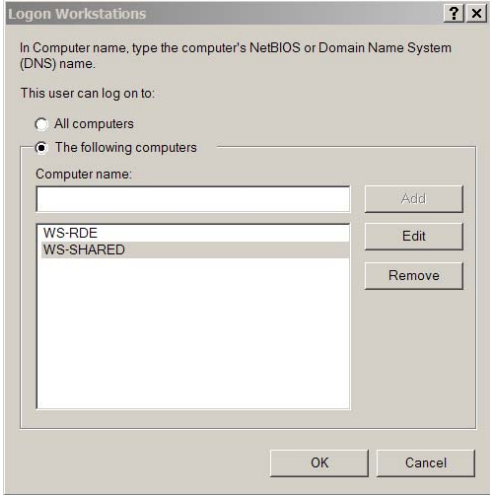
533	Failure	<p>Logon Failure: User not allowed to log on at this computer</p>  <p>Applies only to domain accounts</p>
534	Failure	Logon Failure: The user has not been granted the requested logon type at this machine (see Figure 3-2)
535	Failure	Logon Failure: The specified account's password has expired
536	Failure	Logon Failure: The NetLogon component is not active
537	Failure	<p>Logon failure: The logon attempt failed for other reasons</p> <p>A variety of situations can cause this unusual event; new articles about this event ID are frequently added to the Microsoft Knowledge Base</p>
539	Failure	Logon Failure:Account locked out

Figure 5-5

Terminal Services Events

The Logon/Logoff category includes two event IDs associated with Terminal Services (aka Remote Desktop and Remote Assistance) events that occur between logon and logoff. A user can disconnect from an RDP session without actually logging off. The session remains active in the background of the server. Later, the user can reconnect to the session from the original client or from a different client, and pick up where he or she left off. Windows logs these Logon/Logoff events as listed in Figure 5-6.

Event ID	Type	Description
682	Success	Session reconnected to winstation
683	Success	Session disconnected from winstation

Figure 5-6 Terminal Services events

New Description Fields in Windows 2003

Windows 2003 leaves Logon/Logoff event IDs unchanged but does add some new fields to these events' descriptions. Particularly useful is the new Source Network Address field, which identifies the IP address of the client workstation when different than the local server. Windows 2003 also provides the Source Port field, but this field typically isn't useful because client source ports are usually randomly chosen. Windows 2003 also adds four new caller description fields—Caller User Name, Caller Domain, Caller Logon ID, Caller Process ID—which correspond to the requesting logon process.

Bottom Line

Logon/Logoff events are useful in auditing all attempts to gain entry to a system, no matter the type of account or type of logon used. But you must monitor the Security log of each computer on which you need to perform such auditing. In contrast, you can use Account Logon events to track domain-account authentication fairly centrally, monitoring only DC Security logs. However, monitoring Account Logon events on DCs doesn't alert you to attempts to enter computers by using local SAM accounts. To catch those attempts, you must monitor the local Security log of workstations and member servers for either Account Logon NTLM events (which report only attempts to logon with a local account and don't report the source IP address) or Logon/Logoff events (which report both local-account and domain-account logon attempts and source IP address).

Chapter 6

Detailed Tracking

With the Detailed Tracking category, Windows gives you the ability to track programs executed on the system and to link those process events to logon sessions reported by Logon/Logoff events (which I discuss in Chapter 5) and to file access events generated by the Object Access category (which I discuss in Chapter 7). For instance, you can use Detailed Tracking events to determine that Joe opened Excel. By linking Detailed Tracking events to Logon/Logoff events, you can further show that Joe opened Excel during a remote desktop logon; by linking Detailed Tracking events to Object Access events, you can document that Joe used Excel to open and modify c:\files\payroll.xls. Detailed Tracking also provides event IDs for monitoring the installation and removal of services and the maintenance of scheduled tasks.

Process Tracking

When a system process or a user opens an executable, Windows creates a process in which that executable runs and logs event ID 592, which identifies the executable and the account that started the process. When the process terminates, Windows logs event ID 593, as Figure 6-1 and Figure 6-2 show.

Event ID	Type	Description
592	Success	A new process has been created
593	Success	A process has exited

Figure 6-1 Process event IDs

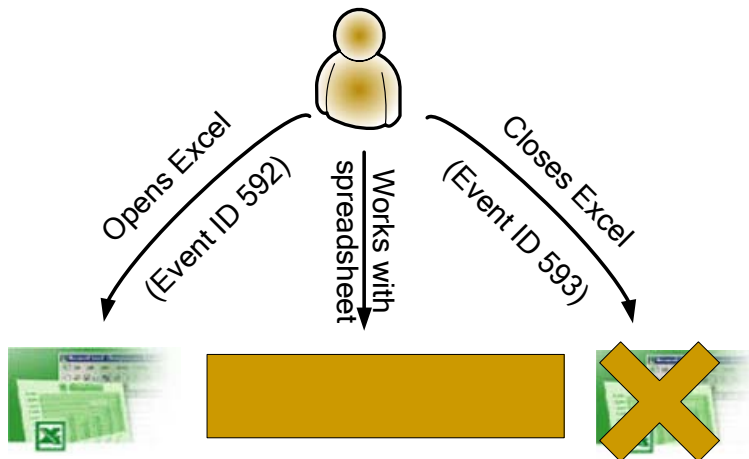


Figure 6-2 Process events

These events' Image File Name description field identifies the full path name of the executable that was started. For instance, if a user opens Notepad, the event will show something similar to `\WINDOWS\system32\notepad.exe` as the image filename.

To uniquely identify each process during a system boot session, Windows uses a Process ID, which you might have noticed when viewing the Task Manager Processes tab. Event ID 592 lists the process ID of a new process in the New Process ID field and lists the ID of the process that created the new process in the Creator Process ID field, as Figure 6-3 shows. The creator process ID gives you more information about why and how the program was started. For instance, programs that users open from the Start Menu will list the creator process as the current ID of Explorer—the Windows desktop. Processes that start as system services will list the ID of the service control manager as the creator process.

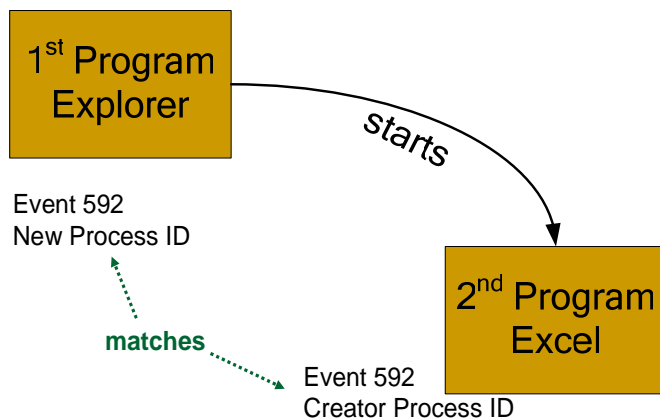


Figure 6-3 New process ID and creator process ID

Event ID 593 also lists the Process ID field in the description, so you can link an instance of that event to the corresponding instance of event ID 592 and thus determine not just when a program started but when it stopped as well. You'll also find process ID in many other security events, giving you the ability to determine the program that generated the event.

To determine the logon session during which a process started, look at the Logon ID description field in event ID 592, then find the preceding instance of event ID 528 or event ID 540 that has the same logon ID. Check the Logon Type and Logon Process fields to determine whether the process was started during an interactive or remote desktop session or some other type of logon session, as illustrated in Figure 6-4.

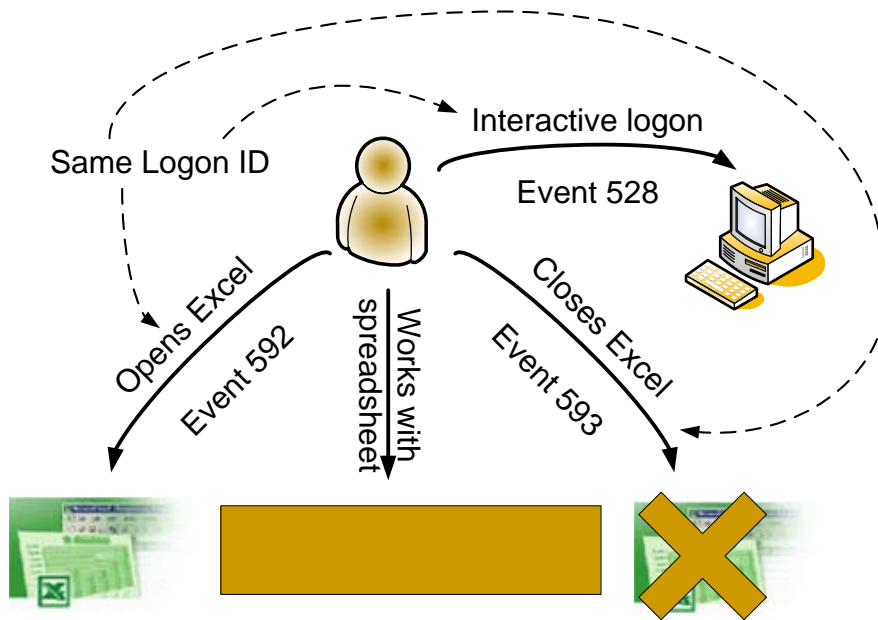


Figure 6-4 Linking detailed tracking and Logon/Logoff events

Service and Scheduled Task Events

Windows 2003 introduces two new Detailed Tracking events, as Figure 6-5 shows.

Event ID	Type	Description
601	Success Failure	Attempt to install service
602	Success	Scheduled Task created

Figure 6-5 Service and scheduled task events

Windows 2003 logs event ID 601 when a new service is installed. Although this event might be useful in finding new services that are being installed, it doesn't inform you about other software installations (e.g., user applications). Nor does this event help you to detect Trojan horses or backdoor programs because malware seldom installs as a service. The Service Name description field provides you with the short system name of the service. (You can use the "sc query" command to get a cross-reference of service names and their more-familiar display names.) The Service File Name field provides the full path name of the executable that hosts the service. Keep in mind that one executable can host multiple services.

Figure 6-6 and Figure 6-7 list the values in event ID 601's Service Type and Service Start Type description fields. Refer to Microsoft Developer Network (MSDN) documentation for more information about service types.

Service Type	Name
0x2	SERVICE_FILE_SYSTEM_DRIVER
0x4	SERVICE_ADAPTER
0x8	SERVICE_RECOGNIZER_DRIVER
0xB	SERVICE_DRIVER
0x10	SERVICE_WIN32_OWN_PROCESS
0x20	SERVICE_WIN32_SHARE_PROCESS
0x30	SERVICE_WIN32
0x100	SERVICE_INTERACTIVE_PROCESS
0x110	SERVICE_INTERACTIVE_OWN_PROCESS
0x120	SERVICE_INTERACTIVE_SHARE_PROCESS

Figure 6-6 Service types

Service Start Type	Name	Display type in Services MMC snap-in
0x1	SYSTEM_START	n/a
0x2	AUTO_START	Automatic
0x3	DEMAND_START	Manual
0x4	DISABLED	Disabled

Figure 6-7 Service Start types

Event ID 601's Service Account field specifies the account under which the service is configured to run. Other description fields identify the account, domain, and logon ID of the user who installed the service.

The other new Detailed Tracking event is event ID 602, which provides information about changes to scheduled tasks. Windows 2003 logs event ID 602 when a scheduled task is created or an existing task is changed, but the description always includes the field Scheduled Task created. Event ID 602's

description fields identify the name of the task (in the File Name field) and the command that the task executed (in the Command field). The Triggers and Time fields document the schedule of the service, and the Target User field specifies the domain and user under which the task will run. Other description fields identify the account, domain, and logon ID of the user who created or modified the task.

