

Introducing Group Policy

As far back as 1996, top developers and architects from Microsoft's Windows NT engineering team traveled around to their large enterprise customers, asking what improvements they wanted most from the next version of NT. At that time, NT 4.0 was out, and Microsoft was planning the next major release. One common theme that was delivered loud and clear from customers was the need to reduce the cost of managing and maintaining Windows NT desktops and servers in large enterprise environments.

The whole topic of *Total Cost of Ownership (TCO)*—presented most vocally by the Gartner Group in its discussions of the cost of maintaining desktops in a corporate environment—took on new significance during that time. Even to this day, TCO plays a major part in how organizations look at the return on their Windows infrastructure investment. Until NT 4.0, Microsoft had spent less time worrying about TCO and more time adding features and increasing the stability and reliability of its fledgling operating system (OS).

With the upcoming Windows 2000 (Win2K, originally called NT 5.0), Microsoft focused squarely on the TCO challenge. In NT 4.0 (and Windows 95), Microsoft introduced *System Policy*, a mechanism for using the Registry to “lock down” portions of a user's desktop to prevent him or her from “futzing” with the configuration. It was believed that enough futzing would break something on the desktop, thus generating a costly call to the Help Desk. In the final analysis, futzing may turn out to be an overrated issue, but it nonetheless helped to increase the awareness of TCO in the enterprise.

System Policy was mildly effective, but it didn't completely address most enterprise TCO issues. For example, it hardly addressed software distribution, configuration management, or security configuration. So Microsoft got to work developing *Group Policy objects (GPOs)* in Win2K as the successor to System Policy. You can think of Group Policy objects as System Policy on steroids.



For simplicity's sake, I'll refer to GPOs throughout the rest of this chapter and the book using their commonly accepted acronym of GPO.

In this chapter, I'll introduce the capabilities of GPOs and take a detailed look at the features they provide, out-of-the-box. I say “out-of-the-box” because, in addition to delivering more functionality with the GPO infrastructure than System Policy, Microsoft also made the GPO technology extensible so that third parties could add additional management capabilities for their applications and services.



This chapter is written for readers who are already familiar with the concepts of Active Directory (AD) and Microsoft Management Console (MMC). If you need a refresher, check out <http://www.realtimepublishers.com> for references to other free eBooks on AD, including *The Definitive Guide to Windows 2000 Administration*, by Sean Daily and Darren Mar-Elia, and *The Definitive Guide to Active Directory Troubleshooting* by Sean Daily.

Overview of Group Policy

Group Policy is implemented on Win2K only. NT 4.0 doesn't support storing or processing GPOs. However, Win2K devices can process the old-style NT 4.0 System Policy (for example, ntconfig.pol) files when a user logs on to an NT 4.0 domain from a Win2K workstation or server. Group Policy can be defined on a local Win2K device or in an AD infrastructure. A local GPO exists on every Win2K workstation and server installation by default.

 The local GPO is stored by default in the folder %Systemroot%\System32\Grouppolicy.

The local GPO is a “stand-alone” object in that you must manage it individually against each Win2K device using the Microsoft Management Console (MMC) Group Policy snap-in. Also, some of the capabilities that GPOs provide in an Active Directory (AD)–based domain environment aren't available on local GPOs. (Examples are software deployment and Folder Redirection, described later in this chapter in “Windows Settings | Folder Redirection.”) For these and other reasons, GPOs only begin to fulfill their full promise when you've deployed an AD infrastructure and have begun to migrate all of your workstations and servers to Win2K.

Before talking about how GPOs leverage AD, and AD domain controllers, it's important to understand how GPOs are processed.

Processing GPOs

In NT 4.0, when a user logged in to an NT 4.0 domain from an NT 4.0 device, the Winlogon process running on the workstation or server executed a System Policy file stored on the authenticating NT 4.0 domain controller in the NETLOGON share (or elsewhere, as specified by the administrator). NT 4.0 System Policy let administrators distribute changes to Registry keys—per-user changes were stored in HKEY_CURRENT_USER keys, per-machine changes in HKEY_LOCAL_MACHINE keys. These Registry changes were executed according to the instructions of the System Policy file when the user logged on. Both per-user and per-machine changes happened at once.

Server-Side Processing

GPOs maintain a few similarities with System Policy processing. Only user objects and computer objects in an AD domain can process GPOs. However, unlike NT 4.0 System Policy—where only one policy file could be processed by a user per domain—multiple GPOs can be “linked” to AD *container objects* at various levels in the AD hierarchy, providing support for processing many GPOs for a given user and computer.

In addition to storing links in the AD itself, GPOs leverage file storage on AD domain controllers like NT 4.0 System Policy leveraged the NETLOGON share on NT 4.0 domain controllers. GPOs are stored in the SYSVOL share on domain controllers in an AD domain.

GPOs are made up of the *Group Policy template* (GPT) and the *Group Policy container* (GPC). The GPT stores the files that are stored by GPOs in SYSVOL. These files make up the guts of the GPO. (A portion of the GPO is stored in the AD itself.)

The GPC is created for each GPO you define. The GPC is a class of AD object that represents a GPO in the AD and keeps some additional information related to the software deployment features of GPOs.

Sites, domains, and OUs link to GPOs using an AD attribute on these container objects that refers to the linked GPO by its Globally Unique ID (GUID). More details of what is actually stored in the GPT and GPC, as well as how they relate to GPO functionality, are discussed in detail in Chapter 2 of this book.



Container objects are AD objects that can contain other AD objects. For example, an AD domain is a container object that can contain OUs, users, and so on. An Organizational Unit (OU) is a container object that can contain other OUs, users, groups, and so on. Finally, an AD site is a container object that can contain subnet objects.

Unlike NT 4.0's System Policy, which was processed when the user logged on, GPO-based policy directed at computers is processed by those computers that are members of an AD domain when they first start up and establish communication with the domain controller. This computer-specific policy processing happens independently of any user-based policy processing. Then, when a user defined in the AD logs on to an AD domain from a Win2K workstation (or server), any user-specific GPO policy is processed at that time.

This split between computer and user processing implies that there are two parts to a GPO, and that is indeed the case. As you'll see further on in this chapter, GPOs are organized in the GPO MMC snap-in into Computer Configuration and User Configuration sections. The section to which a policy is defined determines whether the computer or the user will process it. (In other words, computers process the Computer Configuration section, and users process the User Configuration section.)

Using LSDOU Ordering

In addition to delineating computer and user policies in a GPO, users and computers process GPOs using the following order of precedence:

1. Local GPO
2. AD site-based GPO(s)
3. AD domain-based GPO(s)
4. AD OU-based GPO(s)

You often see this order represented by the acronym LSDOU (Local, Site, Domain, OU). This means that computers and users process GPOs using LSDOU based on where those user and computer objects reside in the AD. LSDOU specifies an order of precedence in processing GPOs: local GPOs are processed first, followed by site, domain, and OU-based GPOs. The effective policy that a user or computer receives is often referred to as *Resultant Set of Policy*, or *RSoP*. I'll talk more about RSoP in Chapter 2 of this book.

To further illustrate LSDOU, Figure 1.1 shows an example of an AD domain with several GPOs defined.

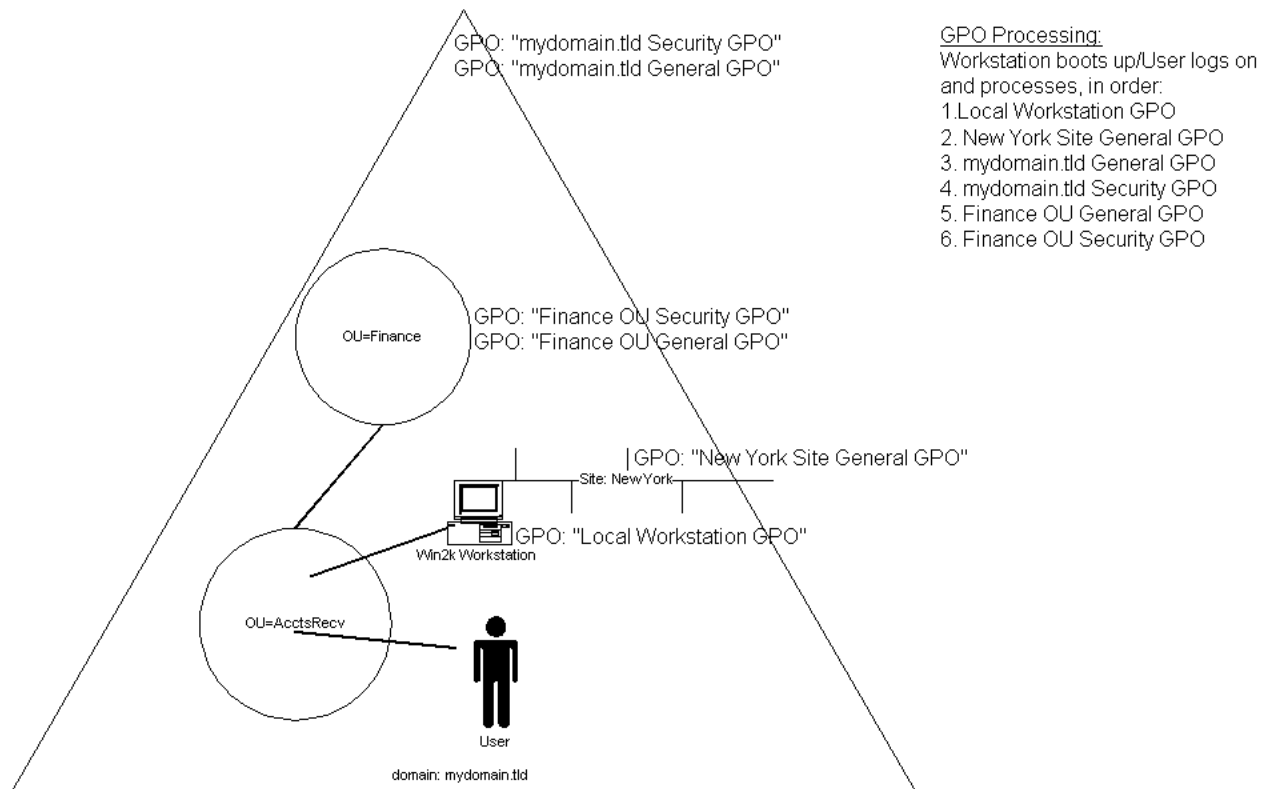


Figure 1.1: Processing GPOs using LSDOU ordering.

A lot of Figure 1.1 describes how GPOs are processed in an AD domain environment. First, note that the name of this particular AD domain is mydomain.tld. This domain contains an OU called Finance and a “child” OU of Finance called AcctsRecv. In AcctsRecv, there is a Win2K workstation object and a user object. The workstation object is on a subnet defined in the New York Site.



AD sites aren't associated with a particular domain. Instead, they create logical boundaries of physical Internet Protocol (IP) network subnets that can encompass devices from multiple domains. Thus, a site-based GPO can be processed by users and computers in multiple AD domains.

In Figure 1.1, six GPOs are defined:

- The local GPO on the workstation
- One on the New York Site
- Two on the Finance OU
- Two at the domain level

When the workstation starts up or the user logs on to the domain, those six GPOs are processed in the order shown in the text to the right of the drawing, reflecting the LSDOU ordering. There are a few things to note about this figure. First, in the case of the GPOs linked to the Finance

OU, even though the workstation and user are defined in the AcctsRecv OU, they still process the GPOs linked to the “parent” OU—Finance—thanks to AD inheritance.

Also, two GPOs are defined at each of the OU and domain levels. As you’ll see in Chapter 4, there is no problem linking multiple GPOs to a given container, but you must define an order of precedence for processing so that one GPO processes before the other. In this figure, the second GPO in the list is processed first, followed by the first one in the list for each container.

Finally, the names I chose for the GPOs are meant to help an administrator understand the purpose of each GPO. I’ll talk more about naming conventions for GPOs in Chapter 4, but when you define your GPOs, it’s definitely a good idea to descriptively and consistently name them.

Client-Side Processing

In addition to knowing about the ordering of GPO processing, it’s important to understand the mechanism on Win2K devices by which GPOs are processed. While in an AD infrastructure, the “guts” of a GPO are stored in the GPT (in SYSVOL) and the GPC (in AD itself), it’s the client (the Win2K server or workstation) that actually processes the GPO. The client uses a series of *client-side extensions (CSEs)*—Windows Dynamic Link Libraries (DLLs)—that are knowledgeable about the stuff stored on the server side of the GPO (the GPT and GPC).

As in NT 4.0, Winlogon calls these CSEs to process machine- and user-specific GPO settings. Each major default capability in the Win2K GPO has a corresponding CSE that ships with Win2K. The extensible model that GPOs provide lets Independent Software Vendors (ISVs) write their own CSEs to support custom GPO capabilities. You can think of the CSE as providing the real policy functionality in the GPO, while the GPT and GPC are the storage mechanism for whatever data that policy functionality requires.

For the list of CSEs registered on a given Win2K device, check the Registry key HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Gpextensions. Figure 1.2 shows a typical CSE registration.

