realtimepublishers.com®

The Definitive Guide™ To

Securing Windows in the Enterprise

SCRIPTLOGIC

Don Jones

Chapter 4: Securing Active Directory....................76
Becoming Less Than Secure....................76
Getting AD Under Control....................77
Using Granular Permissions....................81
    Permission Strategy....................81
    The Delegation of Control Wizard....................82
    Manually Working with Permissions....................84
Maintaining Consistency....................87
    Command-Line Utilities....................87
    Template-Based Administration....................88
Security Reporting....................92
Overlooked (and Ignored) Security Problems....................95
    Unnecessary Security Principles....................95
    Disabled Accounts....................96
    Accounts with Non-Expiring or Old Passwords....................97
    Unlinked GPOs....................98
Summary....................99

Copyright Statement

Chapter 4: Securing Active Directory

AD comes out of the box in a pretty secure state, particularly in WS2K3, which uses secure Lightweight Directory Access Protocol (LDAP) by default and uses a fairly locked-down set of default permissions and configuration settings. (Win2K also uses secure LDAP by default once you install SP3 or later.) Unfortunately, AD is literally what you make of it, meaning it comes out of the box almost entirely useless until you create users, computers, organizational units (OUs), Group Policy objects (GPOs), and so forth.

Initially, most organizations take the time to plan AD deployments and come up with a reasonably secure initial configuration. However, over time—as objects are added, removed, updated, and moved around—AD often becomes somewhat less than secure. Nobody is really to blame: AD itself is designed to give you a lot of flexibility and won't complain if you don't follow best practices because you may have specific organizational needs that prevent you from doing so. Administrators can't be faulted, because too often they're caught up in the heat of the battle, dealing with less-experienced junior administrators, inheriting environments that weren't well-configured to begin with, and so forth. In addition, AD itself can be incredibly complex, so it's hardly surprising that a few security practices drop through the cracks now and again.

But AD is, much like your physical network infrastructure, a core part of your organization's operations and security. As more applications rely on AD as a central directory, making and keeping AD secure becomes more important to securing everything else in your enterprise. This chapter focuses on how AD becomes less-than-secure, how you can work with AD security more effectively, which AD security items are often overlooked or neglected, and how you can work to keep AD more secure on a continuing basis.

Becoming Less Than Secure

How AD becomes less than totally secure differs from organization to organization, but fundamentally the problem is *evolution*. When an organization first deploys AD, there's often a focus—or at least a concern—on security, meaning care is taken to set up security groups, place users in the right OUs, and so forth. Many experienced administrators deploying AD have lived in older, chaotic environments and take the AD deployment as an opportunity to "do things right" for a change. The deployment becomes a *revolution*, meaning the old ways—and old domains—are thrown out the window and a new, done-right-this-time way is instituted. Few organizations, in other words, do an initial AD deployment that has a lot of security problems built in.

Over time, though, AD *evolves* to meet the organization's continuing needs. By evolution, I'm referring to small, incremental changes made by a large number of people over a longer period of time. Users come and go, user groups are added and changed, operational and other issues drive changes to the directory structure, and so forth. Unfortunately, many of these incremental changes are made without as strong a security consideration as in the initial deployment. Often, these changes are made without a full understanding of the directory's initial structure and the reasons behind it, meaning the changes often defeat or work around the directory's initial security designs. For example, in the rush to fix a problem, an administrator might assign permissions in the directory directly to a user account rather than placing that user into a group that already has the necessary permissions. This direct assignment of permissions to a user account—particularly in light of AD's complex system of permissions inheritance—often leads to unforeseen privileges being granted to that user, either immediately or later down the road as the directory evolves further.

A significant problem with AD, as it ships, is that it provides very few tools for gaining any kind of deep perspective into the directory's security configuration. It's difficult, for example, to determine what permissions a given user may have in various places in the directory; AD's administrative tools simply weren't design to provide that kind of functionality. Therein lies perhaps the biggest security risk in AD: An inability to quickly and easily see what's already in place, security-wise. Because AD's permissions layer on one another, making changes without seeing what's already in place can compound existing problems as well as create new ones. So what can you do?