The Detailed Tracking category logs three other events that possess little if any value for typical monitoring scenarios. Figure 6-8 lists these events.

Event ID	Type	Description
594	Success	A handle to an object has been duplicated
595	Success	Indirect access to an object has been obtained
600	Success	A process was assigned a primary token

Figure 6-8 Miscellaneous Detailed Tracking events

As you can see, the primary importance of the Detailed Tracking category is its logging of event ID 592 and event ID 593, which provide a complete audit trail of all Windows programs executed on the monitored system. Detailed Tracking also provides a way to monitor changes to scheduled tasks and installation of services. Detailed Tracking does not report specific events for programs that are not Windows executables. For instance, if you execute a VBScript, you've simply executed `cscript.exe` or `wscript.exe` as far as Windows is concerned, and that's all event ID 592 will tell you.

To track access to objects such as files and folders, you need to use Object Access events. You can also tie events from the Logon/Logoff, Detailed Tracking, and Object Access categories together for more-sophisticated monitoring.

Chapter 7

Object Access Events

You can use the Object Access Security log category to audit any and all attempts to access files and other Windows objects. The only auditable objects not covered by this category are AD objects, which you can track by using the Directory Service category. In addition to tracking files, you can track success and failure access attempts on folders, services, registry keys, and printer objects. The way in which you define Object Access audit policy and the format of information recorded in the Security log for this category are closely related to the structure of the ACLs that all objects use to define who can access the object and how.

When you enable the *Audit object access events* policy for a given computer, Windows doesn't immediately begin auditing all access events for all objects because the system would immediately grind to a halt. Activating object access auditing is a two-step procedure. First, enable the *Audit object access events* policy on the system that contains the objects you want to monitor. Second, select specific objects and define the types of access you want to monitor. you make these selections in the object's audit settings, which you'll find on the object's Advanced Security Settings dialog box. For instance, Figure 7-1 displays the audit settings for a folder named Accounting Data.

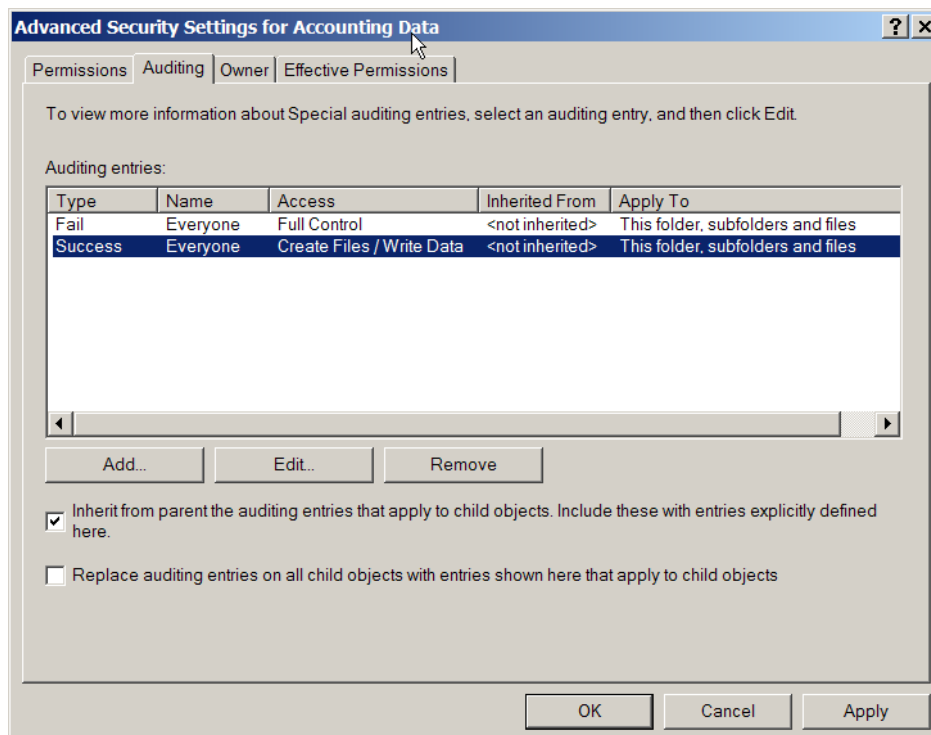


Figure 7-1 Object audit policy

As you can see in this figure, you can limit auditing according to the user or group accessing the object (the Name column), the permissions requested (the Access column), and whether the access attempt was successful or failed (the Type column).

Windows evaluates an object's audit policy much as it evaluates the object's permissions. When a user attempts to access an object (e.g., when Fred tries to open budget.xls), Windows must determine whether to report the attempt to the Security log. Windows analyzes all the audit entries that apply to the user who is attempting to access the object. If the object's audit policy contains entries for several groups, and the user attempting to access the object belongs to two or more of those groups, Windows will log the event if any of the applicable entries match (i.e., if the user requested one or more of the permissions flagged for auditing and the outcome of the attempt matches the success or failure criteria of the audit entry).

Suppose that Bob has Modify access to the Accounting Data folder and its files, which include budget.xls. Write Data is among the permissions that comprise Modify access. Bob uses Excel to open and write to budget.xls. Windows matches Bob's action to the second entry in Figure 7-1. Bob is a member of the Everyone group, his permissions to the accessed file include Write Data, and his attempt to write to the file was successful. Windows would not match an attempt by Bob to open the file simply to view it: Such an attempt would successfully use Read Data permissions, which aren't specified for auditing. Now let's suppose that Fred, who has no access to Accounting Data, is snooping around the network and attempts to list the files in Accounting Data. This failed access attempt will match the first entry in the folder's audit policy and trigger an Object Access event in the Security log.

Figure 7-2 provides a complete list of permissions, the corresponding names used by Object Access events in the Security log, and an explanation the permission as applied to folders and files.

Although you can limit auditing for a given object to specific groups or even individual users, I recommend sticking with the Everyone group. Singling out specific groups or users for monitoring puts you at risk of creating an incomplete audit trail and might expose you to claims of unfairness or raise questions as to the integrity of your information.

Be careful when specifying the type of access to monitor and when choosing whether to audit for Success or Fail types. You can easily create too inclusive an audit policy and deluge the Security log with useless noise. In particular, think twice about enabling success auditing of Read Data, Read Attributes, Ready Extended Attributes, Read Permissions, and Execute because these permissions are used so frequently by legitimate users during the course of work.

Chapter 7 – Object Access Events

Permission	Name in Object Access Events	Explanation	
		Folder	File
Traverse Folder / Execute File	Execute/Traverse	Folder traversed while browsing file system Permits movement through folders to reach other files or folders, if the user has no other permissions to the folders being traversed;has no effect unless the user lacks the <i>Bypass traverse checking</i> user right, which is assigned to Everyone by default	Script or .exe file executed
List Folder / Read Data	ReadData (or ListDirectory)	Names of subfolders and files queried by Explorer, dir command, or application	Actual content of file read
Read Attributes	ReadAttributes	Attributes (Read Only, Archive, System, Hidden) queried by Explorer, attrib command, or application	
Read Extended Attributes	ReadEA	Extended attributes (as defined by applications) queried by Explorer, attrib command, or application	
Create Files / Write Data	WriteData (or AddFile)	New file created in the folder	Content of file changed
Create Folders / Append Data	AppendData (or AddSubdirectory or CreatePipeInstance)	New subfolder created in the folder	Content appended to end of file
Write Attributes	WriteAttributes	Attributes (Read Only, Archive, System, Hidden) modified by Explorer, attrib command, or application	
Write Extended	WriteEA	Extended attributes (as defined by applications) modified by Explorer, attrib	

		Explanation
Attributes		command, or application
Delete Subfolders and Files	DELETE	Object deleted; allows or denies deleting subfolders and files even when the user lacks Delete permission on the specific subfolder or file
Delete	DELETE	Object deleted; overridden by <i>Delete Subfolders and Files</i> permission on the parent folder
Read Permissions	READ_CONTROL	Object's ACL queried
Change Permissions	WRITE_DAC	Object's ACL modified
Take Ownership	SeTakeOwnershipPrivilege	Owner of object changed

Figure 7-2 Auditable permissions on files and folders

In addition to the Type, Name, and Access columns, the Advanced Security Settings contains an Apply To column. For container objects such as folders, you can specify values for this column to control whether and how Windows propagates the audit entry to child objects. The Apply To value defaults to *This folder, subfolders and files* but can be changed to any combination of the folder, subfolders, and files. You can use the Apply To setting to fine tune your audit policy so that it ignores file or folder access events that are irrelevant to your audit needs, thus eliminating some noise from the Security log. For instance, you might need a record of who is accessing sensitive files in a certain folder, but you are not interested in folder-level access such as folder listings or creation of subfolders and files. In that case, you can enable auditing for the appropriate permissions but change the Apply To value to Files only.

If you'd like to restrict the behavior of Apply To to the immediate level of child objects and prevent an audit entry from propagating down to grandchild objects, you can select the *Apply these permissions to objects and/or containers within this container only* checkbox, as Figure 7-3 shows.

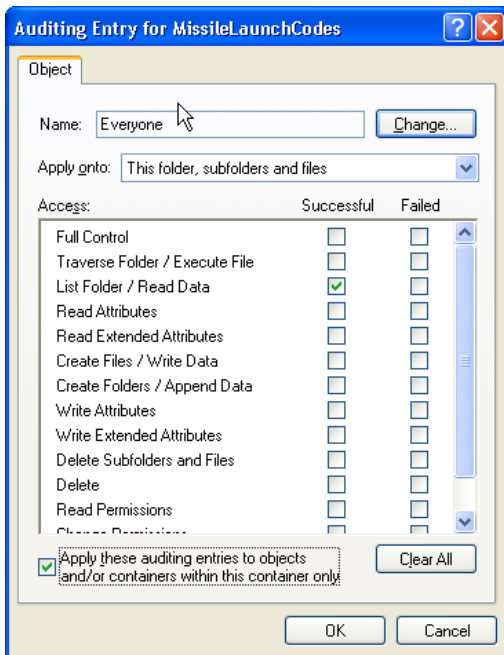


Figure 7-3 Limit propagation of permissions

To properly use the Apply To setting, be sure you understand the dual meaning of certain permissions. Note that some of the permissions listed in Figure 7-2 have a different meaning for files than for folders. For instance, Create Folder/Append Data for a folder means that the user can create new subfolders within the folder; for a file, the permission means that the user can append data to the end of the file. Likewise, List Folder/Read Data for a folder lets users only list the names of files and subfolders within the folder, but for a file, the permission lets users read the actual data contents of the file. What if you want to audit one of these dual meaning permissions for the folder only, not for the files within the folder? Or what if you need to audit access to the files within the folder but not access attempts to the folder? In such cases, use the Apply To setting.

When viewing an object's audit policy, you can determine where each audit entry is defined by consulting the read-only Inherited From column. In the example that Figure 7-1 shows, both entries are explicitly defined on the immediate folder, so this column reads <not inherited>. If at a certain level in your folder hierarchy you need to break the flow of inherited permissions and block them from propagating down to a particular subfolder, you can clear the *Inherit from parent the permission entries that apply to child objects* checkbox.

Object auditing is basically the selective logging of access control decisions that Windows makes. Have clear in your mind what you are trying to accomplish before enabling or configuring an object's audit policy. Figure 7-4 provides a number of common audit goals and the corresponding audit settings.