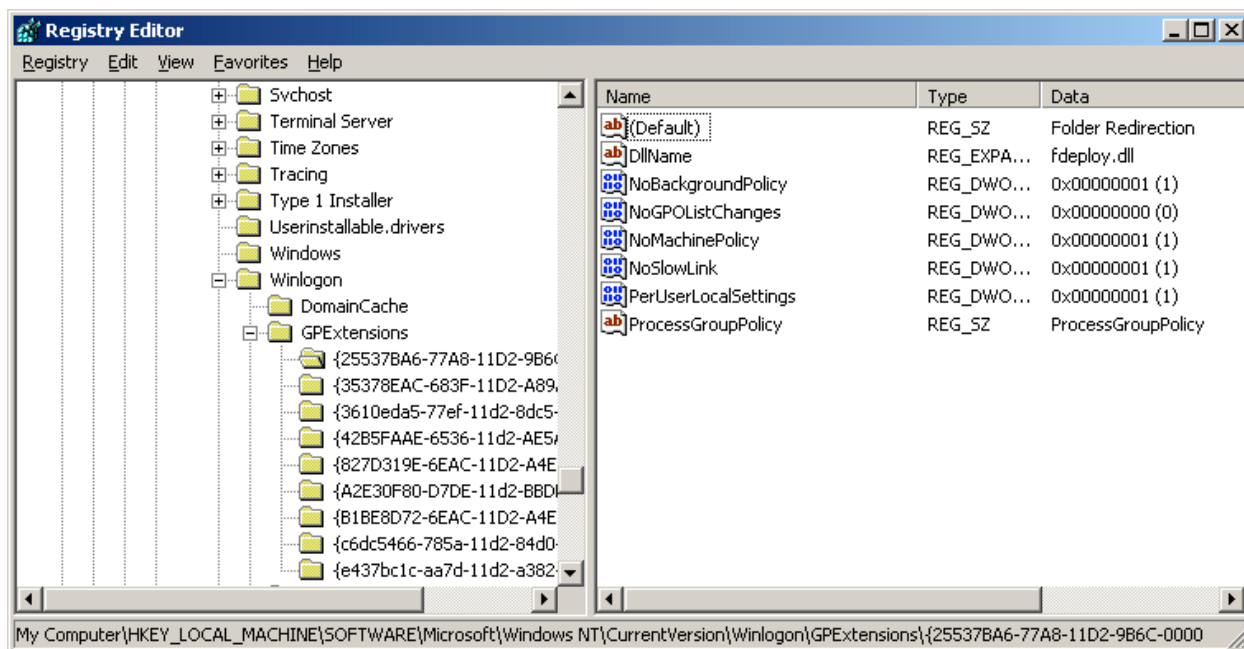


Figure 1.2: Viewing a client-side extension registration in the Registry.

In Figure 1.2, the Folder Redirection capability is provided by a CSE DLL called fdeploy.dll.

 See Chapter 2 for more details on the role of CSEs in Group Policy processing.

Viewing and Editing GPOs

Before I discuss in detail the capabilities of a GPO, it's a good idea to review how to view and edit them. The basic tool for managing GPOs is the MMC Group Policy snap-in. This snap-in is installed as part of every Win2K installation.

Microsoft has also pre-created an MMC console tool so that you can edit local GPOs on Win2K workstations and servers. This console tool is called gpedit.msc. Choose Start>Run, then type gpedit.exe to launch MMC with the Group Policy snap-in loaded and focused on the local GPO from where it's being run, as shown in Figure 1.3.

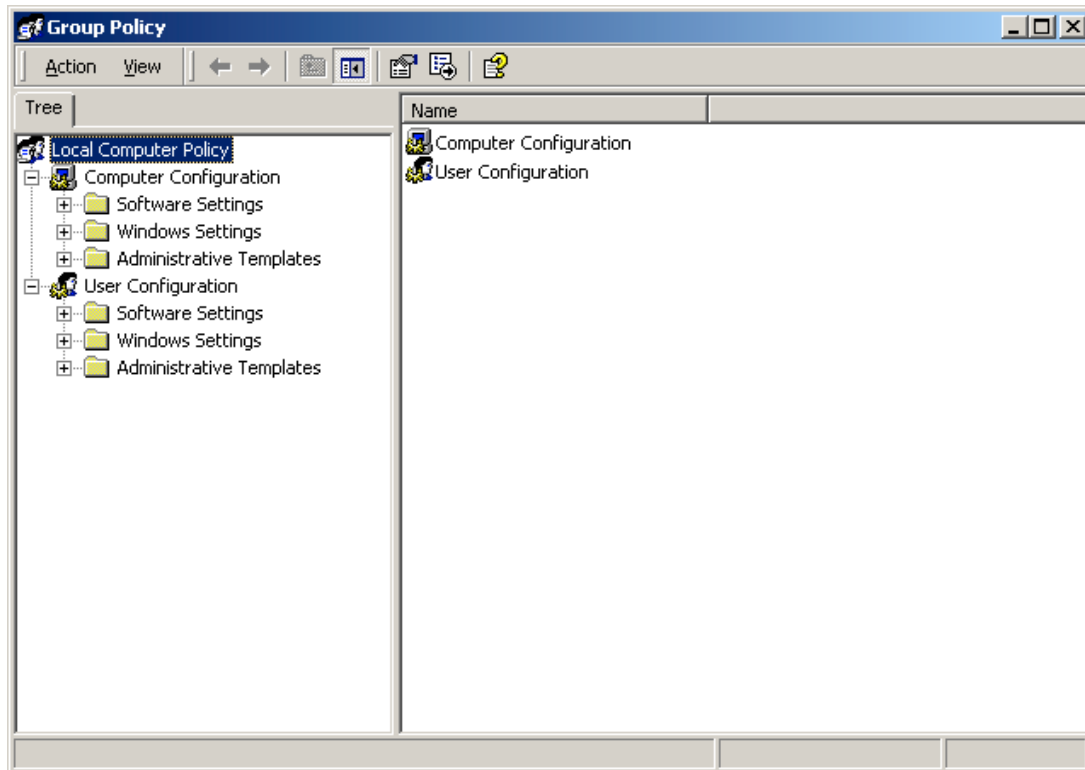


Figure 1.3: Viewing the Group Policy snap-in console tool gpedit.msc.

 In this section, I discuss using MMC and MMC console tools to create custom GPO tools. If you need a refresher on using MMC and creating MMC console tools, check out the MMC overview on Microsoft's MS Developers Network Web site at: http://msdn.microsoft.com/library/psdk/mmc/mmcint00_62pj.htm. In addition, check RealTimePublishers' Web site <http://www.realtimepublishers.com> for other free online titles that discuss using MMC.

In Figure 1.3, you can see two distinct “sections” to this local GPO. The Computer Configuration section holds policy definitions for computer-specific settings. As I mentioned earlier in this chapter, computer-specific policy is processed by Win2K workstations and servers when they start up.

 While it's true that most computer-specific policies only run when a workstation or server starts up, some policy items, such as certain security settings, are processed periodically to ensure that the policy is enforced without regard to a computer restarting.

The User Configuration section holds policy definitions for user-specific policies, and it's typically processed when the user logs on (or logs off, in the case of logoff script policy). In the next section of this chapter (“Capabilities of GPOs”), I'll drill into these two sections to further explore their policy capabilities.

Gpedit.msc is a pre-defined MMC console tool that lets you view and edit a local GPO. The actual MMC snap-in file that provides GPO editing functionality is called gpedit.dll and is located in %Systemroot%\System32. Gpedit.msc is locked down using MMC authoring features, so it only provides scope over the local GPO.

A more likely scenario is that you'll need to edit either GPOs defined on an AD domain or local GPOs defined on a remote workstation or server. In this case, you'll need to create your own MMC console tool. Choose Start>Run, then type mmc.exe. Once the blank MMC console appears, select Console, Add/Remove Snap-in, then click Add. A list of available snap-ins will appear, as in Figure 1.4.

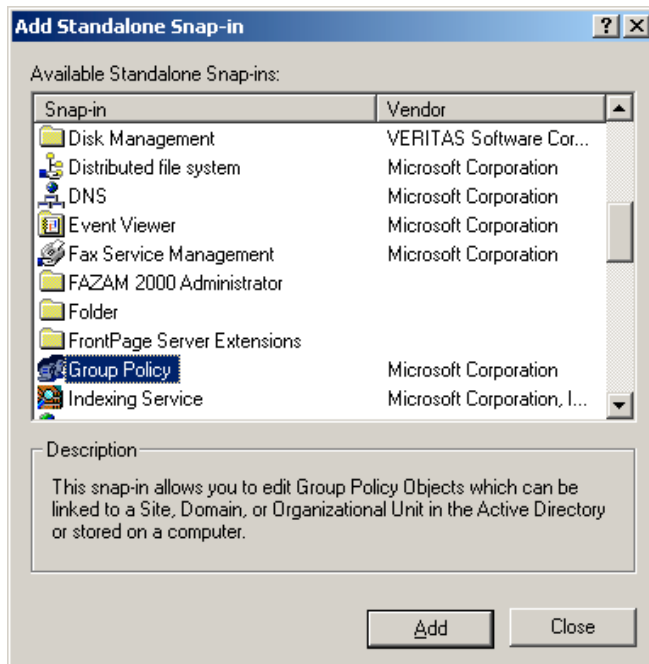


Figure 1.4: Viewing available MMC snap-ins on a Win2K device.

Scroll down to the Group Policy snap-in, select it, then click Add (or double-click Group Policy). The dialog box shown in Figure 1.5 will appear and let you choose the particular GPO that you want to manage (local or AD-based).

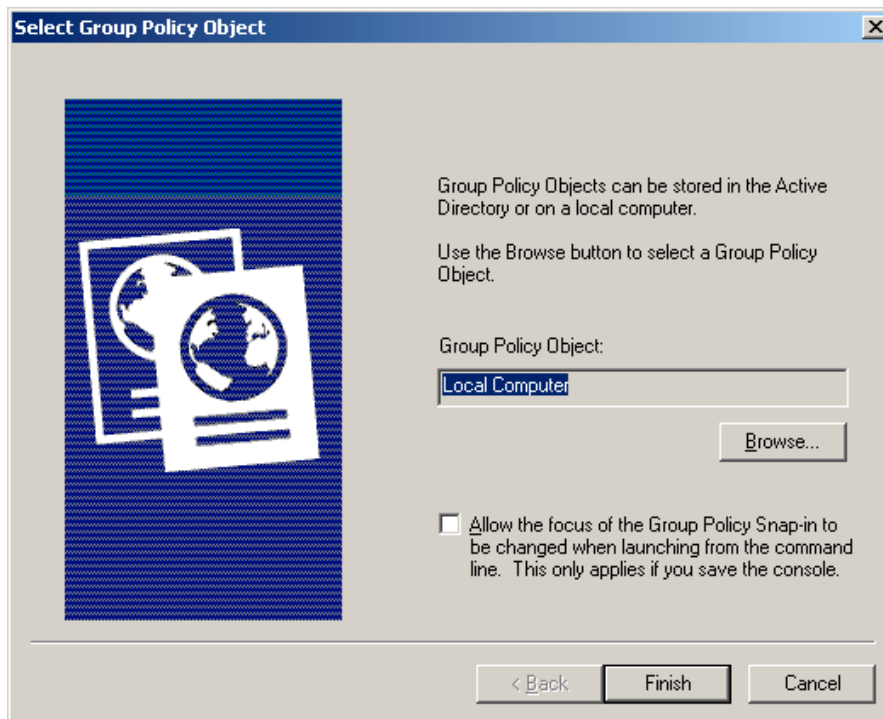


Figure 1.5: Choosing a GPO to load in the MMC console.

Click Browse to choose a GPO located on a local computer; in an AD domain, site, or OU; or all objects available in the current AD domain.

As you can see in Figure 1.5, the Group Policy snap-in provides a check box that allows you to change the focus of the GPO tool from the command line. The reason is that if you check this box, then save your MMC tool to an .msc file, you can change which GPO the tool focuses on when you launch MMC from the command-line. (However, in my testing, this feature doesn't work at all—launching an MMC tool from the command-line and providing the name of a different GPO to focus on is simply ignored.)

Once you've chosen a GPO to focus on and loaded the Group Policy snap-in, you can either stop there or add additional GPO references to this MMC tool. In fact, you can create custom MMC tools to meet a variety of GPO editing needs. For example, I like to create an MMC tool that has all of the GPOs I typically need to view and edit in my infrastructure, loaded in a single console for easy access. Figure 1.6 shows an example of such a tool.

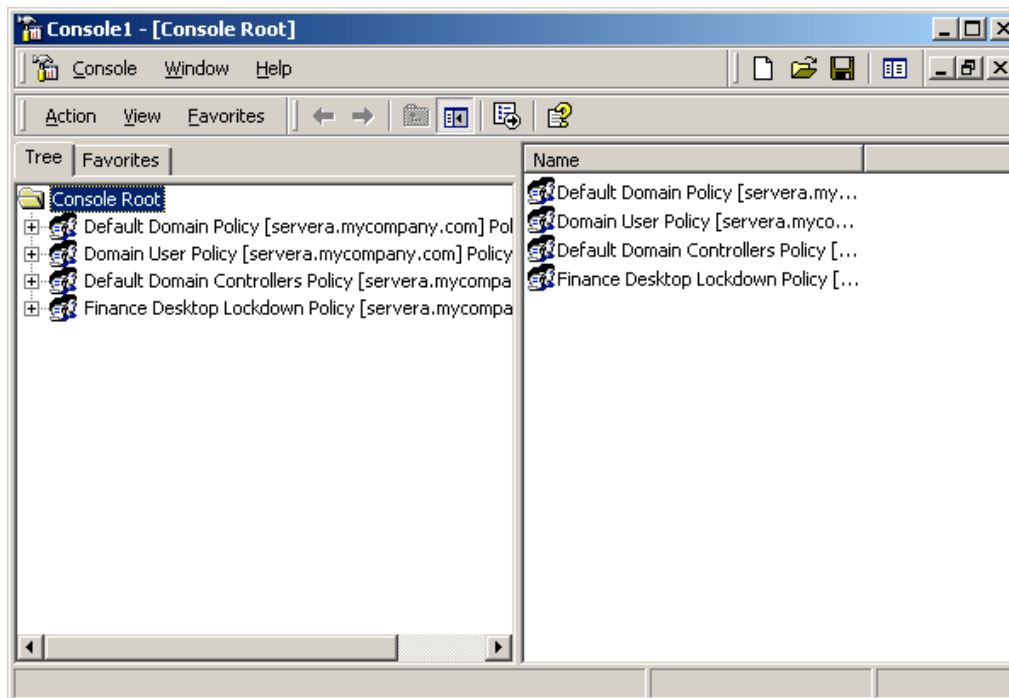


Figure 1.6: Viewing an MMC console tool that references multiple GPOs.