Getting AD Under Control

The first step is to get AD under control, meaning you need to design and implement a solid AD change management system. This system doesn't need to be terribly complex, and it doesn't need to involve expensive tools (although some great tools exist to help). The idea behind change management is that change is inherently dangerous. Why? Take an organization that has deployed a well-designed, well thought-out directory. In the beginning, everything is good—things are secure, well-configured, and so forth. It's not until changes begin to happen—evolution—that problems begin to creep in and build up. So change is dangerous. Not bad, exactly; without changes, the directory quickly becomes useless. Change is dangerous simply because it has the potential to cause problems.

Change management attempts to mitigate those problems by putting changes through a process designed to ensure that changes are always reviewed in context of the directory's original design. In other words, changes are reviewed to make sure they're not defeating or bypassing the security structure that was built into the directory upon its deployment. Figure 4.1 shows a sample change management process. It's a relatively simple process that's designed to help safeguard the directory against changes which, over time, will reduce the directory's overall security.

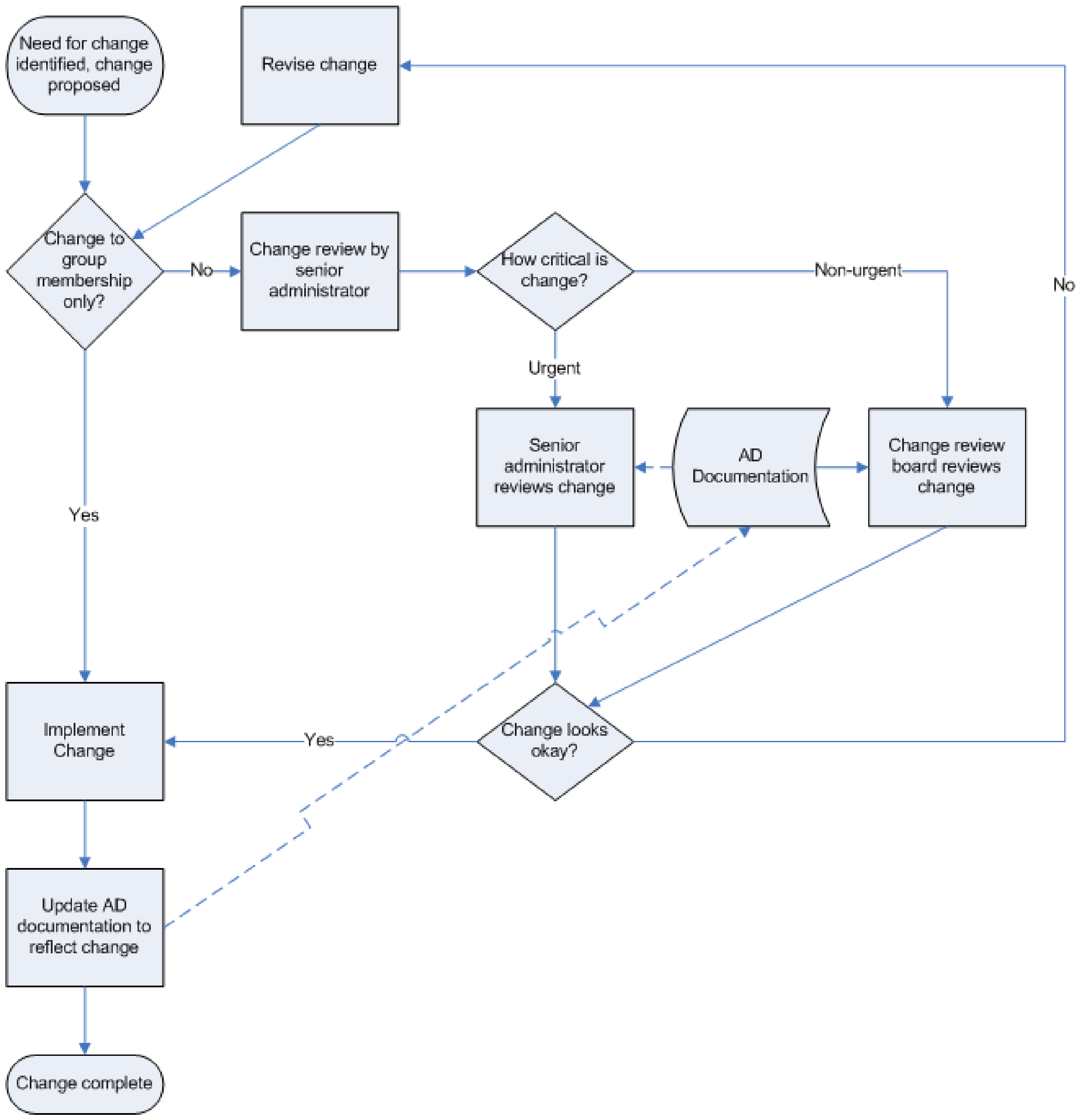

Need for change identified, change proposed
Revise change
Change to group membership only?
No
Change review by senior administrator
How critical is change?
Non-urgent
No
Urgent
Senior administrator reviews change
AD Documentation
Change review board reviews change
Yes
Implement Change
Yes
Change looks okay?
Update AD documentation to reflect change
Change complete

Figure 4.1: Example AD change management process.

This change management process—based on best practices recommended by the Information Technology Infrastructure Library (ITIL—for more information, go to http://www.ogc.gov.uk/index.asp?id=2261)—is designed to meet several security and organizational needs. First, it begins when a change is proposed, perhaps a change to group membership or a change to permissions assignments. If the change is to group membership only, the remainder of the process is bypassed and the change is immediately implemented *and documented*.

> ☞ Always document changes to AD. This documentation provides a central repository of knowledge against which future changes can be reviewed. If you're simply changing a group membership, document the group's new membership along with a reason why the change was made ("User left the company," for example).