Goals	Audit Settings	
	Type	Access
Audit trail of changes to a file	Success	Write Data Append Data
Know when someone tries to access an object that they shouldn't	Failure	All permissions
Audit trail of access control changes to a folder ⁴	Success	Change Permissions Take Ownership
Audit trail of deletion of a folder or of any file in the folder	Success	Delete Subfolders and Files Delete

Figure 7-4 Common object audit policies

Object Access Events

Logon/Logoff and Detailed Tracking provide both an initialization and termination event ID corresponding to the beginning and end of a logon session or process. Likewise, Object Auditing generates event ID 560 when an object is opened and event ID 562 when the object is closed (as Figure 7-5 shows), with other events occurring between depending on what the application does with the open object. These events' Handle ID description field serves the same purpose as the Process ID and Logon ID fields in Detailed Tracking and Logon/Logoff events. To determine how long a file was open, simply look for an instance of event ID 562 that has the same Handle ID as the preceding event ID 560.

Event ID	Type	Description
560	Success Failure	Object Open
562	Success	Handle Closed

Figure 7-5 Object Access open and close events

It's important to understand that Object Access events reflect the interaction between Windows and an application—not between a user and the application. For instance, when Bob uses Microsoft Word to open memo.doc, edits a paragraph, and then closes the file, you might expect to find one instance of

⁴ Carefully consider the best setting for the Apply To column. If you choose *This folder, subfolders and files*, you'll achieve a complete audit trail, but permission changes to a parent folder that has many child objects will generate many Object Access events as Windows propagates the permission change. If you choose *This folder and subfolders*, you'll receive a more reasonable amount of events but you'll miss any permission changes to individual files, should such occur.

event ID 560 followed by event ID 562. Actually, unbeknownst to Bob, Word opens and closes the file multiple times in connection with his actions, and you'll find events reflecting all this activity. The fact that Object Access reflects lower-level, application-to-operating system activity doesn't render the category useless, but it generate many more events than you might expect or want, and that makes analysis more complex.

Some objects (e.g., files) are never kept open for very long. Operations such as listing a folder or deleting a file or folder are just single atomic actions but still generate the open and close instances of event ID 560 and event ID 562 in the log.

Event ID 560 provides many description fields that cover the object accessed, the user and program involved, and the permissions requested. Figure 7-6 lists these fields.

The Windows Server 2003 Security Log Revealed

Description Field	Explanation
Object Type	File, Folder, Service, Registry Key, Printer
Object Name	Full path name of file or other type of object
Handle ID	Unique number identifying the object session during which the program has the object open; all subsequent access events triggered by the application will reflect the same Handle ID
Operation ID	Unknown
Process ID	Process ID of the program attempting to open the object; same Process ID as logged by event ID 528(see Chapter 6, Detailed Tracking)
Image File Name	Full path name of the program attempting to open the object
Primary User Name	User name of the account under which the program was started
Primary Domain	Domain name of the account under which the program was started
Primary Logon ID	Logon ID of the logon session during which the access attempt occurred; same Logon ID as logged by event ID 528 and event ID 540 (see Chapter 3, Understanding Authentication and Logon)
Client User Name	User name of the client account being impersonated by the server program, if applicable
Client Domain	Domain of the client account being impersonated by the server program, if applicable
Client Logon ID	User name of the client account being impersonated by the server program, if applicable
Accesses	The requested permissions requested (see Figure 7-2)
Privileges	Not used
Restricted SID Count	Unknown

Figure 7-6 Event ID 560 description fields

Event ID 560's Primary and Client description fields provide information about users attempting to access an object. Depending on the scenario, the information you need might be in either set of fields. When a user accesses an object on the same system to which he or she has logged on interactively, the application is directly accessing the object; Windows logs the user's identity in the Primary User Name and Primary Domain fields and leaves the client fields blank. When the user accesses a file in a shared folder on a remote computer from over the network, the application is no longer directly accessing the object. Instead, Windows redirects the open request over the network to the Server service on the remote system. In turn, the Server service opens the file on the user's behalf. Windows then fills in the Primary User Name and Primary Domain fields with the account under which the Server service is running (usually SYSTEM). Then Windows fills in the Client fields with the account of the user at the client workstation. Figure 7-7 illustrates these two scenarios.

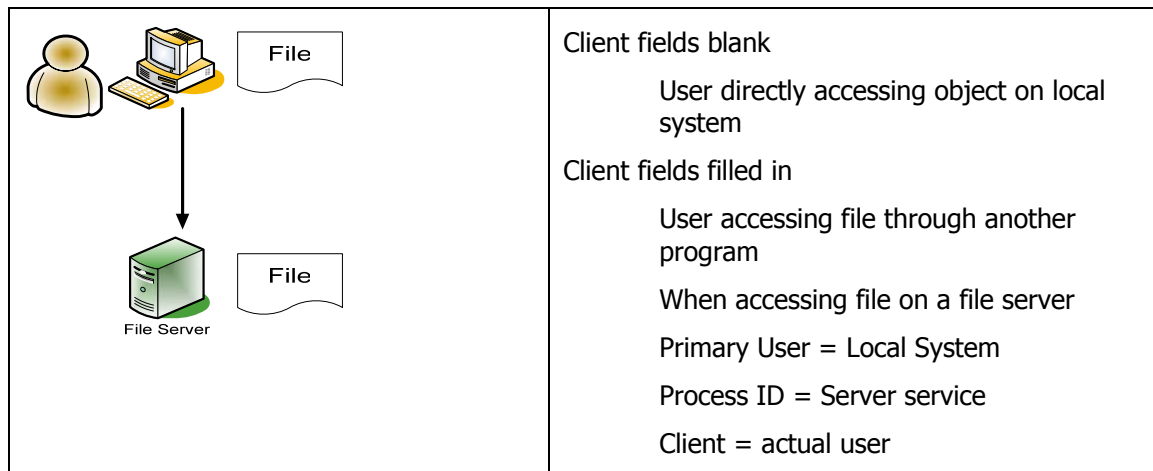


Figure 7-7 Primary and Client fields

Operation-Based Auditing

Event ID 560 logs the permissions requested by the application that's trying to open a handle to the object, but that doesn't mean the application actually exercised those permissions before closing the object. For instance, a user might successfully open an object for Read and Write access but close the file without ever changing its content. In Windows 2000, there's no way to know if the application used the permissions that are listed in successful instances of event ID 560. Windows 2003 introduces a new feature, called operation-based auditing, and the new event ID 567, Object Access. After a user successfully opens a handle to an object and then exercises one or more permissions, Windows logs event ID 567, which lists the Handle ID originally logged by event ID 560 as well as the permissions exercised. Any subsequent uses of the same permissions do not trigger duplicate instances of event ID 567; Windows simply logs event ID 567 the first time the user exercises a given permission between the opening and closing of the object. Unfortunately, event ID 567 lists only the

Handle ID, not the object's name, so you can't simply monitor the log for event ID 567 for the object name and type of permission you want to track. Instead, you must watch for event ID 560 for the desired object name and permission, then look for a subsequent event ID 567 with the same handle ID and permission to affirm the actual use of the permission as shown in Figure 7-8.

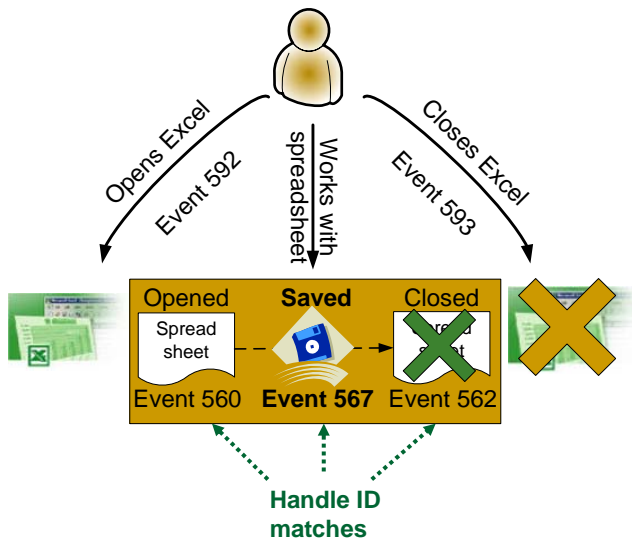


Figure 7-8 Operation-based auditing

Windows handles object deletions a little different than other access events. In addition to logging event ID 560, Windows also logs event ID 564, Object Deleted, which lists the handle ID originated in event ID 560. When successful delete access has been enabled for auditing on an object, Windows logs event ID 564 when that object is deleted. To determine the name of the deleted object, look for a prior event 560 with the same Handle ID. Typically, event 560 and event 564 will be in close proximity but it is possible for a process to open an object for delete access and then actually delete the object much later.

Sometimes when you install new updates to Windows, the installer is unable to replace existing files because they are in use by the operating system. In such cases, the common approach is to instruct NTFS to delete the object the next time the system boots but before the file is put into use. Windows logs such deferred deletions in event ID 563, Object Open for Delete. Windows does not log event ID 563 on typical file deletions. Microsoft documentation states that *An attempt was made to open an object with the intent to delete it. Note: This is used by file systems when the FILE_DELETE_ON_CLOSE flag is specified in Createfile().* This flag is the only way to delete files opened exclusively by another program.

The Object Access category also includes event ID 561, Handle Duplicated. Handle duplication is an application-level event which bears little if any relevance to security and auditing. Therefore, I recommend you ignore instances of event ID 561. All of these objects are listed in Figure 7-9.

Event ID	Type	Description
561	Success	Handle Duplicated
563	Success	Object Opened for Delete
564	Success	Object Deleted
567	Success	Object Access Attempt

Figure 7-9 Other object access events

You can use Object Access auditing to track who tried to access objects of interest, how they tried to access those objects, and whether they were successful. In Windows 2003, you can determine whether users who opened an object for a specific type of access actually used that access or closed the object without performing the operation. It would be easier if Windows logged the object's name in instances of event ID 564 and event ID 567, but you must connect event ID 560 and the subsequent event ID 564 or event ID 567 by using the Handle ID field. Such multiple-event correlation or pattern recognition is beyond the ability of most current event-log software, but I expect that to change as interest in the Security log continues to increase.

Chapter 8

Account Management

The Account Management Security log category is particularly valuable because you can use it to track maintenance of user, group, and computer objects in AD as well as to track local users and groups in member server and workstation SAMs. This category is also very easy to use because Windows uses a different event ID for each type of object and operation.

You can use Account Management events to track things like new user accounts, password resets, and new members being added to groups. Monitoring maintenance of domain users and groups can be a key aspect of compliance with legislation such as the Sarbanes-Oxley (SOX) Act and Health Insurance Portability and Accountability Act (HIPAA) because access to private or financially significant information is controlled largely through group membership and based on user-account authentication. When you enable this category on DCs, each DC will begin recording maintenance events against users, group, and computer objects executed against that DC. To get a complete record of all Account Management events for AD objects, you'll need to combine this category's activity from all your DC Security logs. Figure 8-1 shows all the event IDs associated with maintenance of users and groups.

			Created	Changed	Deleted	Member	
						Added	Removed
User			624	642	630	Not applicable	
Computer			645	646	647		
Groups	Security	Local	635	641	638	636	637
		Global	631	639	634	632	633
		Universal	658	659	662	660	661
	Distribution	Local	648	649	652	650	651
		Global	653	654	657	655	656
		Universal	663	664	667	665	666


 Indicates events are logged on DCs only

Figure 8-1 Account Management events

As you might have noted when viewing Figure 8-1, Windows logs different event IDs for each type and scope of group. You select the group's scope and type when you create the group, as Figure 8-2 shows.