In this figure, I've loaded into one console four GPOs (and four instances of the Group Policy snap-in) that I frequently use, making it easy for me to get to them and edit them in one fell swoop. Similarly, if you have GPOs linked to OUs that are administered by different administrators in your enterprise, you can create MMC console tools with just those GPO references loaded, then distribute those console tools to your OU administrators.

These console tools provide you with some level of control over who is viewing and editing GPOs in your AD infrastructure. Of course, you can still delegate AD access rights on a particular GPO, and this allows you to control who can access and edit a GPO. But the MMC console-tool approach is quick and easy.

Once you've loaded the references to the GPOs you're interested in managing, choose Console from the MMC menu and Save (or Save As) to give a name to your .msc file. I like to name my GPO console files descriptively to help distinguish their function (for example, Domain-Wide GPOs or Finance OU GPOs).

The GPO snap-in organizes GPO capabilities into *nodes*. You can expand these nodes to drill into each capability. For example, if I expand the Computer Configuration node of the local GPO shown in Figure 1.3, I see a number of different categories of functionality—Software Settings, Windows Settings, and Administrative Templates. The same is true under User Configuration. In each of these categories are the nodes that describe the individual policy functionality provided by this GPO. I'll talk more about this functionality in the next section, but let's drill into one in particular to see how you can edit a GPO.

In Figure 1.7, I've drilled into the Windows Settings | Security Settings node under Computer Configuration in my local GPO.

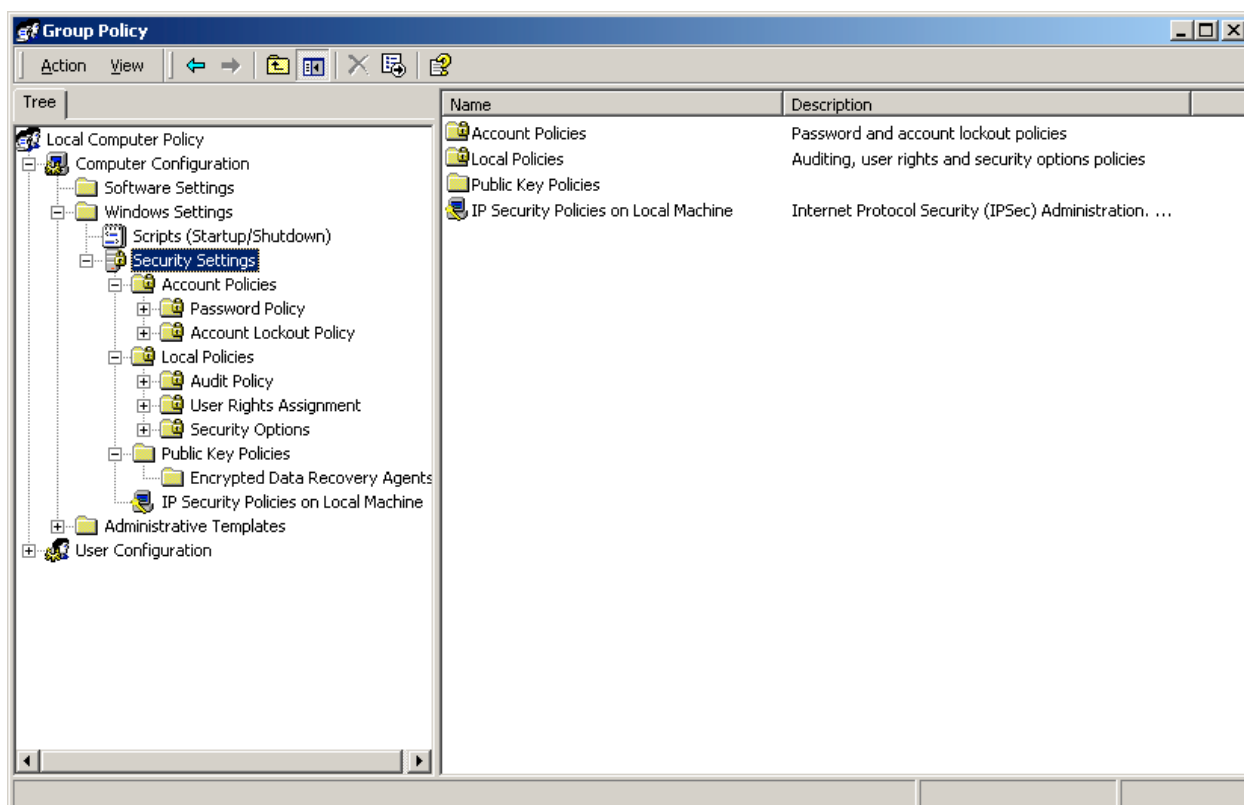


Figure 1.7: Viewing the expanded Security Settings node in a local GPO.

In this figure, you can see that the Security Settings node is where I can set a variety of security policy in this local GPO. Indeed, on a local workstation or server that isn't part of an AD domain, this node provides the equivalent functionality to NT 4.0's User Manager\Policy menu items.

Within Security Settings, there are three subcategories—Account Policies, Local Policies, and Public Key Policies. If you drill into each of these, you see additional capabilities. For example, under Account Policies, you see the Password Policy, where you can set restrictions such as

minimum password length and age. Figure 1.8 shows where I've drilled into Password Policy to expose the actual policy items available.

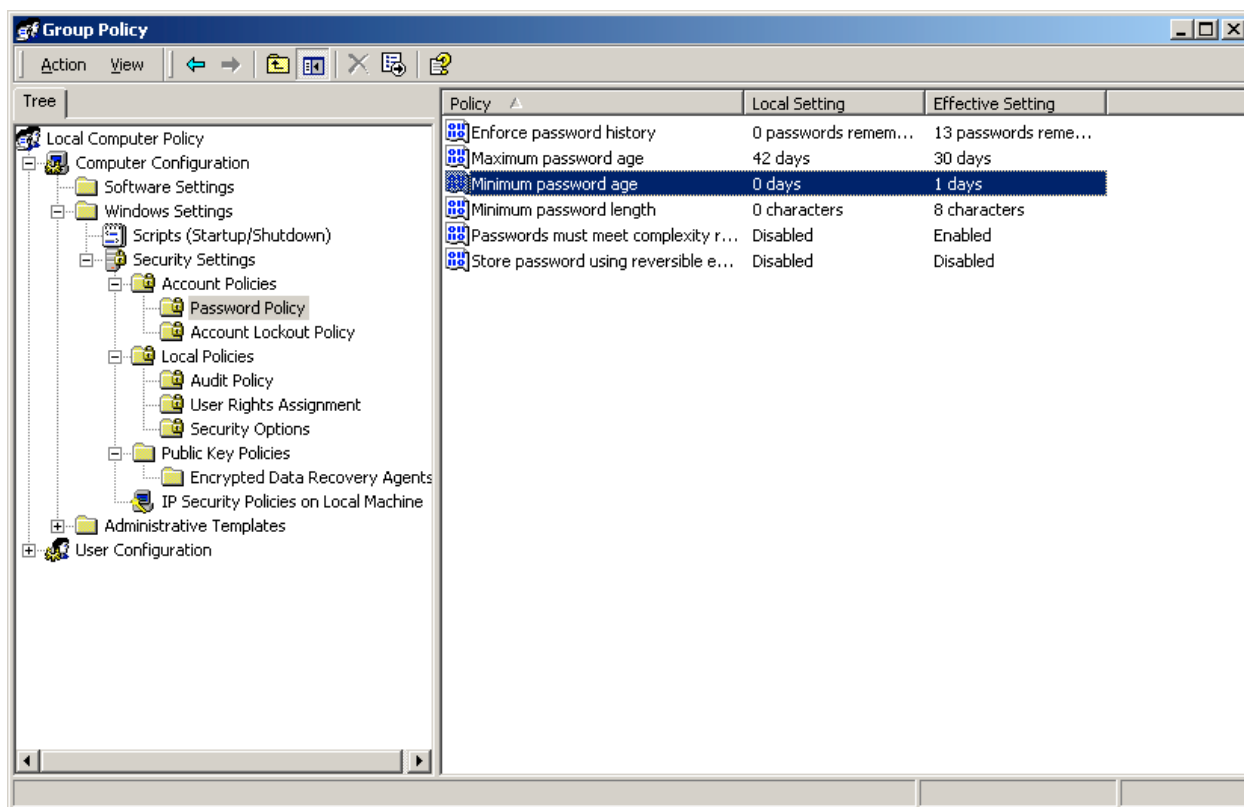


Figure 1.8: Viewing the policy options under the Password Policy section of a local GPO.

This figure shows the actual policy items in the results pane in the MMC snap-in. If you double-click one of the items, such as Minimum Password Age, you can set the policy in this GPO.

When you set a policy item in a GPO, there is no concept of saving or undoing it. Once you make a change, the change is committed unless you change it to another value. For example, if you set Minimum Password Age to 10 days, then click OK, the policy is set. You don't need to save the GPO (in fact, there is no place in the MMC snap-in to save it). Likewise, you can't simply click an Undo button to back out of the change. You need to go back into the Minimum Password Age dialog box and reset the value to 0.

This issue underscores an important point when you're editing GPOs: Make sure you document the changes you're making. Unfortunately, there is no good automated way to manage GPO changes in Win2K, but I suspect that third party software companies will develop products that solve this problem in the near future. In the meantime, one tip that you might find useful is to disable the GPO before you edit it. This will prevent users who may be logging in or restarting their computers from processing your GPO while you're in the middle of changing it.

In Chapter 3, I describe how you can disable a GPO before you edit it (or for other reasons).

In Figure 1,8, you see two columns in the results pane of the MMC console tool—Local Setting and Effective Setting. This distinction is important to understand if you’re working in an AD environment with AD-based GPOs.

As I mentioned, every Win2K device comes with a local GPO, which is what Figure 1.8 is focused on. The Local Setting column shows the policy that has been set for that item in the local GPO. The Effective Setting column shows what the current policy is actually set to. This may be different than the Local Setting value in an AD environment because AD-based GPOs are processed after the local GPO (remember LSDOU) and therefore can override settings made in a local GPO. Of course, for Win2K devices that aren’t members of a domain, Local Setting and Effective Setting values should be identical.

 The distinction between Local Setting and Effective Setting is a function of the Security Settings node. Other policy nodes in a GPO may or may not present this information.

Keep in mind that each GPO node has its own requirements for enabling or disabling policy. For example, in the Security Settings section, you may be required to provide an absolute numeric value like number of days between password changes. However, in other areas, it may be a matter of simply enabling or disabling policy. Still other areas, like logon scripts, may require you to enter the path to a file or folder. The important thing to remember is that there is no one standard way of manipulating GPOs.

Capabilities of GPOs

Now that I’ve introduced the basic concepts of GPOs and how you can view and edit them, I’ll describe the policy capabilities that are available out-of-the-box for Win2K GPOs.

The capabilities of GPOs are formidable and complex. In this section (and this entire book), I’ll discuss all that GPOs are capable of and help you use them in your own enterprise to your best advantage. While the GPO infrastructure is extensible, allowing developers of third-party applications to develop their own CSEs and policy features, this section describes what Microsoft has developed in Win2K and the major features of each policy area.

 Chapter 3 discusses the hands-on use of each of these GPO capabilities.

Table 1.1 shows a brief overview of the default GPO capabilities available.

This Capability (Computer Configuration)	Does This
Software Settings Software Installation	Lets you assign Windows Installer–based application installation packages to AD machine objects.
Windows Settings Security Settings	Provides machine-specific security configuration using security configuration templates.
Windows Settings Scripts (Startup/Shutdown)	Machine-specific startup or shutdown scripts. These can be batch files, executables, or Windows Script Host (WSH) scripts.
Administrative Templates	Machine-specific Administrative Templates let you control Registry

	settings in the HKEY_LOCAL_MACHINE Registry hive. Similar to NT 4.0 System Policy.
This Capability (User Configuration)	Does This
Software Settings Software Installation	Lets you assign Windows Installer–based application installation packages to AD user objects.
Windows Settings Internet Explorer Maintenance	Lets you manage IE configurations for each user. Transfers the capabilities of the IE Admin Kit to GPOs.
Windows Settings Folder Redirection	Redirects Windows Explorer folders like the Desktop, My Documents, and Startup to server-based folders rather than the user's profile.
Windows Settings Security Settings	Enforces public key infrastructure (PKI) policy for AD user objects.
Windows Settings Remote Installation Services	Defines how the Remote Installation Service (RIS) works on a per-user basis.
Windows Settings Scripts (Logon/Logoff)	User-specific logon and logoff scripts. Similar to startup and shutdown scripts, they support any number of scripting environments.
Administrative Templates	User-specific Administrative Templates let you control Registry settings in the HKEY_CURRENT_USER Registry hive. Similar to NT 4.0 System Policy.

Table 1.1: The default capabilities of GPOs.

I discuss each of these capabilities in the sections that follow.

Software Settings | Software Installation

Software Installation—both user- and computer-specific—is a key feature in Microsoft's GPO infrastructure. You can think of Software Installation as a mid-range software-distribution capability, similar to what Microsoft provides with the Systems Management Server (SMS) for larger enterprises. Software Installation differs from products like SMS in several important ways, however. It's missing many of the features that large enterprises look for in a robust software distribution product, such as:

- Ability to capture hardware and software inventory (asset management)
- Robust, centralized reporting on the success and failure of software distributions
- The ability to distribute software to servers in an unattended way (SI is really designed for desktop-based deployments)
- Solid, centralized troubleshooting tools

Interestingly, not many capabilities of SMS 2.0 have been integrated into Software Installation. Despite that fact, the Software Installation provides a good software-distribution capability for small- to medium-sized enterprises. It lets you *assign* applications on a per-computer basis and assign and *publish* applications on a per-user basis. I'll describe assignment and publishing in a

moment. But first, I want to talk a little bit more about Software Installation and the Windows Installer.