The theory is that AD permissions are assigned only to groups; thus, changing a group's membership doesn't actually directly impact the directory's security. True, some business controls need to be placed to control who can belong to what group, but those controls would occur in advance of this process so that by the time you're ready to make the change, it has already been approved at a business policy level.

Any other changes are reviewed by a senior administrator, who categorizes them based on their priority. Urgent changes are handled immediately by the same administrator (who must be different from the person proposing or implementing the change). The reviewer simply examines the change—which should be written in a very detailed fashion—and reviews the existing AD documentation. The purpose of the review is to catch any incompatibilities between the proposed change and the directory's existing configuration and security design.

If the change isn't urgent, it's queued for a change review board, which meets on a regular basis (*regular* might be weekly, or even daily, depending on your organization and the volume of changes you're dealing with). The idea is that a board composed of two or three people will catch more potential problems. For non-critical changes, waiting a few days for the board to review and approve the change isn't a big deal, and it reserves the senior administrators' time for urgent changes. Once approved, the change is implemented and the organization's AD documentation updated. It cannot be stressed enough how critical this documentation is to maintaining the directory's security over the long haul. The following list highlights examples of information that the documentation should include:

- Infrastructure information, such as where domain controllers are located, which domain controllers act as site bridges, where site links are configured, and so forth should be included.
- A listing of every group—particularly security groups, but also distribution groups—and what they are used for should be documented. In the case of security groups, a detailed list of directory permissions should be included. In the case of groups used to control file server or application permissions, document what the group provides access to.
- A listing of every user that *should* be included in every group. This listing can be used to audit your actual group memberships; any discrepancies are cause for investigation.