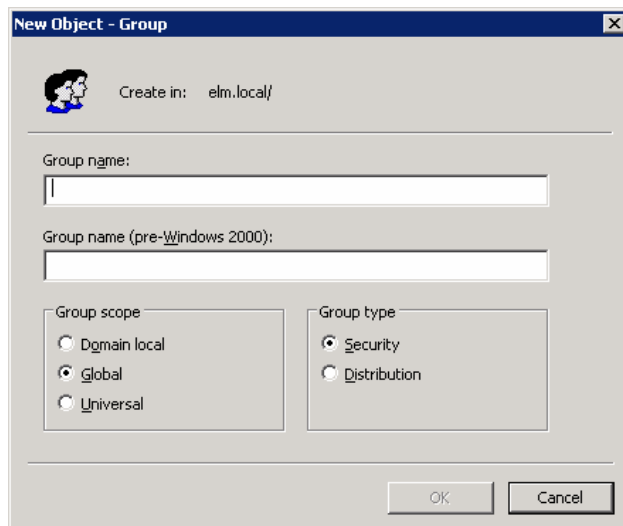


Figure 8-2 Selecting group scope and type

AD supports two types of groups. *Distribution groups* are irrelevant to security; you can't grant permissions or rights to distribution groups or use them for any other security-related purpose. Distribution groups serve as distribution lists (DLs) for Microsoft Outlook and Microsoft Exchange Server users. *Security groups* are, as the name would indicate, the type of groups you use whenever you grant or deny access; for what its worth, security groups can also be used as distribution groups. In the descriptions of Account Management events, Windows describes security groups as Security Enabled and distribution groups as Security Disabled. Aside from being one of these two types, a group is also configured as one of four scopes, as Figure 8-3 lists. A group's scope determines the computers on which the group can be used to control access as well as the types of users and groups that can be members of the group.

Scope	Potential Members	Permissions Granted	Where Defined
Universal	Users and global or universal groups from any domain in the forest	Anywhere in the forest	AD
Global	Users and other global groups from same the domain	Anywhere in the forest	AD
Domain local	Users and global or universal groups from any domain in the forest and domain local groups from the same domain	Within the same domain	AD
Machine local	Global or universal groups and users from any domain in the forest	Within the same system	Local SAM

Figure 8-3 Group scopes

Account Management events also provide important information for member servers. It's best to avoid using local SAM users and groups in member servers when domain accounts could be used instead because of the maintenance and weaker authentication issues that arise from using local accounts.⁵ Local user accounts are also the first target of intruders, who know that these accounts are less likely than domain accounts to be secure. In fact, a successful intruder often creates a legitimate local user account for use in the future, to avoid detection. Given the risks of local accounts (and especially when you have a policy to avoid using local accounts), you need to know when any activity associated with the creation or modification of local users and groups occurs. This is exactly what enabling *Audit account management* events on member servers provides. Because the local SAM contains no distribution, global, or universal groups, event IDs associated with these domain-only groups are not logged on member servers.

The description fields that Account Management events log are, for the most part, self explanatory or similar to description fields from the categories that I've already discussed. Windows uses the New or Target fields to identify the user, group, or computer that is the object of the event and identifies the object's name and domain. Windows uses the Caller User Name, Caller Domain, and Caller Logon ID fields to identify the user who performed the action.

The Account Management category uses a number of event IDs to track user-account changes. Account Management logs event ID 642 for any type of user-account change. The first description field usually provides a brief text description of what about the account was changed. For instance, when you reset a user's password, the first description string might simply say *password*

⁵ For more information about the risks associated with local accounts, see the *Windows IT Pro* article "Limiting Risk Associated with Local Accounts" at <http://www.windowsitpro.com/Article/ArticleID/42578/42578.html>.

reset. There's no list of the descriptions of all possible changes; if you want to filter instances of event ID 642 for a certain type of change you'll need to execute that type of change on a sample user account, find the instance of event ID 642, and capture the text description. Before going to that trouble, check Figure 8-4 to see whether the category offers an event ID for that specific type of user-account change. As you can see in Figure 8-4, Account Management provides additional event IDs that make it easier to identify certain types of user-account changes.

Note the difference between password changes and password resets. Password changes occur when a user changes his or her password. Password resets indicate that an administrator or other support person reset the password for a user.

Event ID	Description	Notes	
		Windows 2000	Windows 2003
626	User account enabled	Not logged	
627	Change password attempt	Logged for both password changes and resets	Logged only for password changes
628	User account password set	Not logged	Logged for password resets
644	User account locked out		
671	User account unlocked	Not logged	

Figure 8-4 Other user change events

Account Management's coverage of user account maintenance is well laid out, but you should be aware of one significant caveat. When you create a user account, you'll find an expected instance of event ID 624, *User account created*. But because of the way that the MMC Active Directory Users and Creators snap-in interacts with AD, you'll also see a series of event ID 642 instances interspersed with an instance of event ID 626, *User account enabled*, and event ID 628, *User account password set*. These "extra" instances of event ID 642, event ID 626, and event ID 628 can throw off reports or alerts you've set up unless your event-log management solution can do some multi-event correlation. If you can, filter out instances of these three event IDs when they are preceded within a couple seconds by an instance of event ID 624 that shares the same target user name.

I have one other note about auditing user-account changes on Windows 2003. Windows 2003 adds almost 20 new description fields to event ID 642. In Windows 2000, event ID 642 simply notes the fact that a specified administrator

had changed a specified user account, and usually provides a brief text explanation of the change. Windows 2003 provides much more granular information about which user account attributes were changed, including the options and restrictions found on the Account tab of the user account's Properties dialog box in the MMC Active Directory Users and Computers snap-in.

Account Management logs just two other events. Event ID 643 informs you when the domain's password or lockout policy is changed. In some cases, Windows 2000 DCs log this event each time they apply Group Policy (5 minutes by default), even when the policy settings remain unchanged. (To fix this behavior, apply the most recent Windows 2000 service pack.) Windows 2003 DCs correctly log event ID 643 (i.e., only when the settings have changed). Furthermore, Windows 2003 provides new description fields that spell out the new values of the policy settings that changed. Windows 2003 also logs event ID 668 whenever a group's type or scope changes. Both of these events are listed in figure Figure 8-5.

Windows 2003 only	Event ID	Description
•	668	Group type changed
	643	Domain policy changed

Figure 8-5 Other Account Management events

Account Management is one of the best designed Security log categories. You can use this category to track important maintenance to user accounts and groups. The category's use of many distinct event IDs simplifies the filtering process. (Don't forget about the "extra" events that Windows logs in connection with user-account creations.) Account Management does a good job of auditing changes to users, groups, and computers, but what about important AD objects such as OUs, GPOs, and domains? The Directory Service Access category provides granular tracking of any class of object in AD, even down to the attribute level.

Chapter 9

Directory Service Access Events

Whereas Account Management events provide excellent auditing of user, group, and computer maintenance, Directory Service Access events make low-level auditing available for all types of objects in AD. Directory Service Access events not only identify the object that was accessed and by whom but also document exactly which object properties were accessed. Directory Service Access events work a lot like Object Access events because you must first enable the audit policy at the system level, then activate auditing on the specific objects you want to monitor.

To enable auditing on a file, open the file's Properties dialog box from within Windows Explorer, select the Security tab, click Advanced, and then select the Auditing tab on the Advanced Security Settings dialog box. To enable auditing on an AD object, follow the same path but from within the Active Directory Users and Computers snap-in (rather than Windows Explorer), as Figure 9-1 shows. Then, specify the permissions you want to audit when users request access to the object.

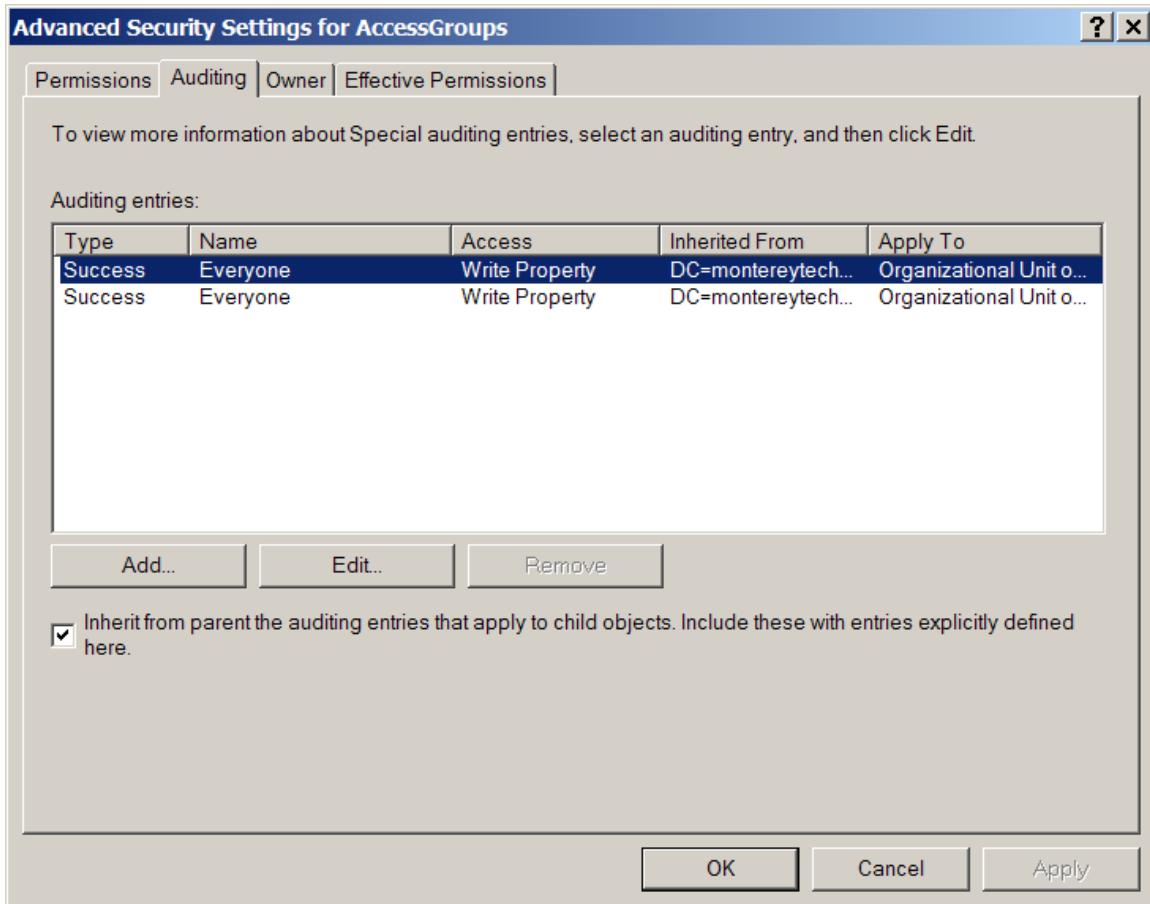


Figure 9-1 Audit policy for an OU

At this point, the procedure for auditing AD objects diverges a bit from file-system auditing. For files, you simply select auditing for the permissions that can be performed against the file. However, AD has two types of operations: operations performed against the object, and Read and Write operations against individual properties of the object. These two types of operations correspond to the Object and Properties tabs that Figure 9-2 shows.