Software Installation leverages Microsoft's Windows Installer, which ships with Win2K to perform application installations. The Windows Installer is an application-packaging format, installation engine, and set of APIs, and it provides a better application setup than earlier technologies. Windows Installer packages have an .msi file extension. Applications like Office 2000 are good examples of software that leverages the .msi package format.

One of the big advantages of the .msi format and the Windows Installer engine is the ability to install applications using an elevated security context. This means that applications that normally require administrator access to install can be installed by regular users without an administrator's intervention and without the need to grant the users privileged access to their workstations.

When an application is deployed using Software Installation, it's called a *managed application*, and it can thus take advantage of the elevated security privileges in Windows Installer. Figure 1.9 shows an example of an application that has been deployed in Software Installation.

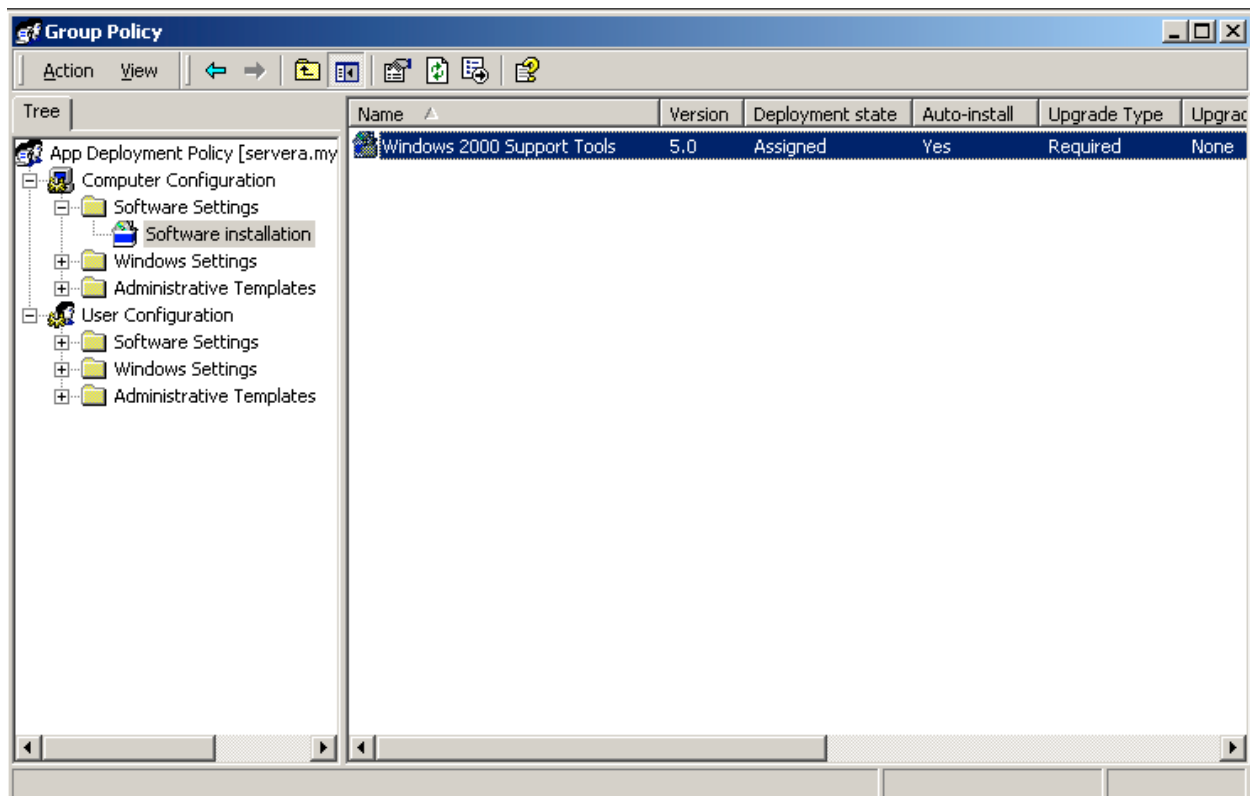


Figure 1.9: Defining an application deployment using GPO-based Software Installation.

In this figure, I've assigned the Windows 2000 Support Tools application to be deployed in this GPO called App Deployment Policy. This is a computer-specific deployment, so all computers subject to this GPO will have the application installed.

Assignment versus Publishing

Applications can be deployed using Software Installation to either computers or users. An application deployed to a Win2K computer is installed the next time that computer is restarted. Applications deployed to users can either be “partially installed” when the users log on (assignment), or installed on demand by the users themselves (publishing).

Applications that are assigned to a user are *advertised* to that user the next time the user logs on. Advertisement is a feature of Windows Installer packages (.msi files) that is tightly integrated into the Windows Explorer shell. Advertisement puts an application’s *presence* on the user’s desktop. A presence may be a shortcut to an application placed on the user’s Start menu, a file association to the application (for example, .doc files are associated with Microsoft Word), or even a Component Object Model (COM) component registration that refers to the application.

However, during advertisement, the application isn’t really installed. Rather, these presences are placed in the user’s environment. The first time a user activates one of these presences (for example, clicks a shortcut or opens a file with an advertised association), the Windows Installer engine starts up and installs the application from its setup package. This is known as “Install on Demand,” and it’s a powerful feature of GPO-based Software Installation deployments. It means that only those applications and application features that the user needs are installed when they’re needed.

Publishing takes a different approach from assignment in GPO-based Software Installation. Publishing an .msi-packaged application makes it available to the user in the Add/Remove Programs control panel that comes on every Win2K device. Figure 1.10 shows an example of an application that has been published in a GPO.

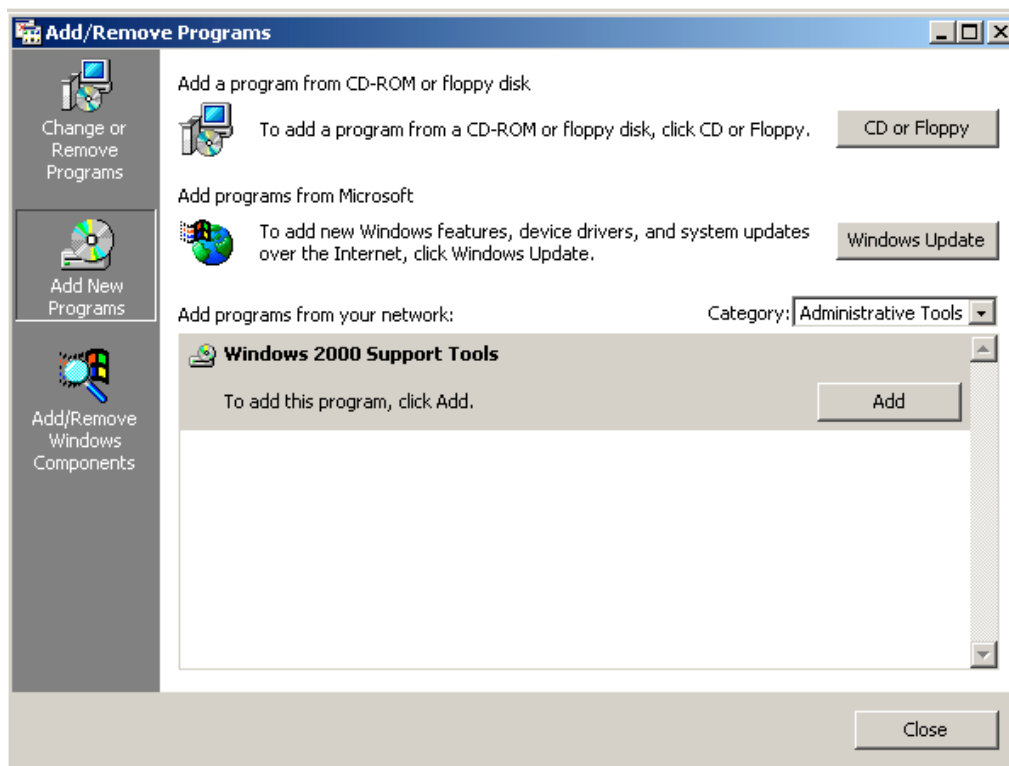


Figure 1.10: Using Software Installation publishing to make applications available in the Add/Remove Programs control panel.

In this figure, you see that the Windows 2000 Support Tools application has been published so that a user can install it by clicking the Add button. In GPO-based software installation, you can also categorize published applications to make it easier for your users to find and install them. Notice the Category drop-down list in Figure 1.10. You can define categories in the GPO Software Installation node to help delineate your applications in the Add/Remove Programs applet. For example, in Figure 1.10, I created a category called Administrative Tools, where I've placed this published application.

Another useful feature of Software Installation is managing the life cycle of applications. Managing a life cycle is upgrading, patching, rolling back, and uninstalling applications centrally, and what's more important, cleanly. Managing a life cycle in Software Installation depends on the Windows Installer technology to succeed. However, it's an important feature that more advanced products like SMS don't do, or do poorly.

In Software Installation, you can create update dependencies among applications. When a new or patched release of an application comes out, all those users or computers that previously had the application installed can be set to receive the new version. Additionally, if you're trying to retire an application, you can deploy it using Software Installation in such a way that if the application is no longer deployed using Group Policy, it'll be uninstalled. These life cycle management features are tremendously valuable for organizations that are deploying and uninstalling many applications over time.

Windows Settings | Security Settings

The Security Settings feature in GPOs may be one of the most confusing for many administrators. The ability to set many basic security policies in Win2K is now only done using Group Policy (either local or AD-based GPOs). This differs dramatically from NT 4.0, where features like account and audit policy were set from the User Manager, and more advanced security features like setting service and file system permissions could be managed using the Security Configuration Manager (SCM).

Microsoft folded the capabilities of SCM into Group Policy. In fact, Security Settings policy in a GPO can use the same security template files (file names usually ending in .inf) to set security policy settings that SCM (both the NT 4.0 and Win2K versions) uses. At the same time, Microsoft augmented the capabilities found in NT 4.0 SCM to provide a one-stop shop for setting all security policy in Win2K. Indeed, you can think of the Security Settings feature—both computer- and user-specific—as being an excellent centralized tool for managing security across all of your AD-based Win2K devices.

The confusing part about GPO-based Security Settings comes when you try to ensure that the right security policy is applied to the right set of computers and users. For example, machine-based security policy defined in GPOs that are processed by domain controllers affects policy for the entire domain. On the other hand, security policy defined in GPOs that are processed by member servers or workstations only affects the local Security Account Manager (SAM) and resources on those local computers.

I mentioned earlier that machine-based policy is typically processed when a computer restarts and user-based policy when a user logs on. However, the Security Settings policies are also processed on a periodic basis (by default, every 90 minutes).

☞ You can also manually force security policy to refresh by issuing the following command:

```
secedit /refreshpolicy machine_policy
```

or

```
secedit /refreshpolicy user_policy
```

The Security Settings capability provides a number security policy features, discussed in the following sections.

Account Policies

Account Policies are machine-specific policies. Account Policy defined for an entire AD domain can only be set in a domain-linked GPO. That is, if you have an AD domain controller that is subject to a GPO linked to an OU, and that GPO contains Account Policy settings, they'll be ignored in favor of the Account Policy defined in any domain-linked GPOs that are present.

Account Policy includes three sections—Password Policy, Account Lockout Policy, and Kerberos Policy. Password Policy, as Figure 1.11 shows, is where you set policy such as minimum password length, password age, and password complexity enforcement (rules that say that a password must contain numbers and capital letters, for example).

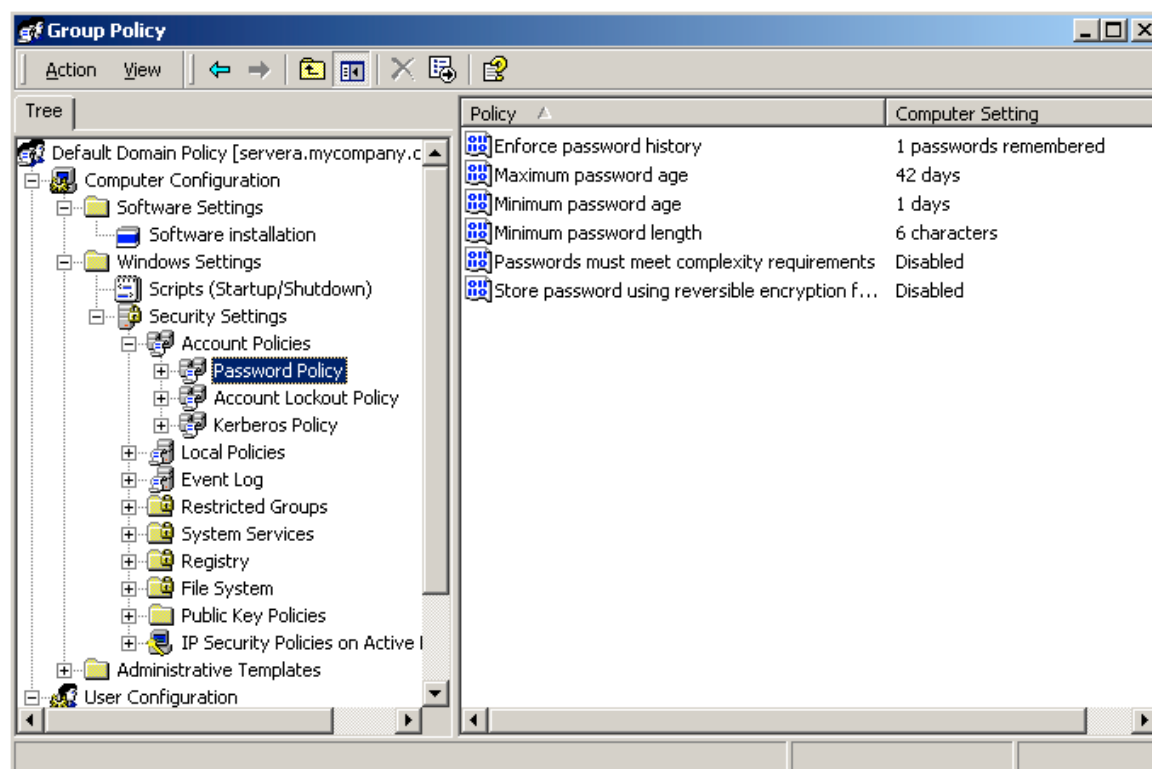


Figure 1.11: Viewing Password Policy options in GPO-based Security Settings.