All of this documentation can be done without fancy tools. However, there are some fancy tools out there that can make change management a little easier. For example, NetPro offers ChangeManager for Active Directory, which Figure 4.2 shows. This tool provides a graphical portal through which changes can be proposed, reviewed, approved, and deployed. By enforcing your change management process' workflow, the tool helps to ensure that the process is always followed. It also provides reporting for recent changes to various aspects of AD, such as GPOs and Sites and Services.

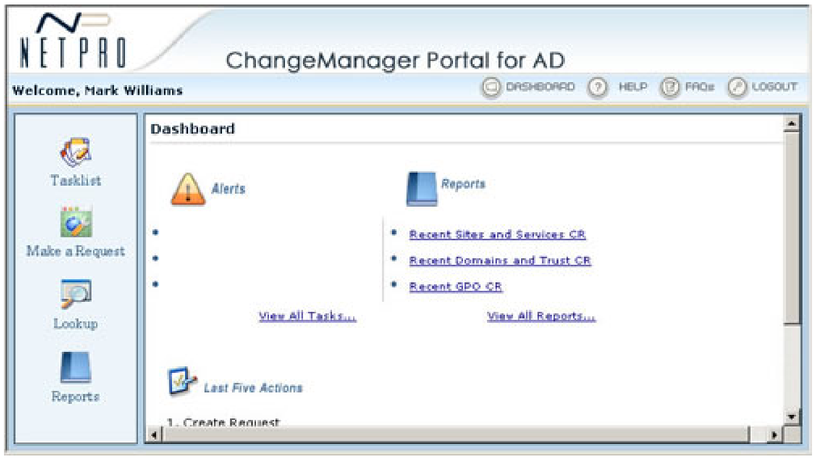
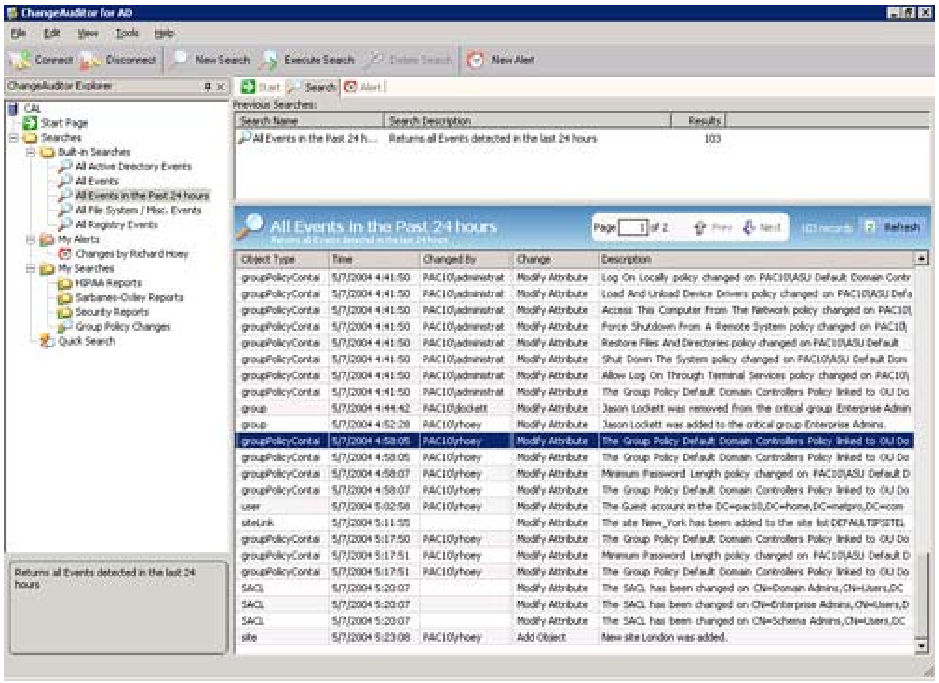


Figure 4.2: NetPro ChangeManager for AD.