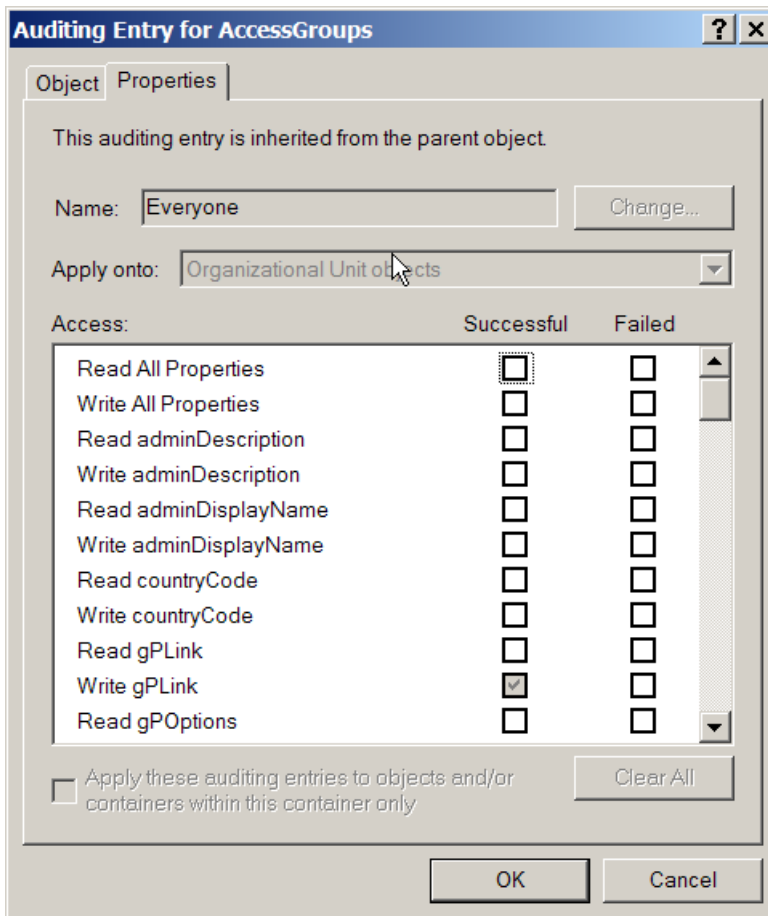


Figure 9-2 Auditing entry for an OU

Each class of object in AD has a set of applicable operations that you can perform on that object and on its properties. Properties are individual settings within an object. For example, a user object has many properties, including phone number, photo, and account expiration date. Certain permissions are common to all classes of objects:

- Full control
- List contents
- Read all properties
- Write all properties
- Delete
- Delete subtree
- Read permissions
- Modify permissions
- Modify owner

Some of these permissions are actually groups of lower-level permissions. For example, *Full control* gives Read and Write access to an object and every property of that object. AD objects also have class-specific permissions appropriate for the type of object. For example, user objects have a *Reset password* permission. AD objects also have Read and Write permissions for each object property. For example, user objects have individual Read/Write permissions for account expiration, company, department, and so forth. This setup allows for granular access control as well as granular auditing.

Just as , defining audit policy at the folder level is preferable in file system auditing, auditing at the OU level and allowing the audit policy to propagate down to the desired child objects is preferable in AD-object auditing. By default, permission and audit entries flow down from parent OUs to child objects, through inheritance. This process lets you control permissions and auditing for entire branches of the domain at the highest appropriate level.

You can customize an object's audit policy to prevent its parent's audit entries from flowing down to it. If the object is an OU, you can also control whether and how audit entries flow from it down to its children. When an object is created, its initial audit policy is derived by processing the parent's audit policy. For container objects such as OUs, each audit entry's *Apply onto* setting specifies whether and how it should be propagated down to child objects:

- This object only
- This object and all child objects
- Child objects only
- Group objects
- Users objects
- Computer objects
- Organizational Unit objects
- Etc.

When the *Apply these auditing entries to objects and/or containers within this container only* checkbox in the Auditing Entry is selected, recursion is limited to the immediate OU and audit entries will not propagate below the immediate children of the OU.

When an object is created, the system examines the parent's audit policy and copies each applicable entry to the new object's audit policy. Later, if the parent's audit policy changes, the changes are repeated on the child.

If the *Inherit from parent the auditing entries that apply to child objects* checkbox is subsequently cleared, you can choose to have all inherited entries deleted from the object's audit policy or added as non-inherited entries. All future changes to upper-parent audit policy will not propagate to this object

unless you use the Default button (which Figure 9-2 shows) to reset the audit policy to each object's default.

Directory Service Access Event IDs

Whereas Account Management logs more than 40 event IDs, Directory Service Access records just two event IDs, as listed in Figure 9-3. Because these events pertain strictly to AD, only DCs log Directory Service Access events. Enabling this audit policy on member servers and workstations has no effect.

Windows 2003 only	Event ID	Type	Description
	565	Success Failure	Object open
•	566	Success	Object operation

Figure 9-3 Directory Service Access event IDs

As you can see in Figure 9-3, Windows 2003 adds event ID 566 as part of the new operation-based auditing feature that I first introduced while discussing file-system auditing on page 60. Event ID 565 is similar to event ID 560 but is limited to recording access-request events on AD objects. Whereas event ID 565 logs the permissions requested by a user or program, event ID 566 logs the permissions actually exercised by the user or program after opening the object. An object might be accessed several times during the same open session, but Windows logs event ID 566 only the first time a given permission is exercised. This event is similar to event ID 567 but is limited to AD object access. Thankfully, there's no need to correlate event ID 566 with event ID 565 to determine the target object's name because event ID 566 itself identifies the object by its X.500 distinguished name (DN). Therefore, if your DCs are Windows 2003 systems, you can concentrate on event ID 566 and ignore event ID 565. (If you are auditing for failed attempts to access certain AD objects, you'll need to continue monitoring for failure type instances of event ID 565.) Both event ID 565 and event ID 566 share the description fields that Figure 9-4 lists.

Description Field	Explanation
Object Type	User, organizationalUnit, gpContainer, etc. Values for this field correspond directly to the class names defined in AD's schema
Object Name	X.500 DN of the accessed object
Handle ID	Blank

Description Field	Explanation
Primary User Name	Computer name of the DC
Primary Domain	Domain of the DC
Primary Logon ID	Logon session of the AD process
Client User Name	User name of the client who requested access
Client Domain	Domain of the client who requested access
Client Logon ID	Logon ID of the logon session of the client who requested access This the same Logon ID as logged by event ID 528 and event ID 540 (see Chapter 3: Understanding Authentication and Logon)
Accesses	Permissions requested
Properties	Names of the properties for which the client requested access; these correspond directly to the attribute names from AD's schema This event lists property group names such as Public Information
Additional Info: Additional Info2: Access Mask:0x20	Unknown

Figure 9-4 Event ID 565 and Event ID 565 Description Fields

Change Control Monitoring for AD

Compliance issues have made the best practice of change control more important than ever, and nowhere is being able to determine who changed what more important than in AD. A misstep in AD can adversely affect thousands of users or computers within minutes.

As I mentioned in Chapter 8, the Account Management category amply supports change control for user, group, and computer maintenance. But administrators carry out plenty of other work in AD, and to track that you need to use Directory Service Access events.

I'm going to focus on the two areas of AD maintenance that most frequently cause security or availability problems: changes to administrator authority and changes to Group Policy.

Auditing Changes to Administrator Authority in AD

Initially, all administrator authority in AD is controlled through the administrator and operator groups that are created when AD is installed. These groups include Enterprise Admins, Schema Admins, Domain Admins, Administrators, and Account Operators in particular.

Larger IT departments that have specialized roles and separation of duty requirements should avoid using these groups except for a handful of top-level administrators. Such organizations should take advantage of AD's *delegation of control* feature, which you can use to follow least privilege and separation of duty, two precepts that are central to a mature, controlled IT security practice.

Top-level administrators can use AD's delegation of control feature to delegate selected permissions over selected objects to groups created for various roles within IT. For instance, imagine that the Help desk needs authority to reset passwords for forgetful users. Instead of making Help desk staffers members of Administrators or Account Operators, you can simply create a group called HelpDesk and grant the Reset Password permission over user accounts in the ACL of each appropriate OU.

Delegation of control in AD is essential to limit the risk posed by too many all-powerful administrators. But you obviously need a way to track delegation of control events. Directory Service Access events make this task easy. In general, AD authority is delegated at the root of the domain and in OUs. Therefore, you simply need to create two audit entries at the root of the domain. As Figure 9-5 shows, I created one audit entry that activates auditing of Modify Permissions and Modify Owner on the root of the domain. Then I added another entry that audits the same permissions for OUs. Why not just use a single entry that applies to all objects from the domain root down? Such an audit policy would result in an overwhelming amount of events in the Security log when permissions were changed on an OU that has many child objects; you'd get an event for every object in that OU and below.

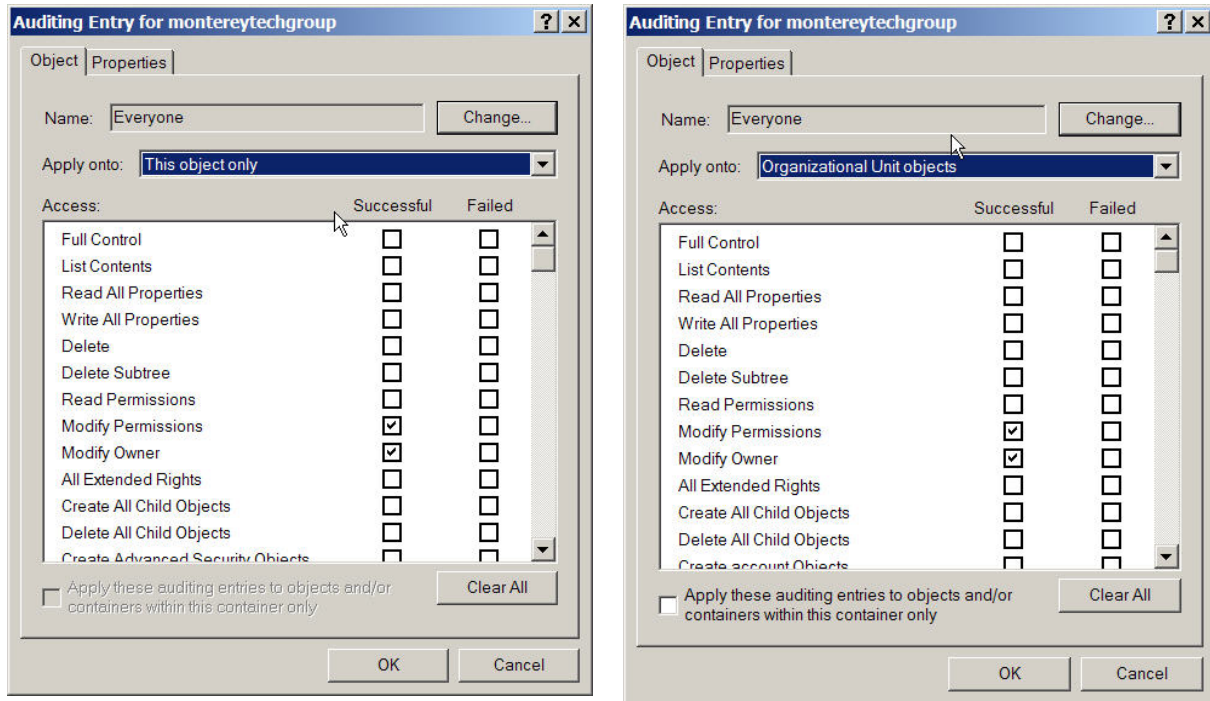


Figure 9-5 Audit entries for tracking delegation of control

Figure 9-6 shows a sample event ID 566 instance in which MTG\rsmith has modified permissions on the Eastern US User Accounts OU. The WRITE_DAC permission in the event's description indicates that the OU's discretionary ACL (DACL) of the was the focus of the change. As you can see, event ID 566 doesn't tell us details of the change (e.g., the permissions changed) but it does alert us to the fact the ACL was changed, tell us whichOU was affected, and who made the change.