Account Lockout Policy is where you set policy to determine whether and for how long a user account is locked out after a number of incorrect password attempts. Kerberos Policy is new in Win2K and supports the MIT Kerberos V implementation that Win2K uses as its default authentication protocol. Figure 1.12 shows the options available in Kerberos Policy.

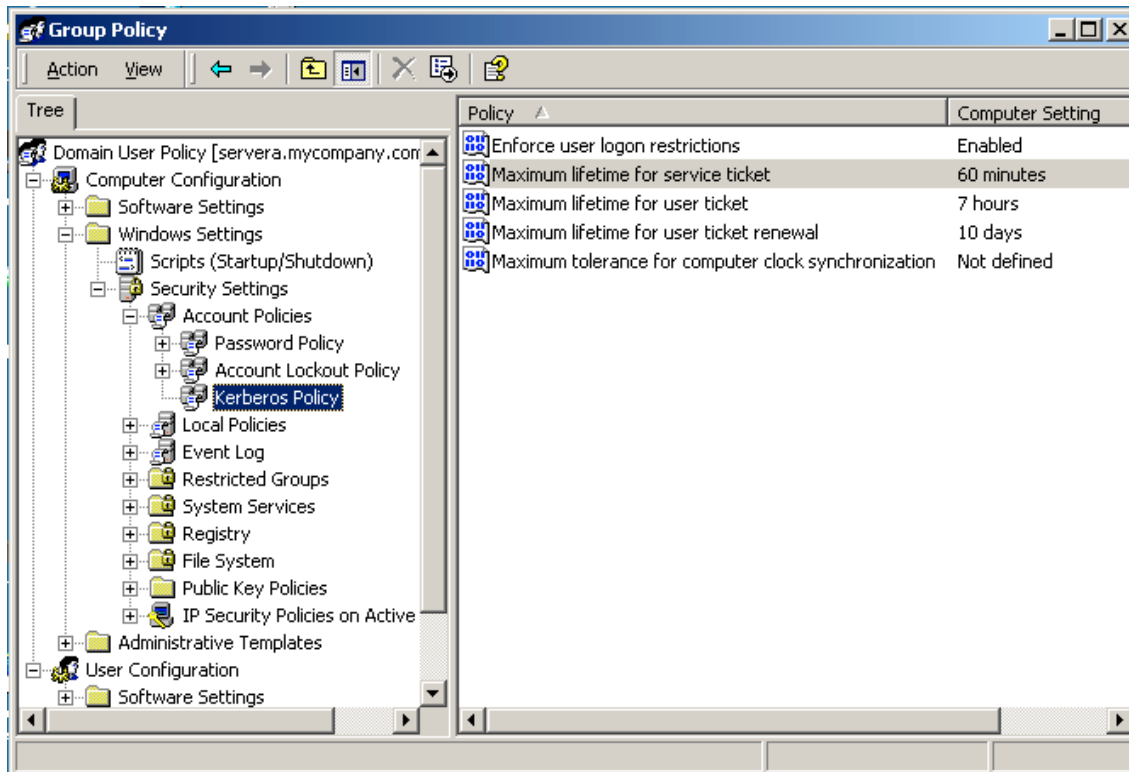


Figure 1.12: Viewing Kerberos Policy options in GPO-based Security Settings.

As the figure shows, Kerberos Policy lets you set features of the Kerberos protocol such as ticket lifetime and ticket renewal intervals.

Local Policies

Local Policies, as the name implies, lets you set security policy that is relevant to the local computer. Local Policy defined in a domain-linked GPO affects all computers in that domain (unless explicitly excluded), including domain controllers. Local Policy applied to a domain controller becomes the local policy for that domain. Local Policy applied to workstations or member servers is just specific to those devices. Local Policies includes three sections—Audit Policy, User Rights Assignment, and Security Options.

Audit Policy

Audit Policy is similar to the NT 4.0 User Manager's Policy Audit menu item. Figure 1.13 shows the options available in Win2K Audit Policy.

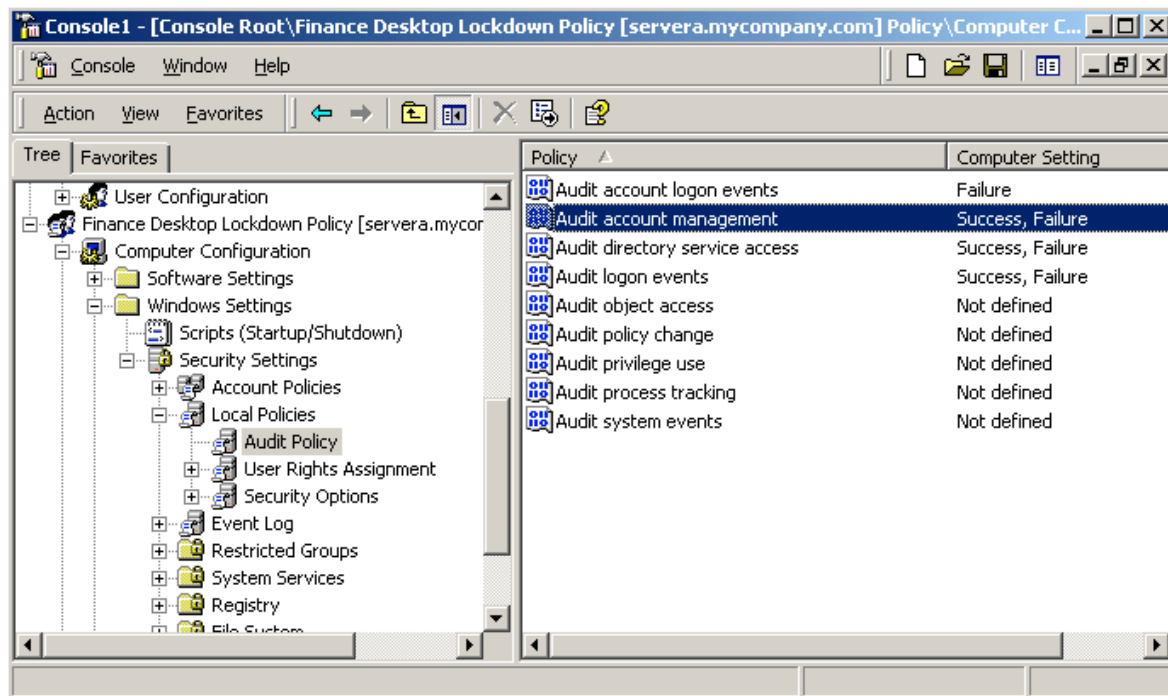


Figure 1.13: Viewing Audit Policy options in GPO-based Security Settings.

Audit Policy allows you to enable auditing events for various kinds of system access. Audit events are stored in the security event log on the Win2K system where they're generated. As with Account Policy, Audit Policy defined in a GPO that is processed by a domain controller will become the policy for that AD domain. Audit Policy defined on GPOs processed by workstations or member servers affects only those devices. Auditing success indicates that you want to raise an event log entry each time that particular type of event (for example, Audit Account Management) succeeds. Similarly, auditing failure means that you want to raise an event log entry when that type of event fails.

Auditing NT File System (NTFS) and Registry access is a two-part process. First you must enable the Audit Object Access event type. Then, in the file system or Registry itself, you need to enable auditing for a particular folder or key.

User Rights Assignment

User Rights Assignment is similar to NT 4.0 Users Manager's Policy User Rights menu item. It's where you define which users and groups have the right to perform certain kinds of operations. Typical user rights include the ability to shut down a computer, log on to the local console of the computer, and add computer accounts to a domain. Win2K adds quite a few rights above and beyond what NT 4.0 provided, so it's worth scanning through them to see if there are any that offer some useful additional protection. Again, User Rights Assignment defined on a GPO that is processed by domain controllers becomes the user rights policy for that domain.

Security Options

Security Options lets you set what I call the miscellaneous security lockdowns. Many of the security settings you see in this section previously required Registry tweaks to set or weren't available at all. Figure 1.14 shows many of the policies in this section.

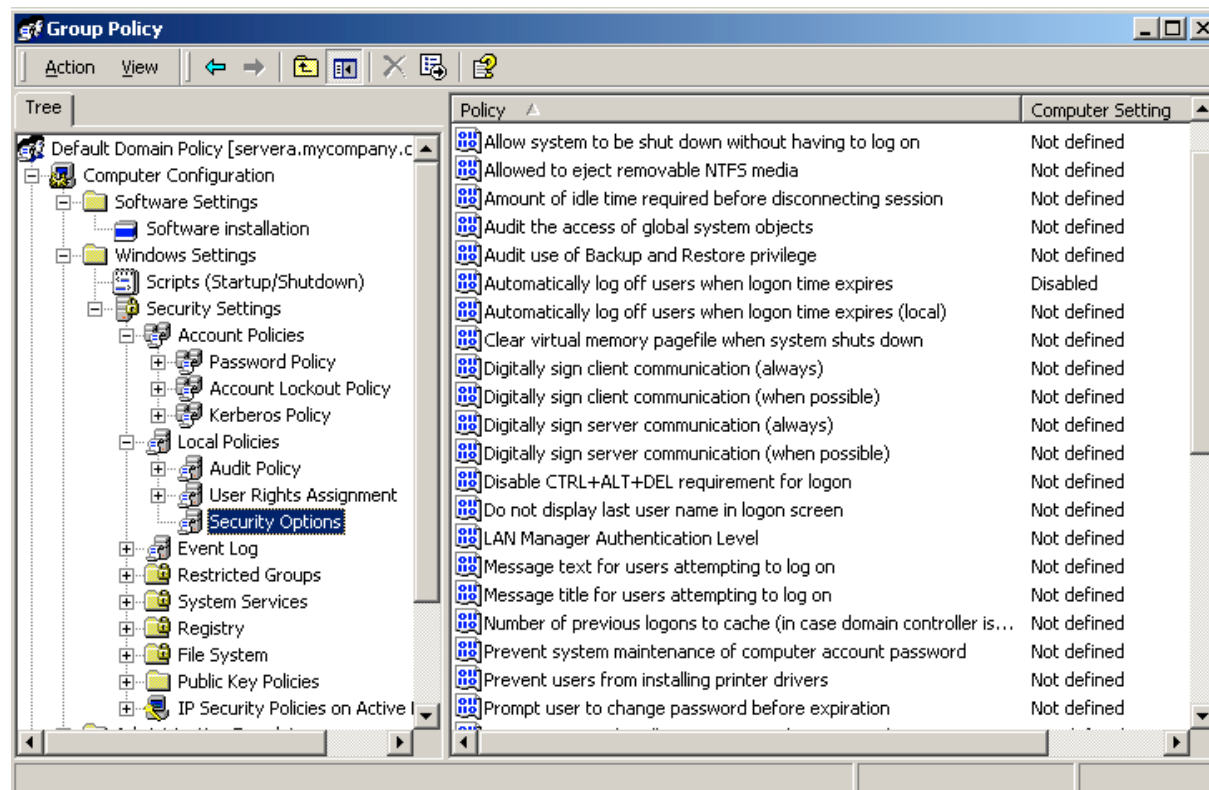


Figure 1.14: Viewing Security Options in GPO-based Security Settings.

As you can see in this figure, features such as “Message Text for Users Attempting to Log On” was typically something you had to set directly in the Registry (or using System Policy). You can think of the Security Options section as the dustbin of Win2K security policy. Everything that was left over when Microsoft created the other security policy sections was placed in here! However, it also contains some useful features, especially if you’re deploying Internet-facing Win2K systems, so it’s worth having a look.

Event Log

The Event Log policy section is another computer-specific security setting. This section lets you centrally define, in a GPO, the event log settings you want to enforce across computers in your domain (or locally on a computer, in the case of the local GPO). Figure 1.15 shows the options available in this section.