NetPro offers a complementary tool, ChangeAuditor for AD (see Figure 4.3), that tracks all changes that occur within AD. As part of a change management process, you must accept the fact that out-of-process changes are likely to occur. ChangeAuditor catches them as they happen and logs them to a database, often showing you not only what the change was but also what that particular setting looked like *before* the change was made. By providing detailed reports of recent changes and filters that allow you to focus on critical changes (such as changes to security groups), you can easily audit your environment to spot changes that occurred outside your change management process.



Previous Searches:

Search Name	Search Description	Results
All Events in the Past 24 h...	Returns all Events detected in the last 24 hours	103

Object Type	Time	Changed By	Change	Description
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Log On Locally policy changed on PAC10\ASU Default Domain Contr
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Load And Unload Device Drivers policy changed on PAC10\ASU Defa
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Access This Computer From The Network policy changed on PAC10\
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Force Shutdown From A Remote System policy changed on PAC10\
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Restore Files And Directories policy changed on PAC10\ASU Default
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Shut Down The System policy changed on PAC10\ASU Default Dom
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	Allow Log On Through Terminal Services policy changed on PAC10\
groupPolicyContai	5/7/2004 4:41:50	PAC10\administrat	Modify Attribute	The Group Policy Default Domain Controllers Policy linked to OU Do
group	5/7/2004 4:44:42	PAC10\jlockett	Modify Attribute	Jason Lockett was removed from the critical group Enterprise Admin
group	5/7/2004 4:52:28	PAC10\rhoey	Modify Attribute	Jason Lockett was added to the critical group Enterprise Admins.
groupPolicyContai	5/7/2004 4:58:05	PAC10\rhoey	Modify Attribute	The Group Policy Default Domain Controllers Policy linked to OU Do
groupPolicyContai	5/7/2004 4:58:05	PAC10\rhoey	Modify Attribute	The Group Policy Default Domain Controllers Policy linked to OU Do
groupPolicyContai	5/7/2004 4:58:07	PAC10\rhoey	Modify Attribute	Minimum Password Length policy changed on PAC10\ASU Default D
groupPolicyContai	5/7/2004 4:58:07	PAC10\rhoey	Modify Attribute	The Group Policy Default Domain Controllers Policy linked to OU Do
user	5/7/2004 5:02:58	PAC10\rhoey	Modify Attribute	The Guest account in the DC=pac10,DC=home,DC=netpro,DC=com
siteLink	5/7/2004 5:11:55		Modify Attribute	The site New_York has been added to the site list DEFAULTIPSITEL
groupPolicyContai	5/7/2004 5:17:50	PAC10\rhoey	Modify Attribute	The Group Policy Default Domain Controllers Policy linked to OU Do
groupPolicyContai	5/7/2004 5:17:51	PAC10\rhoey	Modify Attribute	Minimum Password Length policy changed on PAC10\ASU Default D
groupPolicyContai	5/7/2004 5:17:51	PAC10\rhoey	Modify Attribute	The Group Policy Default Domain Controllers Policy linked to OU Do
SACL	5/7/2004 5:20:07		Modify Attribute	The SACL has been changed on CN=Domain Admins,CN=Users,DC
SACL	5/7/2004 5:20:07		Modify Attribute	The SACL has been changed on CN=Enterprise Admins,CN=Users,D
SACL	5/7/2004 5:20:07		Modify Attribute	The SACL has been changed on CN=Schema Admins,CN=Users,DC
site	5/7/2004 5:23:08	PAC10\rhoey	Add Object	New site London was added.

Returns all Events detected in the last 24 hours

Figure 4.3: NetPro ChangeAuditor for AD.

The ultimate goal of a change management process is to help AD remain consistently secure on an ongoing basis. Of course, if you're in an organization that hasn't had a change management process in place, you'll likely have some cleaning up to do first.