Event Type:	Success Audit
Event Source:	Security
Event Category:	Directory Service Access
Event ID:	566
Date:	12/29/2005
Time:	3:08:03 PM
User:	MTG\rsmith
Computer:	MTG1
Description:	
Object Operation:	
Object Server:	DS
Operation Type:	Object Access
Object Type:	organizationalUnit
Object Name:	OU=Eastern US User
Accounts,OU=test,DC=montereytechgroup,DC=com	
Handle ID:	-
Primary User Name:	MTG1\$
Primary Domain:	MTG
Primary Logon ID:	(0x0,0x3E7)
Client User Name:	rsmith
Client Domain:	MTG
Client Logon ID:	(0x0,0x100339CC)
Accesses:	WRITE_DAC
Properties:	
WRITE_DAC	
organizationalUnit	
Additional Info:	
Additional Info2:	
Access Mask:	0x40000

Figure 9-6 Event showing delegation of control

In addition to auditing permission changes on the domain root and OUs, you should audit changes to the ACLs of GPOs. There are two reasons for modifying GPO permissions. First, you might modify GPO permissions to give someone else the ability to edit the GPO. Second, you might use the Apply Group Policy permission to limit application of a GPO to a subset of users or computers that would otherwise receive the GPO's policies.⁶ To enable auditing of Modify Permission and Modify Owner for all current and future GPOs, open Active Directory Users and Computers and then enable View\Advanced Features, which reveals the System folder. Maneuver to the System\Policies folder, which is the

⁶ For more information about using a GPO's permissions to limit its application, see my *Windows IT Pro* article "Don't Shoot Yourself in the Foot with Group Policy Security Settings, Part 1".

physical location of GPOs in AD, and create the two audit entries as defined for the domain root and OUs but select groupPolicyContainer objects as the choice for the *Apply onto* setting. groupPolicyContainer is the system name of GPOs in AD's schema. At this point, you'll see an instance of event ID 565 or event ID 566 whenever someone modifies the permissions or takes ownership of the domain root, Ous, or GPOs. If you want even more change-control auditing, you can define similar audit policies on objects in the MMC Active Directory Sites and Services console, thus tracking changes to your forest's site topology.

Tracking Changes to Group Policy

Group Policy is a powerful tool for managing thousands of computer configurations and user-level settings. And as with any powerful tool, you can shoot yourself in the foot if you aren't careful. A bad change in the wrong GPO can potentially take out an entire domain, so having an audit trail of Group Policy changes is an obvious requirement for any controlled environment.

To establish such an audit trail, add an audit entry to the System\Policies folder activating auditing of Write All Properties on groupPolicyContainer objects. Then look for instances of event ID 566 (or event ID 565 on Windows 2000 DCs) in which the object type is groupPolicyContainer, the Accesses field specifies Write Property, and the Properties field lists versionNumber. Whenever someone edits a GPO, AD increments the versionNumber property of the GPO. The specified audit policy will generate an event like the one depicted in Figure 9-7. To determine which GPO was changed, refer to the Object Name in the event's description. As you can see in Figure 9-7, event ID 566 doesn't provide the GPO's friendly name, which you are accustomed to seeing in Active Directory Users and Computers or the MMC Group Policy Management Console. Instead, the event lists the GPO's globally unique identifier (GUID). To figure out exactly which GPO that GUID identifies, you can use the Group Policy Management Console. Select a given GPO and then select the Details tab of the GPO. The Unique ID field specifies the GUID of the GPO.

Event Type:	Success Audit
Event Source:	Security
Event Category:	Directory Service Access
Event ID:	566
Date:	12/29/2005
Time:	4:00:37 PM
User:	MTG\rsmith
Computer:	MTG1
Description:	

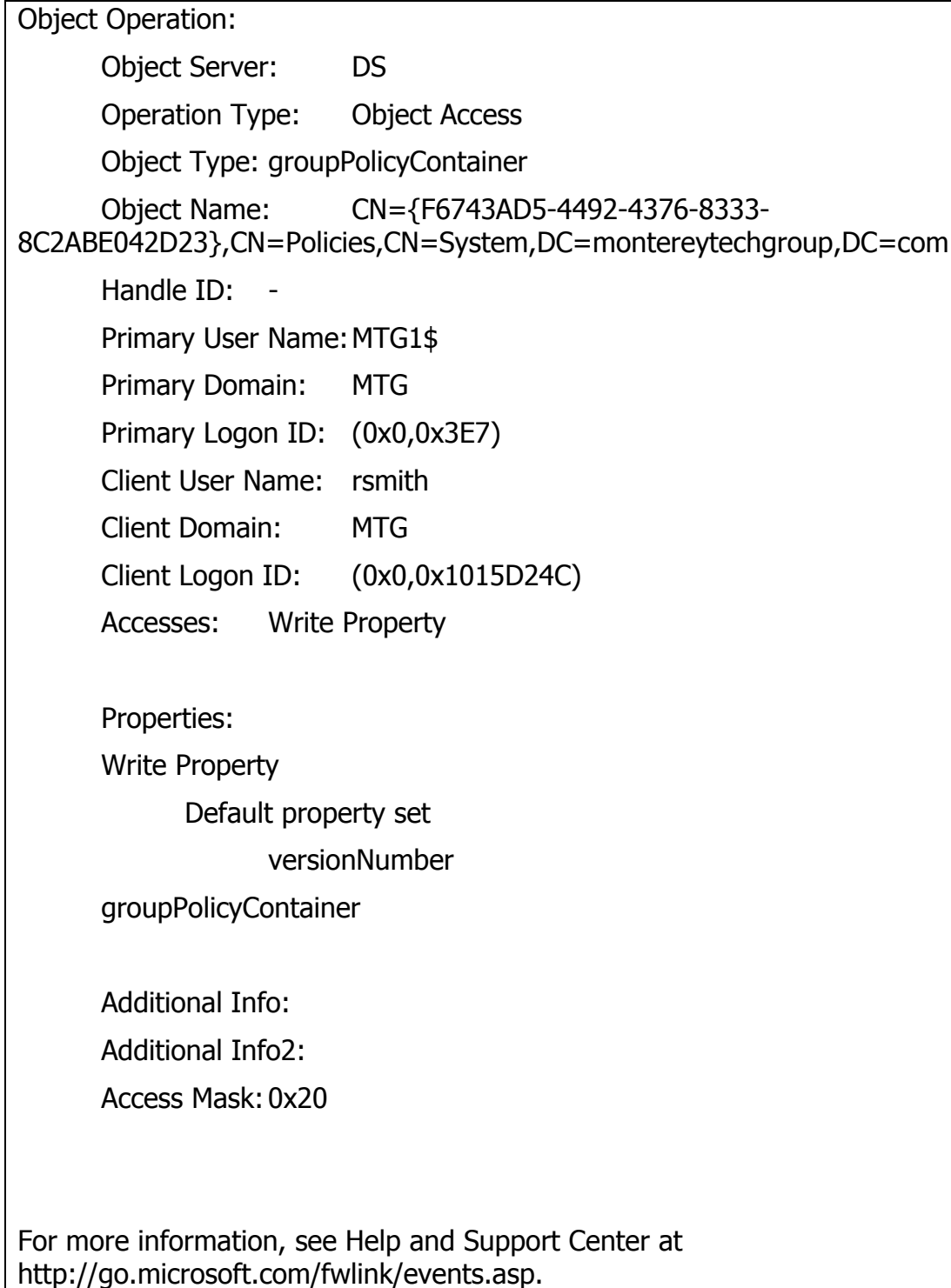


Figure 9-7 GPO modification

Besides direct modification of GPOs you can also greatly alter the application of Group Policy by linking or unlinking a GPO to an OU, domain root, or site or by enabling the *Block policy inheritance* option on an OU. You manage GPO links

and the *Block policy inheritance* check box on the Group Policy tab of the Properties dialog box of OUs, domain roots, and sites, as shown in Figure 9-8.

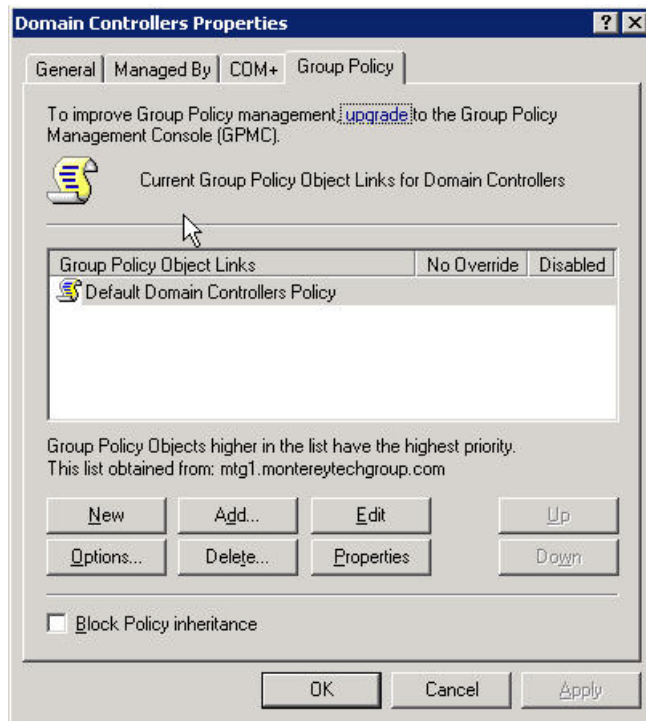


Figure 9-8 Group policy tab

Internally, two properties on OUs, domain roots, and site objects correspond to the settings you see in Figure 9-8. The gPLink property defines the GPOs listed in Group Policy Object Links and holds the No Override and Disabled check box columns as well. The Block Policy Inheritance check box corresponds to the the gPOptions property. Create an audit policy at the root of the domain to monitor Write access for these two properties for the domain root and OUs. To be thorough, you can enable the same audit policy on site objects in Active Directory Sites and Services. Then monitor your DC Security logs for events similar to the one that Figure 9-6 shows, except that the Accesses field will list Write Property and the Properties field will specify gPLink or gPOptions. Again, the event doesn't tell you the full details of the change but does alert you that something about Group Policy was changed for the specified OU, domain, or site as well as when the change happened and who made the change.

Directory Service Access events provide an audit trail for an important area of your IT infrastructure. By enabling auditing of OUs, domain roots, sites, and GPOs, you can effectively track changes in administrative authority and catch potentially wide-sweeping effects of Group Policy modifications.

Chapter 10

Privilege Use Events

You can use the Privilege Use category to track the exercise of user rights. Microsoft uses the terms *privilege*, *right*, and *permission* inconsistently. In this case, *privileges* refer to the user rights you find in Local Security Policy under Security Settings\Local Policies\User Right Assignment, as Figure 10-1 shows.