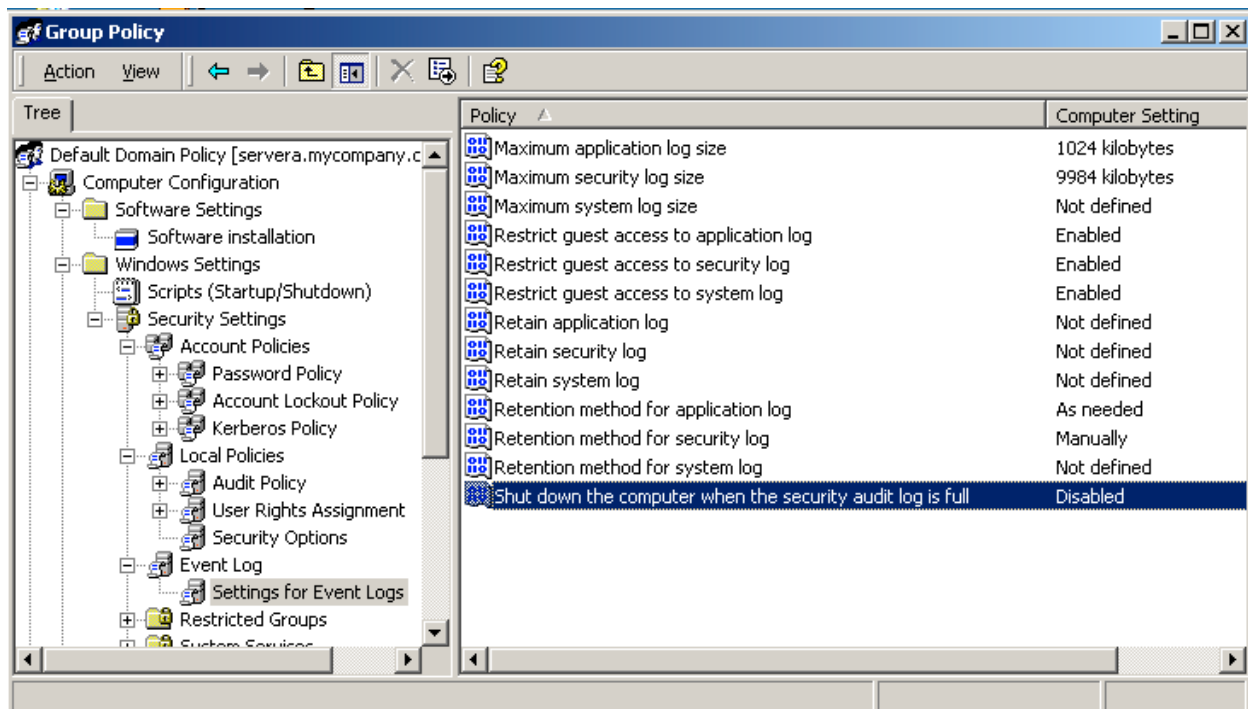


Figure 1.15: Viewing Event Log policy settings in GPO-based Security Settings.

To enforce some consistency of logging in NT 4.0, you had to set these options on every non-domain controller computer. Using Group Policy, you can control how big event log files can grow and what to do with them when they've reached their limit.

 Most of the security policy I've been discussing in this section lets you define a policy item with a particular value or leave it as Not Defined. Note that Not Defined isn't the same as disabled. That is, a particular security policy (such as Application Event Log Size) may come with a default value when you install Win2K. *Not defining* in a GPO simply means that the default value remains, not that (as in this example) application event logging has been disabled.

Restricted Groups

Restricted Groups is an interesting, if not complicated, security policy to implement in GPO. Restricted Groups allows you to define particular user groups who can only have a specific set of users as members. Similarly, you can define a Restricted Groups policy that says that a particular user group must always be a member of another group or groups.

In the former situation, you can use the Restricted Groups feature to say that only Joe Admin's account and the local Administrator account can be members of the local Administrators group on computers subject to the GPO policy. This means that if a user account called Bill Evil is able to compromise a computer and add itself to the local Administrators group, the next time a GPO containing Restricted Groups policy is processed by that computer, Bill Evil's account is automatically removed from the group.

Using the second part of the Restricted Groups feature, you can ensure that a particular user group cannot be removed from other groups. For example, suppose you want to ensure that the

user group Desktop Administrators, defined on your AD domain, is always a member of the local Administrators group. If you define this policy in Restricted Groups, any computers that process the GPO containing this policy will always enforce this restriction. If someone tries to remove your Desktop Administrators group from the local Administrators group on a workstation, the next time the GPO is applied, the policy will be enforced, and your group will be added back in.

As you can probably see, Restricted Groups is a powerful feature. Misusing or misunderstanding how and when it applies, especially when you're restricting group membership to a set of particular users, can cause you to lock yourself out of a particular workstation, a server, or potentially even an entire AD domain. Use Restricted Groups with caution, and remember that when you set a Restricted Group policy in a GPO, any computer that processes that GPO will enforce your policy.

System Services

In the System Services policy, you can define the default startup mode for a particular service. What's more important from a security perspective, you can define users and groups that have the right to start, stop, pause, and change the configuration of a particular service. This is a powerful feature when it comes time to delegate administration of Win2K resources on your network. You can set a System Services policy, for example, that lets you grant your Help Desk user group the right to start and stop the printer Spooler service only on a designated set of Win2K print servers that are subject to that GPO. Figure 1.16 shows an example of the dialog boxes available in the System Services policy.

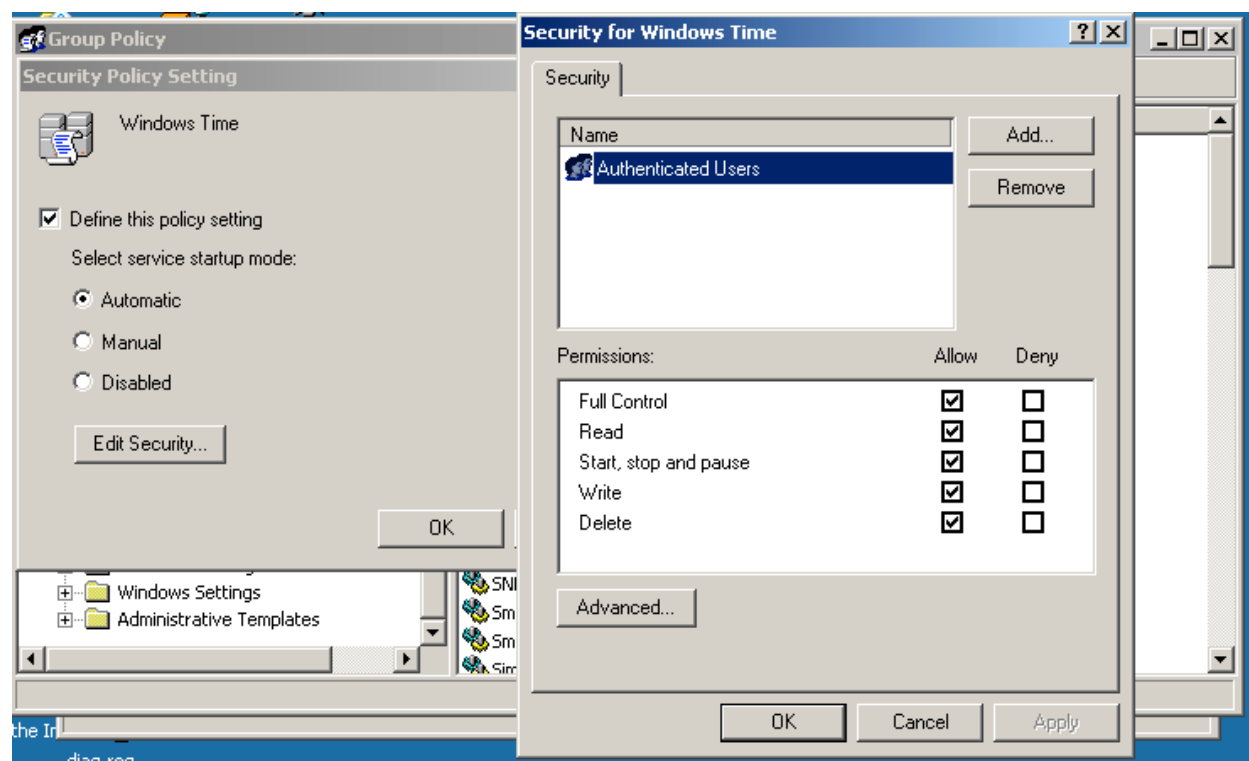


Figure 1.16: Setting the Windows service policy options under the System Services policy.

In this figure, I've set the default startup mode of the Windows Time service to Automatic, and I've granted the Authenticated Users group the ability to start, stop, and pause the service (as well as read, write, and delete the service's configuration information from the Registry). When a computer subject to this policy processes the GPO, these service settings will be enforced, and only Authenticated Users will be able to manage this particular service.

Keep in mind that the list of system services that appears in this policy section is somewhat a function of where you're running the MMC Group Policy snap-in. Ordinarily, the list of services is built using what is currently installed on the computer where you're editing the GPO. This may not be sufficient if you want to control services that only exist on certain types of Win2K devices. For example, you may have a third-party backup application that runs as a service on your servers, but not on the workstation where you're editing the GPO. In that case, you need to either open the Group Policy snap-in on that server and create your service security settings or manually edit the security template file that is being used by that GPO.

 For more information on security templates, see Chapter 3.

Registry

The Registry security policy section allows you to centrally enforce security permissions on Registry keys on computers that are subject to the policy. For example, for security reasons, you may want to lock down a particular Registry key that enables users to change some aspect of their workstation configuration. You can use the Registry security policy to enforce this lockdown.

You can also use Registry security policy to enforce a particular access control list (ACL) on a particular Registry key. For example, you may want to maintain a particular ACL on a key no matter who comes along with another GPO and tries to reset it. If you set this policy in a GPO that is processed high in the LSDOU order of precedence—for example, at the domain level—a subsequent OU-based GPO that tries to change the permissions on that key will be ignored. Figure 1.17 shows an example of the options available for propagating ACLs in the Registry policy.

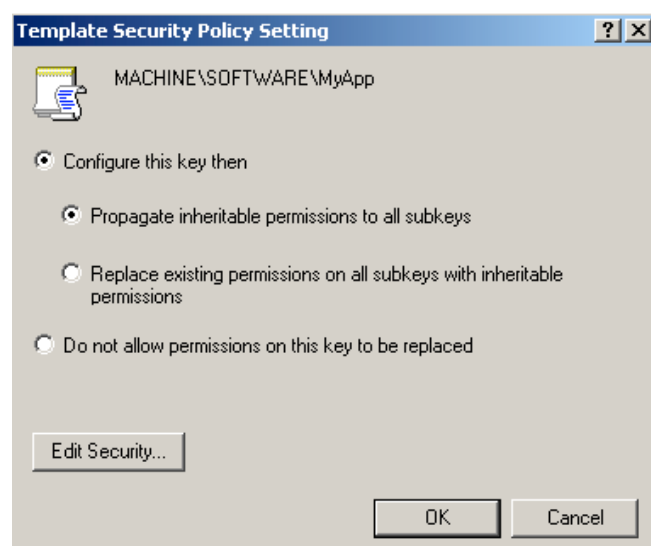


Figure 1.17: Viewing the ACL propagation (or protection) options in the Registry security policy.

In this figure, if you selected Do Not Allow Permissions on This Key to Be Replaced, you'd achieve the effect I just described—protecting the ACLs on this key. You can use Registry security policy to enforce Registry key security on computers that process a given GPO. Note that because the Registry is unique on every Win2K device, this policy is enforced on every device that is subject to it, including all domain controllers, member servers, and workstations.

File System

The File System security policy functions very much like the Registry security policy except that it applies to an NTFS volume rather than Registry keys. File System security lets you centrally manage NTFS permissions on servers and workstations. Keep in mind that this is different than share permissions, which can't be managed using this policy.

File System policy is handy when you want to enforce security on files or folders across multiple servers in your AD infrastructure. However, like Registry security, it can be tricky to manage. For one thing, File System policy requires that you set security restrictions using an absolute drive letter (or environment variable that evaluates to a drive letter) and folder name. For example, suppose you have a number of servers that have a file folder called \Apps, but on each server, \Apps may be on the C: drive or the D: drive. File System policy requires that you specify an absolute drive letter (see Figure 1.18).

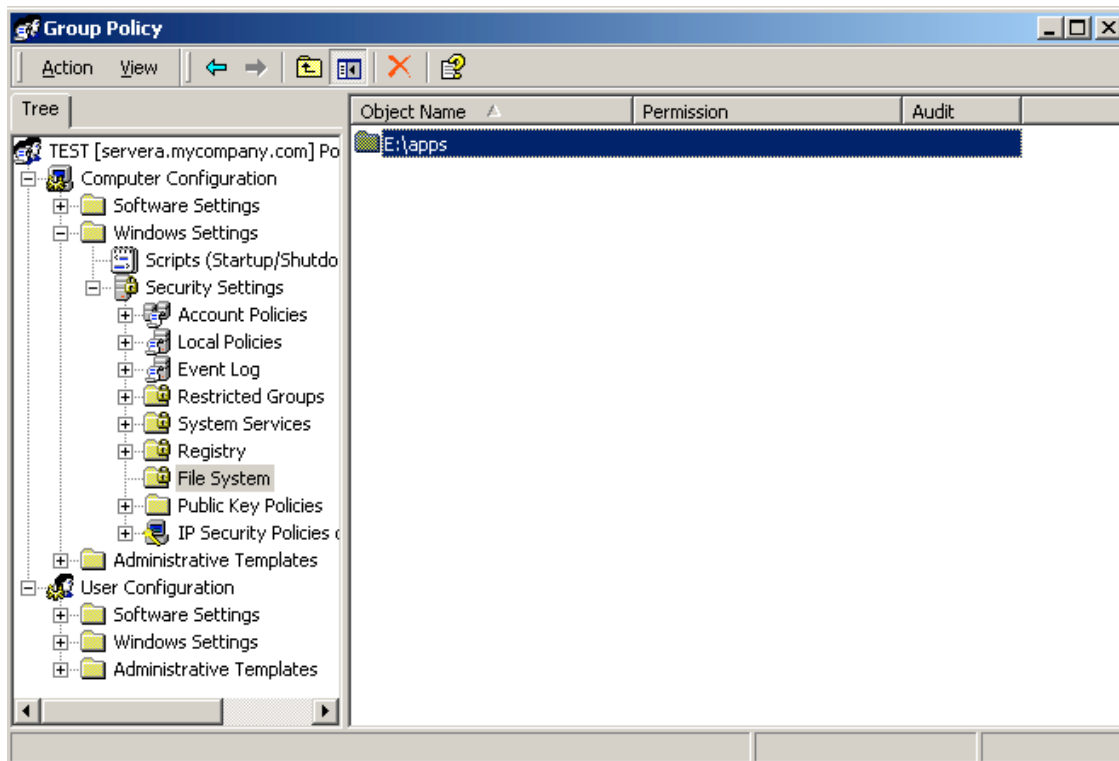


Figure 1.18: Setting up File System security policy using an absolute drive letter.