Using Granular Permissions

A common problem in AD environments is permission assignments that are too broad. In fact, one of my biggest complaints about AD as it ships is that too few built-in security groups are offered, meaning everything pretty much comes down to the all-powerful Domain Admins and Enterprise Admins groups. Unfortunately, those are probably the worst groups to use on a daily basis. Using more granular permissions helps to reduce the negative effects of configuration errors, makes it easier to share permissions (and workload) throughout your organization, and so forth.

Permission Strategy

Most organizations dump the entire burden of AD administration squarely on their administrators and Help desk personnel. That's fine in theory, but AD was designed to be much smarter and more flexible than that. For example, let's suppose Suzie the Human Resources Director has a file server all her own on which she stores her department's sensitive files. Company policy states that only Suzie can decide who should have access to those files. You've done the right thing and set up a couple of security groups in AD, and assigned file permissions on Suzie's file server directly to those groups. However, anytime Suzie wants someone else to access the files, she has to call you to have the group membership updated. It's silly; why not let Suzie make the change herself? Of course, there are issues around creating the exact set of permissions necessary to allow her to do that and nothing more, and of giving her a user interface through which she can manage just those two groups, but I'll touch on those subjects later. My point for right now is that you should carefully examine how your directory is actually used, and come up with a permissions and management strategy that reflects the directory's real-world usage.

Principle of Least Privilege

This is a good time to discuss PoLP, the Principle of Least Privilege. The idea is that any user account should only have enough permissions to do whatever that user does for most of their time. Permissions for other tasks should be assigned to a secondary user account that the user utilizes when those other less frequently performed tasks need to be taken care of.

For example, a senior network administrator in a large organization might spend most of his or her time attending meetings, working on projects, and reading email—not changing user accounts or directly administering the directory. It makes sense, then, for that administrator to use a less-privileged user account for those day-to-day tasks, and to log on (or use the Runas utility) by using a separate user account when he or she needs to administer the directory. That separate user account would be a member of the Domain Admins group, for example; the administrator's primary account wouldn't be.

PoLP is designed to help protect the environment against both accidents and attacks. If the administrator accidentally runs a script, for example, that modifies several accounts in the domain, the script won't run because of the administrator's less-privileged primary account. If a virus attacks the administrator's computer, its effects will be limited to whatever the administrator's less-privileged primary account can do.

PoLP doesn't just apply to administrators, however. Suzie in HR might have a separate account that gives her permission to modify the two security groups that are used to control access to her file server's files; you can then perform more detailed auditing of Suzie's administrative activities by filtering for that privileged account, and you'll ensure that Suzie's regular account can't be compromised to modify access to the sensitive HR files.