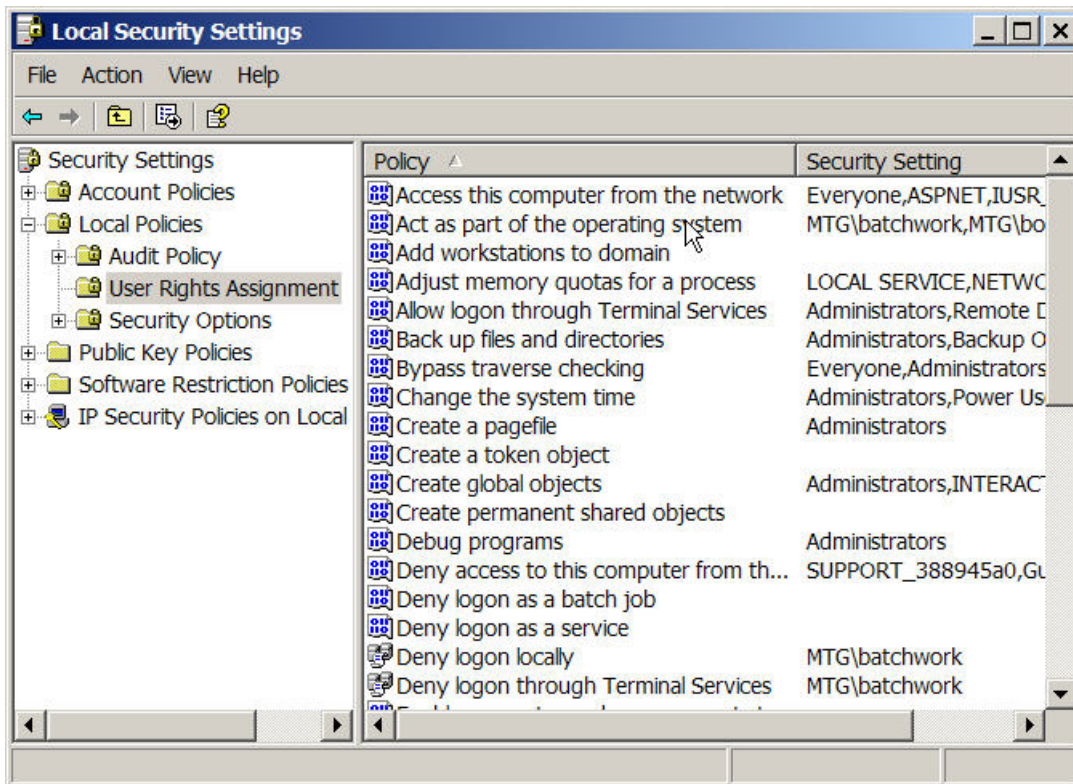


Figure 10-1 User rights configuration

Figure 10-2 lists the 3 events logged by this category.

Event ID	Type	Description
576	Success	Special privileges assigned to new logon
577	Success Failure	Privileged service called
578	Success Failure	Privileged object operation

Figure 10-2 Privilege Use events

For most user rights, Windows logs a Privilege Use event when a user actually exercises the right. However some rights are exercised so frequently during the normal course of a user's activities that the Security log would quickly fill if Windows were to log each use. Among those high-use rights are *Back up files and directories* and *Restore files and directories*, but you can force Windows to log these two rights by enabling the *Audit: Audit the use of Backup and Restore privilege* security option. Be aware that enabling this option will result in a Privilege Use event being logged for every single file, folder, and other object during system backups, which will overwhelm your log with events of questionable value.

For normal user rights, Windows logs either event ID 577 or event ID 578 when the event is exercised. I'm not aware of any rationale that explains why Windows logs one event ID for a given right as opposed to another. Figure 10-3 lists each user right and specifies which event ID Windows uses to log it use. Note that logon rights are never logged by Privilege Use events because the use of logon rights is already documented by the Logon/Logoff category.

Chapter 10 – Privilege Use Events

System name	Description	Notes
SeNetworkLogonRight	Access this computer from the network	Never logged (see event ID 540)
SeTcbPrivilege	Act as part of the operating system	Logged by event ID 577 (also logged by event ID 576 at logon)
SeMachineAccountPrivilege	Add workstations to domain	Logged by 577
SeIncreaseQuotaPrivilege	Adjust memory quotas for a process	Not observed
SeRemoteInteractiveLogonRight	Allow logon through Terminal Services	Never logged (see event ID 528)
SeBackupPrivilege	Back up files and directories	Logged only when <i>Audit: Audit the use of Backup and Restore privilege</i> is enabled (also logged by 576 at logon)
SeChangeNotifyPrivilege	Bypass traverse checking	Never logged
SeSystemtimePrivilege	Change the system time	Logged only by non-services
SeCreatePagefilePrivilege	Create a pagefile	Not observed
SeCreateTokenPrivilege	Create a token object	Logged only at initial logon (see event ID 576)
SeCreatePermanentPrivilege	Create permanent shared objects	Logged by event ID 577
SeDebugPrivilege	Debug programs	Logged only at initial logon (see event ID 576)
SeDenyNetworkLogonRight	Deny access to this computer from the network	Never logged (see event ID 534)
SeDenyBatchLogonRight	Deny logon as a batch job	Never logged (see event ID 534)
SeDenyServiceLogonRight	Deny logon as a service	Never logged (see event ID 534)
SeDenyInteractiveLogonRight	Deny logon locally	Never logged (see event ID 534)

The Windows Server 2003 Security Log Revealed

System name	Description	Notes
SeDenyRemoteInteractiveLogon Right	Deny logon through Terminal Services	Never logged (see event ID 534)
SeEnableDelegationPrivilege	Enable computer and user accounts to be trusted for delegation	Logged by event 577 (also logged by event ID 576 at logon)
SeRemoteShutdownPrivilege	Force shutdown from a remote system	Logged by event ID 578
SeAuditPrivilege	Generate security audits	Logged only at initial logon (see event ID 576)
SeIncreaseBasePriorityPrivilege	Increase scheduling priority	Not observed
SeImpersonatePrivilege	Impersonate a client after authentication	Not observed
SeLoadDriverPrivilege	Load and unload device drivers	Logged by event ID 577 (also logged by event ID 576 at logon)
SeLockMemoryPrivilege	Lock pages in memory	Not observed
SeBatchLogonRight	Log on as a batch job	Never logged (see event ID 528)
SeServiceLogonRight	Log on as a service	Never logged (see event ID 528)
SeInteractiveLogonRight	Log on locally	Never logged (see event ID 528)
SeSecurityPrivilege	Manage auditing and Security log	Not logged for Security log views (logged by event ID 578 when updating object's audit policy; also logged by event ID 576 at logon)
SeSystemEnvironmentPrivilege	Modify firmware environment values	Logged by event ID 576 at logon
SeManageVolumePrivilege	Perform volume maintenance tasks	Not observed
SeProfileSingleProcessPrivilege	Profile single process	Not observed
SeSystemProfilePrivilege	Profile system	Not observed

System name	Description	Notes
	performance	
SeUndockPrivilege	Remove computer from docking station	Unimportant
SeAssignPrimaryTokenPrivilege	Replace a process level token	Logged only at initial logon(see event ID 576)
SeRestorePrivilege	Restore files and directories	Logged only when <i>Audit: Audit the use of Backup and Restore privilege</i> is enabled (also logged by event ID 576 at logon)
SeShutdownPrivilege	Shut down the system	Logged by event ID 577 and event ID 578
SeSyncAgentPrivilege	Synchronize directory service data	Unused
SeTakeOwnershipPrivilege	Take ownership of files or other objects	Logged by event ID 578 (also logged by event ID 576 at logon)

Figure 10-3 User rights

As of the release of Windows 2003 Service Pack 1 (SP1), Windows uses event ID 576 at the time of logon to record the fact that the user holds some administrator-equivalent rights. Microsoft defines *administrator-equivalent rights* as those that can be used to elevate the user to administrator authority or to compromise the audit trail. These administrator-equivalent rights are documented in Figure 10-3 as being logged by event ID 576. Just to be clear, event ID 576 simply lists the administrator-equivalent rights held by the user at logon. The event does not indicate that the user exercised any of the rights. Some administrator-equivalent rights also generate event ID 577 or event ID 578 at the time of use.

All three event IDs specify in their descriptions the user who exercised or holds the right or rights, and list the rights in the Privileges description string.

As indicated in Figure 10-3, there are a number of rights that I've never observed being logged. All things considered, I don't find the Privilege Use category particularly useful, and it generates a massive amount of useless noise on DCs associated with computer accounts.

Chapter 11

Policy Change Events

The Policy Change category provides notification of changes to important security policies on the local system, such as to the system's audit policy or, in the case of DCs, to trust relationships.

Windows logs event ID 612 when it detects a change to the system's audit policy as shown in Figure 11-1.

Event ID	Type	Description
612	Success	Audit Policy Change

Figure 11-1 Event ID 612

This is an important event that might indicate tampering with the Security log. Fortunately, the event tells you the new status of all nine audit policies, as Figure 11-2 shows. Unfortunately, the Changed By fields don't tell you which administrator modified the audit policy; the user name always lists the computer name of the local system. In Windows 2000 and later, no one directly modifies security settings such as audit policy; instead, you edit the system's local GPO or a GPO stored in AD. Subsequently, the system applies Group Policy and so it's the system that makes the actual configuration settings based on the resultant set of policy derived from all applicable GPOs.

Audit Policy Change:		
New Policy:		
	Success	Failure
	+	-
	-	-
	-	-
	+	-
	+	-
	+	-
	-	-
	+	-
	+	-
Changed By:		
User Name: W3DC\$		
Domain Name: WORKGROUP		
Logon ID: (0x0,0x3E7)		

Figure 11-2 Event ID 612 example

Another area of policy change that the Policy Change category logs are changes to user right assignments. Windows logs event ID 608 when a user right is

assigned and event ID 609 when a user right is revoked. Don't confuse event ID 608 and event ID 609 with the Privilege Use events I described in Chapter 10. Event ID 608 and event ID 609 log the assignment or revocation of a user right, whereas Privilege Use events logs the actual use of rights. These two Policy Change events log the user or group who was the target of the change as well as the system name of the right or rights that were assigned or revoked. You can refer to Figure 10-3 to cross reference a right's system and friendly names. Like event ID 617, these two events don't tell you the real user who assigned or revoked the right or rights; the Changed By field always lists the local computer's account.

Event ID	Type	Description
608	Success	User right assigned
609	Success	User right removed

Figure 11-3 Event IDs 608 and 609

Don't be confused if you grant or revoke a logon right in Windows 2003, then can't find the corresponding instance of event ID 608 or event ID 609. Window 2003 uses two other event IDs—event ID 621 and event ID 622—to log assignment and revocation of logon rights such as *Access this computer from the network* or *Log on locally*.

Event ID	Type	Description
621	Success	System Security Access Granted
622	Success	System Security Access Removed

Figure 11-4 Event ID 621 and Event ID 622

Some rights are extremely powerful, to the point of giving the user administrator-equivalent authority, so event ID 608 is valuable because it provides a way to audit the assignment of user and logon rights. Event ID 621 gives you a way to track changes in how accounts are allowed to logon.

The Policy Change category uses three event IDs to log changes to domain trust relationships, as listed in Figure 11-5. All three event IDs specify Trusted Domain, but all three events are logged for both trusted and trusting domain relationship changes.

Event ID	Windows		Description
	2000	2003	
610	•	•	W2k: New Trusted Domain
			W3: New Trusted Domain
672	•	•	W2k: Removing Trusted Domain
			W3: Trusted Domain Removed
611	•	•	Trusted Domain Information Modified

Figure 11-5 Trust relationship change events

Windows logs event ID 610 when an administrator creates a new trust relationship (trusted or trusting). For some reason, Windows 2000 always logs two instances of event ID 610 and specifies who established the trust but tells you nothing about the trust type or its attributes. Windows 2003 logs one instance and specifies information about the trust's type, direction, and other attributes. Windows logs event ID 611, which has the same characteristics as event ID 610 but is used for trust deletions.

In addition to the preceding two events, Windows also logs event ID 620 when you create, delete, or modify an existing trust relationship. Windows 2003 provides more information about what attributes of the trust relationship were affected.