This figure shows how I've set up File System security policy on the E: volume under the apps folder. When computers process the GPO containing this policy, and if they have an E:\Apps folder, the permission policy will be applied.

Public Key Policies (Machine and User)

This policy lets you specify behavior related to both computer- and user-specific use of a public key infrastructure (PKI) implementation based on Microsoft's Certificate Services. Let me just point out that many enterprises may not be implementing a PKI solution that is integrated with their Win2K infrastructure. If that's the case, this policy section won't have much value for you. However, for those who are implementing PKI, and especially a Microsoft-based PKI solution, the Public Key policies listed here support that implementation. Specifically, on a per-machine basis, the following policies are supported:

- **Encrypted Data Recovery Agents:** Lets you specify users or groups who can decrypt files that have been encrypted by users using Win2K's Encrypted File System feature. Recovery Agents are used when the person who encrypted a file or files is unable or unavailable to decrypt them.
- **Automatic Certificate Request Settings:** Lets you specify that machines that are subject to this policy automatically receive prescribed public key certificates from AD.
- **Trusted Root Certification Authorities:** Lets you import lists of trusted public key certificates, certificate authorities (CAs), or certificate revocation lists, which are then distributed to all machines that are subject to this policy.
- **Enterprise Trust:** This feature lets you assign certificate trust lists (CTLs) to machines that are subject to this policy. CTLs are lists of root CAs. If a computer has a list of valid root CAs, it can accept certificates they certify without question.

On a per-user basis, there is a Public Key policy, similar to the Enterprise Trust machine policy, that lets you assign a CTL to users who are subject to the GPO. The final machine-specific Public Key policy is called IP Security Policies on Active Directory. Using this policy (shown in Figure 1.19), you can enforce Internet Protocol Security Protocol (IPSec) usage on computers that are subject to this GPO.

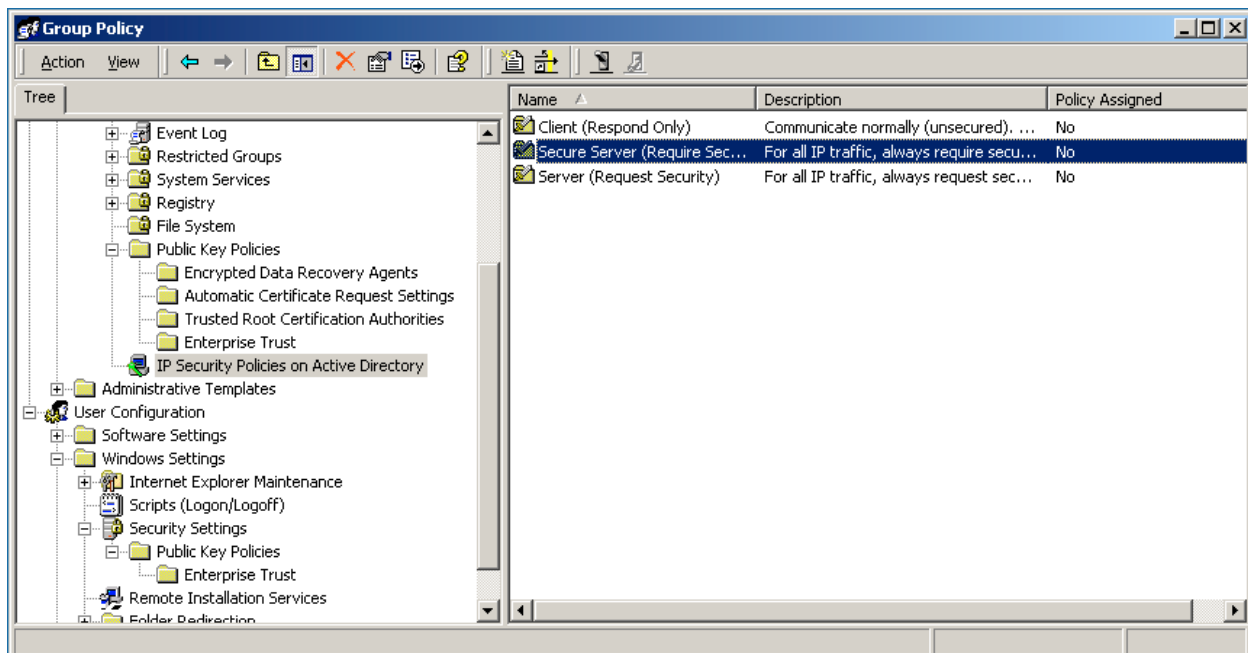


Figure 1.19: Viewing the IPsec policy settings in a GPO.

For example, using this policy, you can specify that all IP traffic to and from servers that are subject to this policy can request or require encrypted (IPsec) communications.

Windows Settings | Scripts (Startup/Shutdown)

Windows NT 4.0 supported the notion of logon scripts that were defined on a per-user basis in User Manager. Logon scripts could be batch files, executables, WSH scripts, or even Perl scripts. In Win2K, the concept has been extended in Group Policy. (Per-user logon scripts are still supported in AD users objects.)

GPOs support computer-specific startup and shutdown scripts. Startup and shutdown scripts work just as the names imply. When a Win2K workstation or server subject to a GPO-based startup script starts up, it processes the GPO and runs the startup script in the context of the special LocalSystem user account. Similarly, when the workstation or server shuts down, any shutdown scripts defined by the GPO are processed at that time.

Startup and shutdown scripts can help you perform administrative tasks that don't require user intervention or report on the behavior of computers when they restart. For example, you can write a shutdown script that sends an e-mail to a particular mailbox whenever a server or group of servers subject to a GPO shuts down. Because these scripts are based on a GPO, and because a computer can be subject to multiple GPOs at various levels (site, domain, and OU), a computer might process many shutdown and startup scripts. Depending on your perspective, this can be either a good or a bad thing!

Windows Settings | Scripts (Logon/Logoff)

Like shutdown and startup scripts in GPOs, logon and logoff scripts are per-user scripts that run in a user's security context whenever a user subject to a GPO that defines these scripts logs on to or off a Win2K device. Figure 1.20 shows how logon scripts are defined in a GPO.

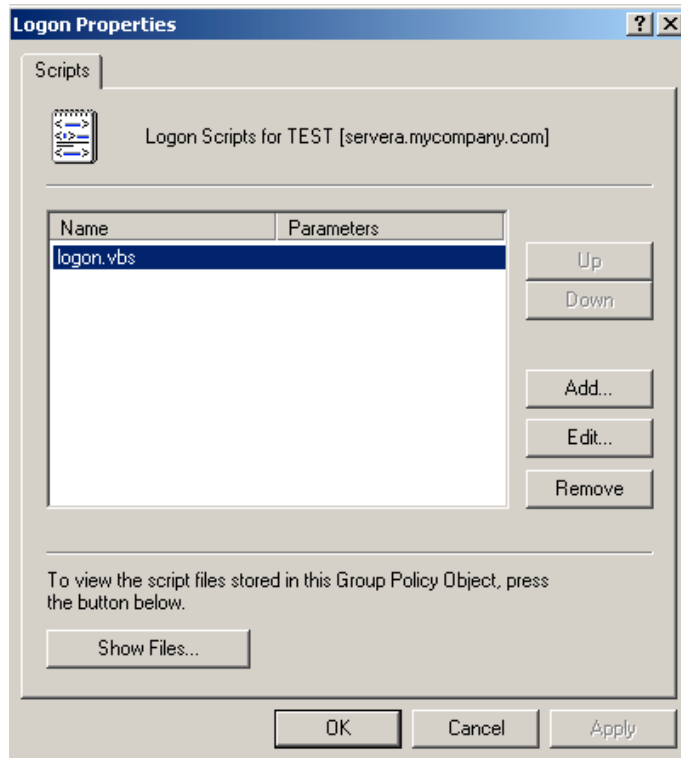


Figure 1.20: Defining a logon script in a GPO.



In addition to defining multiple scripts across GPOs, you can define multiple logon or logoff scripts in a single GPO. As you can see, using this feature can quickly become overwhelming.

Windows Settings | Internet Explorer Maintenance

Internet Explorer (IE) Maintenance policies are user-specific policies designed to apply the features that are available in the IE Administrators Kit to GPO-based policy that can be centrally managed. Using IE Maintenance, you set policy under five categories—Browser User Interface, Connection, URLs, Security, and Programs.

Browser User Interface

This policy section lets you control aspects of the IE user interface (UI) for those users who are subject to the GPO. UI elements such as the IE window title, custom bitmaps, and custom logos are controlled from here.

Connection

This policy section lets you enforce IE configuration related to network connections, automatic proxy configuration scripts (or manual proxy settings), and supported user-agent strings. Figure 1.21 shows an example of setting Connection policy for a manual proxy.

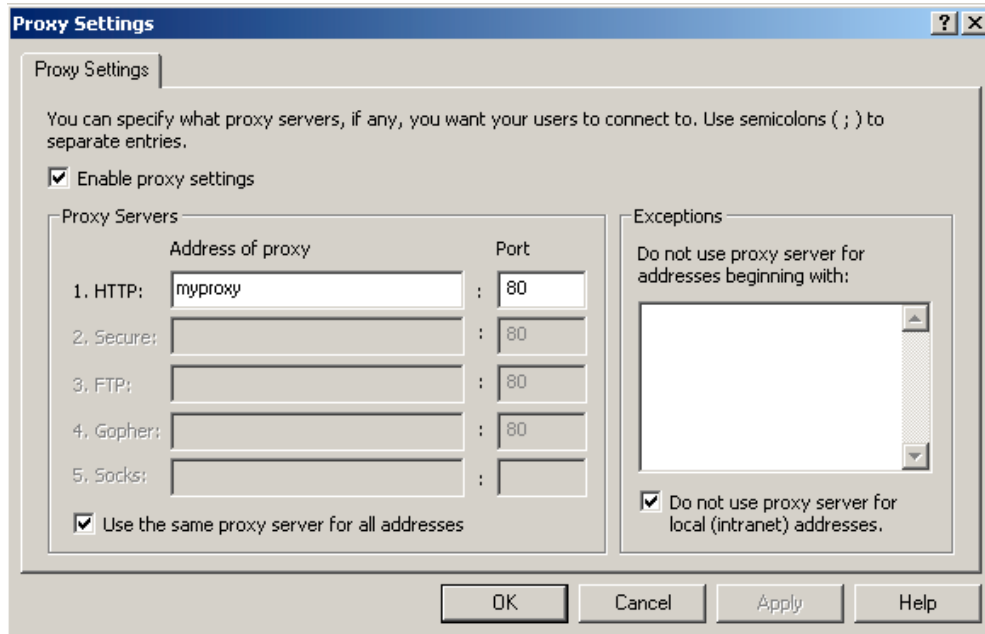


Figure 1.21: Setting manual proxy settings using the IE Maintenance Connection policy.

URLs

The URLs policy lets you define pre-set lists of favorites (bookmarks), important URLs such as home page and search page buttons, and Active Desktop channels.

Security

This policy item lets you define and enforce the different IE security-zone settings (Intranet, Internet, Trusted, and so on) as well as content-rating settings (which restrict content based on its rating). You can also enforce Microsoft Authenticode support in this policy. Authenticode guarantees that code that is downloaded by IE is safe to use, and it uses public key certificates to certify the trustworthiness of the code. The Security policy section lets you defined trusted authorities, whom you trust to allow your users to download code from within IE.

Programs

The Programs policy section lets you enforce the support programs that your IE users use within the context of a Web browsers. For example, if a user clicks an e-mail address on a Web site, what e-mail program is launched? Or if a user wants to edit an HTML file, which HTML editor is launched? Figure 1.22 shows the settings that are available in the Programs section.

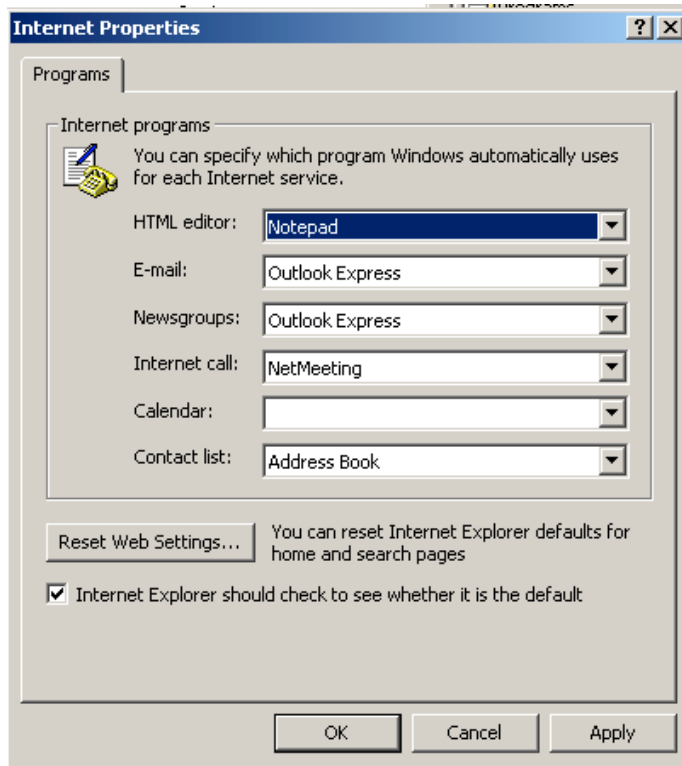


Figure 1.22: Viewing the settings available in the Programs policy item.

Windows Settings | Folder Redirection

Folder Redirection is a user-specific GPO policy that lets you redirect certain well-known user-profile directories to alternate file paths. NT 4.0 System Policy had a similar feature known as Custom Shell Folders. However, Folder Redirection in Group Policy is much “smarter” than the old NT 4.0 approach in a number of ways. However, before I describe how Folder Redirection is smarter, let’s look at what folders can be redirected:

- Application Data
- Desktop
- My Documents (and My Pictures in My Documents)
- Start Menu

Each of these folders typically resides in the user’s profile (usually stored in %Systemroot%\Documents and Settings\%Username%). When users are subject to a GPO-based Folder Redirection policy, Folder Redirection provides a way to redirect these profile-based folders to alternate locations. For example, you might choose to redirect the My Documents folder for all of the users in a particular OU to a server-based home share that is backed up frequently.

Whereas NT 4.0 redirected all users subject to a particular GPO, Folder Redirection allows you to redirect users to different server-based folders based on their group membership. This is known as advanced (as opposed to basic) redirection, and it's illustrated in Figure 1.23.

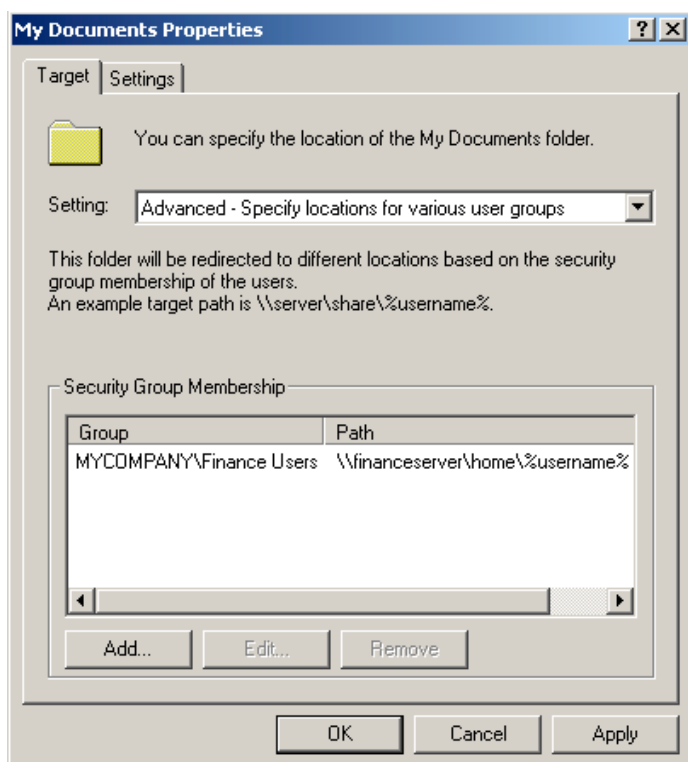


Figure 1.23: Defining the advanced Folder Redirection feature in a GPO.

In this figure, for users processing this GPO who are members of the Finance Users group, I've defined an advanced Folder Redirection policy to redirect My Documents to the home share folders \\Financeserver\Home\%Username% .

Folder Redirection is smart in other ways too. For example, you can define settings for a Folder Redirection policy that instruct the GPO to move the contents of, for example, My Documents from the user's profile to the new, redirected location when the GPO is first applied. But if the user is no longer subject to the policy, you can also tell Folder Redirection to either move the contents back to the user's profile or leave them where they are.

Folder Redirection can also grant exclusive security rights to a folder that has been redirected, as you might want to do for personal documents typically found in My Documents. In this case, Folder Redirection takes care of applying ACLs to the folder so that only the user who is being redirected has access to that folder and its contents. Folder Redirection is also useful for redirecting user folders to read-only server shares. This is especially appropriate for folders like Start Menu or Application Data, whose contents you might want to be identical for all users.

Windows Settings | Remote Installation Services

The Remote Installation Services (RIS) policy is fairly simple. If you're using RIS to install Win2K workstations, you know that one of the first steps is to log on to an AD domain. If you

log on using an AD user account that is subject to a GPO where RIS policy is defined, that policy may restrict what you can and cannot do in the RIS menus. Figure 1.24 shows the RIS policy options that are available.

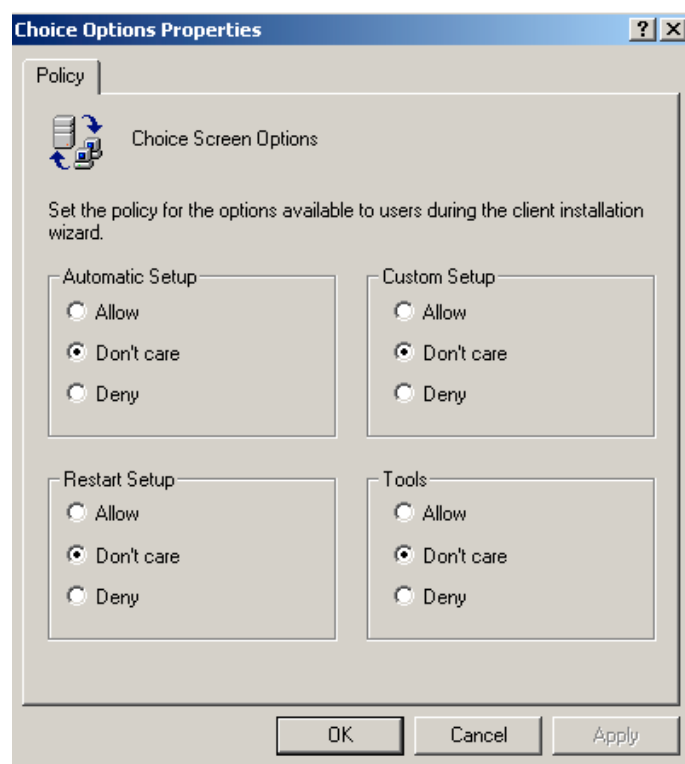


Figure 1.24: Viewing the Remote Installation Services policy settings.

In this figure, you can see that the policy provides the ability to control four aspects of a RIS setup—automatic setup, custom setup, restart setup, and Tools. Each section has three policy settings—Allow, Don't Care, and Deny. If, for example, you define a Deny policy for custom setup, users subject to this GPO cannot choose the manual setup option from the RIS installation menus. In the normal course of your work with GPOs, I'd be surprised if you find much use for this policy. But it's useful to know it's available.

Administrative Templates

Administrative Templates are available in both the per-user and per-computer parts of a GPO. Administrative Templates are NT 4.0 System Policy—or more specifically, an enhanced version of NT 4.0 System Policy. They let you enforce particular Registry settings in either the HKEY_CURRENT_USER (per-user) or HKEY_LOCAL_MACHINE (per-machine) hives.

Administrative Templates use an approach similar to System Policy to define what Registry keys and values can be set. Namely, Microsoft provides (and you can define) .adm files that use a special macro language to define the keys and values that you want to control. When you set Administrative Template policy, you choose from three states for a given policy item:

- **Enabled:** The policy item is actively enforced (corresponds to a checked value in NT 4.0 System Policy).

- **Disabled:** The policy item is actively not enforced or not done. In other words, if a Registry value is set to 1, and 0 is its “off” state, disabling the policy will set the Registry value to 0. (This corresponds to an unchecked value in NT 4.0 System Policy.)
- **Not Configured:** The value for this Registry policy item is left alone. That is, if the value in the Registry is 1, a GPO with *Not Configured* defined for that value will have no impact. (This corresponds to a “grayed” value in NT 4.0 System Policy.)

The .adm files that ship in Win2K are more numerous and provide more Registry-based policy settings than those provided in NT 4.0. This makes sense when you consider the number of additional features in Win2K. Figure 1.25 shows an example of some of the Administrative Template settings available in Win2K GPOs.

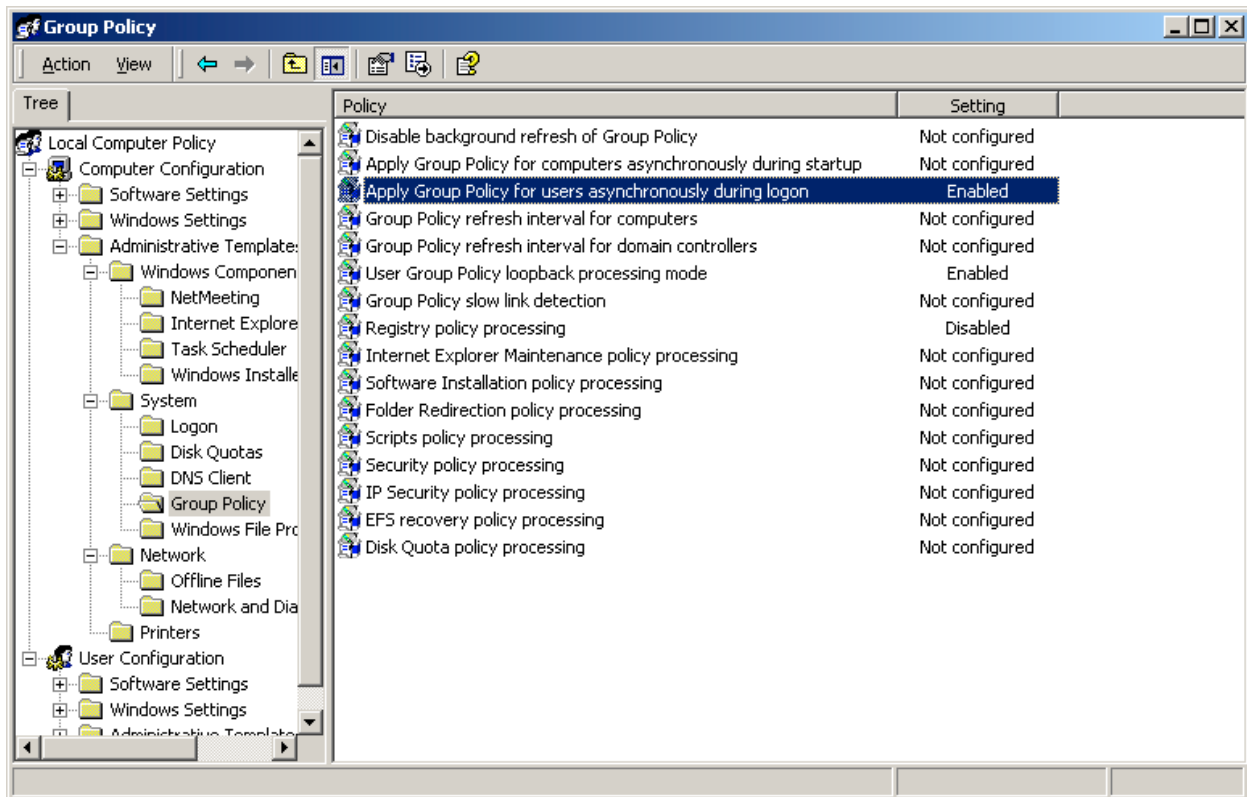


Figure 1.25: Viewing some Administrative Template policy settings in a GPO.

This figure happens to show some Administrative Template policies that you can set to control the behavior of GPO processing itself on a given computer.

In NT 4.0 System Policy, if you set a Registry value by policy and that policy file was removed or no longer applied, the value would remain as the policy had specified—it would remain “tattooed”. However, in Win2K’s Administrative Templates, Microsoft now distinguishes between Registry settings that are policies and those that are preferences.

A policy is defined as any values that are modified under one of the following four Registry keys:

HKLM\Software\Policies (per machine)
HKLM\Software\Microsoft\Windows\CurrentVersion\Policies (per machine)
HKCU\Software\Policies (per user)
HKCU\Software\Microsoft\Windows\CurrentVersion\Policies (per user)

The policies defined by these Registry keys don't suffer the "tattooing" fate. If a GPO-based Administrative Template policy is no longer active, the Registry value is removed as if the policy had never been there. Preferences, on the other hand, are Administrative Template-based policy that makes changes to Registry keys other than the four listed above. If a GPO that sets a preference is removed, the Registry values set by that preference are left as is.

Finally, just as with NT 4.0 System Policy, you can customize existing or create new .adm template files to add additional Registry lockdown functionality to your GPOs. You'll find that the macro-language syntax in Win2K is quite similar to NT 4.0 .adm files, and indeed, NT 4.0 .adm files typically work without change in Win2K.

One nice enhancement Microsoft made to Administrative Templates is support for a new .adm tag called Explain. The Explain tag lets you add detailed textual explanations of what a particular policy item is doing—right in the Group Policy snap-in. You'll notice that Microsoft has done a pretty good job of using this feature to document the .adm files that they provide. If you create your own custom files, I encourage you to follow suit.

Summary

In this chapter, I reviewed the purpose of GPOs, how they're processed, how to edit them, and what features they provide. As you can begin to see, GPOs are a powerful and complex tool that can either greatly enhance or greatly harm your Win2K infrastructure.

 The rest of this book focuses on how to use GPOs to enhance your infrastructure and prevent them from harming it. Chapter 2: Discusses a GPO in detail and how it interacts with AD, Chapters 3 and 4 drill into the features described in this chapter with an eye to helping you use them in practice and design and deploy an effective GPO infrastructure, Chapter 5 talks about some of the limitations you're likely to encounter as you use this technology, and Chapter 6 provides some tools and troubleshooting techniques for managing your GPOs when things go awry!

Copyright statement:

This site contains materials created, developed, or commissioned by Realtimepublishers.com, Inc. and is protected by international copyright and trademark laws. No material (including but not limited to the text, images, audio, and/or video) may be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice. If you have any questions about these terms, or if you would like information about licensing materials from Realtimepublishers.com, please contact us via e-mail at info@realtimepublishers.com.