Windows logs some events associated with the status of IPSec, as shown in Figure 11-6, but these events have little if any value.

Event ID	Windows		Description
	2000	2003	
615	•	•	IPSec Services
613	Does not exist		IPSec policy agent started
614			IPSec policy agent disabled
611	Unconfirmed		IPSec policy agent encountered a potentially serious failure

Figure 11-6 IPSec events

When an administrator changes the domain's Kerberos policy, as shown in Figure 11-8, Windows logs event ID 617 (Figure 11-7).

Event ID	Type	Description
617	Success	Kerberos Policy Changed

Figure 11-7 Event ID 617

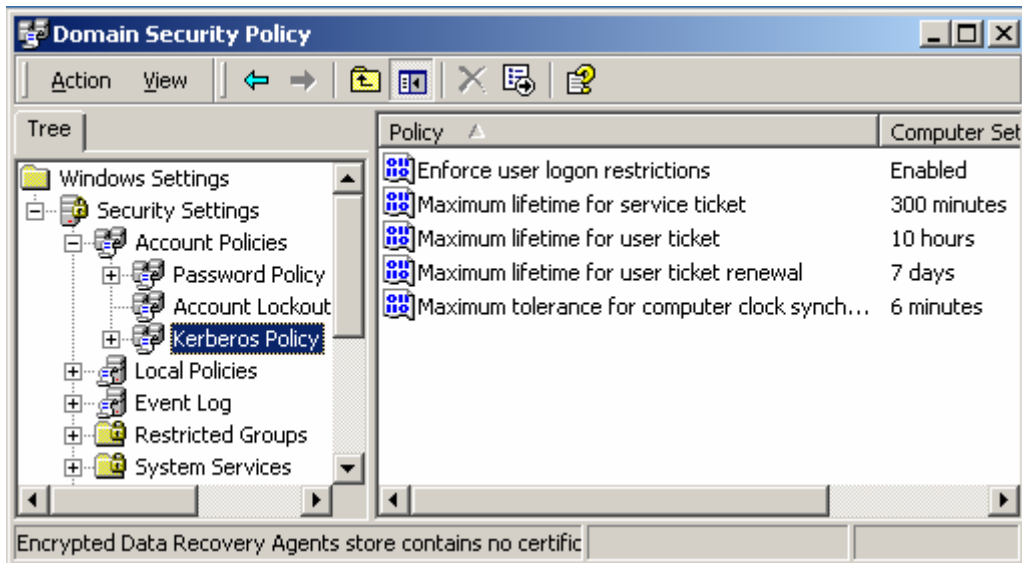


Figure 11-8 Kerberos policy

When an administrator modifies the data-recovery policy for the Encrypting File System (EFS) as shown in Figure 11-10, Windows logs event ID 618 (Figure 11-9) and specifies the old and new values for the policy. This is an important event because anyone who can edit a GPO can add himself or herself as a data recovery agent to one or more computers and thereby access information encrypted by other users. Sometimes Windows logs event ID 618 when nothing has changed, so examine the description of the event to determine its relevance.

Event ID	Type	Description
618	Success	Encrypted Data Recovery Policy Changed

Figure 11-9 Event ID 618

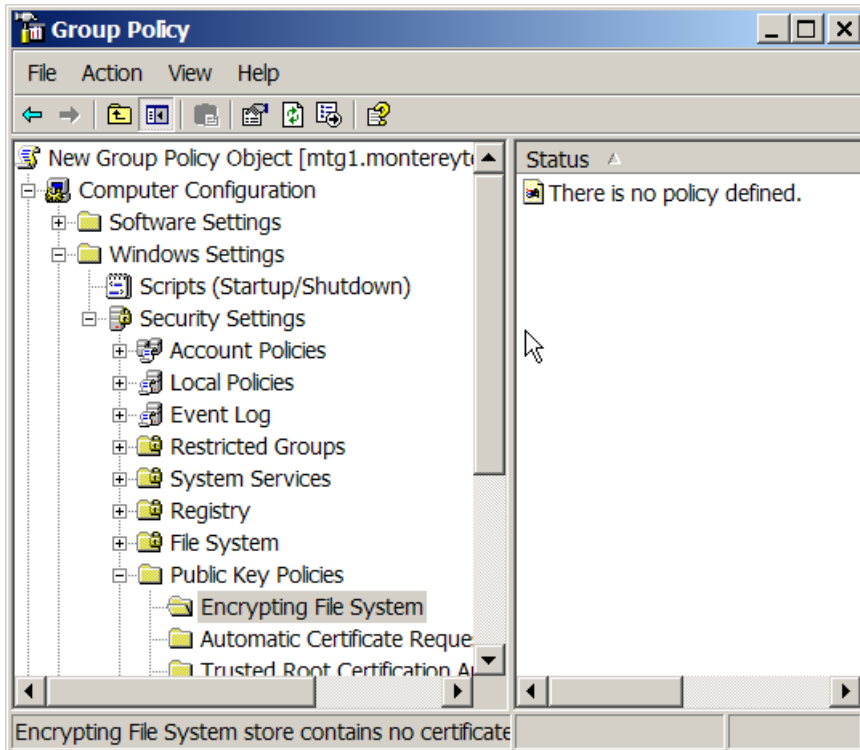


Figure 11-10 Encrypting File System data recovery policy

Windows logs event ID 619 when someone modifies the computer's Quality of Service (QoS) policy, which relates to how Windows can prioritize different types of TCP/IP traffic. Sometimes Windows logs event ID 619 (Figure 11-11) when nothing has changed, so examine the description of the event to determine whether the event is relevant.

Event ID	Type	Description
619	Success	Quality of Service Policy Changed

Figure 11-11 Event ID 619

Policy Change events provide a few important notifications, such as changes to audit policy, user right assignments, EFS recovery policy, and trust relationships. Unfortunately, these events only notify you of the change and though they might specify the before and after values for the changed settings, they don't tell you who caused the change. To track down the user behind the change, you'll have to examine the Directory Service events that I described in Chapter 9 to look for recent changes to GPOs.

Windows 2003 introduces a new feature called *per user selective auditing* and logs two events associated with this feature, as listed in Figure 11-12. Per user selective auditing is supposed to let you enable or disable system-wide auditing for specific accounts. The feature hints at some exciting capabilities but falls

short because you can't use groups in place of user accounts. The feature evidently was added to Windows to meet an obscure requirement of Common Criteria.⁷

Event ID	Type	Description
806	Success	Per User Audit Policy was refreshed (some Microsoft documentation refers to this event as event ID 623)
807	Success	Per user auditing policy set for user (some Microsoft documentation refers to this event as event ID 625)

Figure 11-12 Per user selective audit policy events

⁷ For more information on per user selective audit policy Roger Grimes' *Windows IT Pro* article "Per-User Auditing" at <http://www.windowsitpro.com/Article/ArticleID/46625/46625.html>.

Chapter 12

System Events

System Events are an eclectic mix of system events relevant to security. Windows logs event ID 612 and event ID 613 when the system starts up and shuts down, respectively. These events are important because a Windows system is completely vulnerable while shutdown and at the mercy of anyone who has physical access and the proper skill.

Event ID	Type	Description
512	Success	Windows NT is starting up
513	Success	Windows NT is shutting down

Figure 12-1 Startup and shutdown events

The Windows security infrastructure is designed to be modular and to facilitate new, plug-in security functionality from Microsoft and third-party vendors. These plug-ins can be authentication packages, trusted logon processes, or notification packages. Because these plug-ins are completely trusted modules of code that augment the operating system, Windows logs each plug-in as it loads, using the events that Figure 12-2 lists.

Event ID	Type	Description
514	Success	An authentication package has been loaded
515	Success	A trusted logon process has registered
518	Success	An notification package has been loaded

Figure 12-2 Security plug-in events

System Events also provides two events related to Security log integrity, as listed in Figure 12-3. Event ID 517 documents that someone possessing the *Manage auditing and Security log right* cleared the Security log. Event ID 516 can be logged during periods of extreme load but is very unusual. Windows logs these events regardless of the status the *Audit system events* policy.

Event ID	Type	Description
516	Success	Internal resources allocated for the queuing of audit messages have been exhausted
517	Success	The audit log was cleared

Figure 12-3 Events related to Security log integrity

Another useful event in this category is event ID 520, *The system time was changed*. Event ID 520 specifies the previous and new date and time as well as the executable and user who changed the time. Event ID 519 has not been observed in the field.

Event ID	Type	Description
519	Success	A process is using an invalid local procedure call (LPC) port
520	Success	The system time was changed

Figure 12-4 Miscellaneous events

Windows logs the most important event in this category, event ID 517, whether System Events is activated for auditing or not. However, it's worth enabling this category so that you have an audit trail of system restarts. Enabling Audit system events does not create an undue amount of noise.

Chapter 13

Getting the Most from the Security Log

Even a handful of servers creates more Security log data than you can hope to monitor and analyze manually. LogParser is a terrific free utility that can help you with the task but ultimately most organizations see the need for a full Security log-management solution.

If you are evaluating such solutions, make sure you select one that fits your needs. If you have more than a dozen servers, you need to factor scalability into your evaluation plan. Performance also becomes an issue when you need to monitor systems across slow WANs.

Make sure the solution supports the alert methods that your staff requires, be it pager, email, SNMP traps, or execution of a script. Check out the reporting mechanism. I've never seen a solution that offers all the reports you might ever need, so how sophisticated is the user-definable report capability? What are your archival needs?

Does the solution's architecture fit your environment? Some solutions require you to deploy agents on each monitored system. Agents provide many advantages but also drive up implementation costs and can create problems for server administrators.

Is interoperability (such as support for syslog) important to you? Does the tool need to accept syslog data streams as input? Do you need to be able to send Windows security events to a syslog server?

Don't forget the issue of separation of duty. Do you have a large IT staff that includes separate staff to monitor security? If so, is the solution part of a larger operations framework under the control of the very folks you are supposed to monitor?

Finally, ask yourself whether the solution provides integrity and confidentiality of log data as it traverses your network. What about in the database and archive files?

Keep Learning

As you spend time with the Security log, you'll be able to interpret more and more of its obscure codes as well as make inferences based on patterns you begin to recognize. The best way gain this skill is to perform the actions you want to track and then analyze the events that Windows logs in response to those activities.

That sequence of activities might be the opposite of what you expect. But after many years of analyzing Security logs, I've found it's better to determine what

you want to audit and then find it in the Security log rather than to try to understand and eliminate events. Don't treat the Security log like an exception list in which each item needs to be followed up. The Security log just wasn't designed that way—in fact, it wasn't really designed at all. Only in the past few years has Microsoft created a Windows product group team that owns the Windows audit function. Prior to that, each team basically got a range of event IDs and used them as they saw fit or as dictated by the requirements of Common Criteria. Too much noise exists in the Security log, and too many events can be explained only after a lot of experimentation and conversations with a Microsoft support engineer.

If you choose to ignore the advice in the preceding paragraph, you'll definitely learn a lot about the Security log and discover some of its more-arcane secrets. More than once, I've discovered a new and useful event ID by querying the log for every unique event ID. Other times, I've used this method on a particular description field when I wanted to learn its full set of potential values.

The Security log has plenty to offer those willing to learn and experiment. Researching the log carries an added bonus: The more you learn about the Security log, the more you will understand about the security infrastructure of the largest and most widely used operating system in the world.

The Windows Server Security Log Revealed

Exclusive Discount

10% Discount

On Public, in-person or multi-media eTraining from
www.UltimateWindowsSecurity.com

Use or mention coupon code TWSSLR



MONTEREY TECHNOLOGY GROUP, INC.