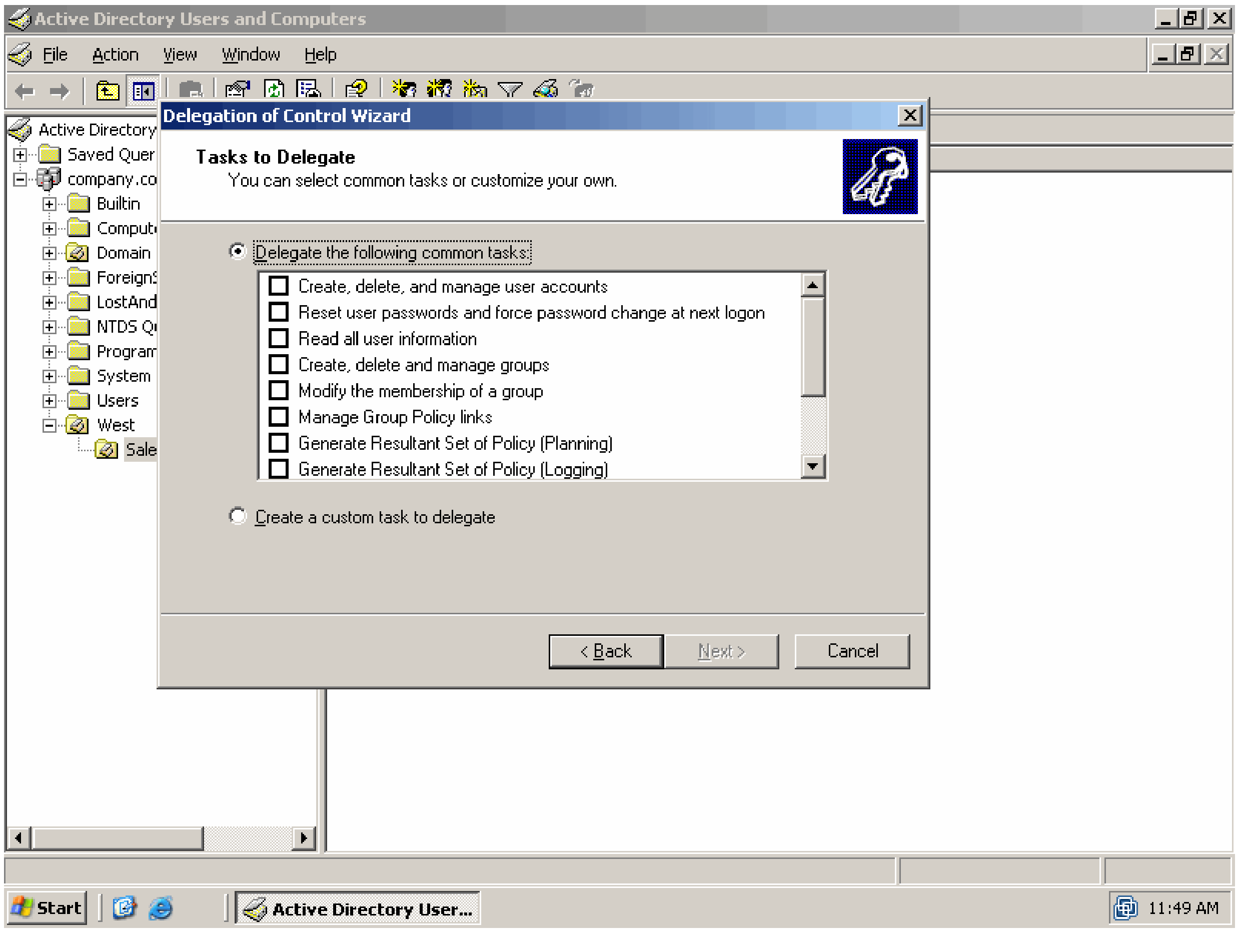
PoLP is something that every environment *should* use, although regrettably very few do, mainly because of the perceived inconvenience of having multiple user accounts per person. In fact, Windows' built-in Runas tool, along with the ability to right-click applications and shortcuts and select a "Run as" menu option, makes using alternative credentials pretty easy.

A first step, then, in securing AD more effectively is to rethink how you use security permissions and user accounts. Think about more granular permissions. I like to start with the notion that nobody should be using a Domain Admin account on any kind of regular basis. That built-in group simply has such broad permissions that it isn't safe. Instead, create your own groups that reflect the actual tasks users are performing, and assign the appropriate permissions and rights to those groups. If one of your groups starts to look suspiciously like a duplicate of the Domain Admins group, then you can go ahead and use Domain Admins for users that need all those rights. But at least you'll have gone through the steps of validating the need for those broad permissions, rather than just accepting them because that is the easy thing to do.

Most security problems in AD come back to administrators taking the easy way out. Yes, Domain Admins is easy and when you're a member everything works just fine. Yes, coming up with more granular permissions and groups is more difficult. But look, if you don't want a more secure environment, why are you reading this book? Anything worth having will take some effort, and security is *very* worth having. Stick with me: I'll be introducing some tools and techniques that'll make things a bit smoother.

The Delegation of Control Wizard

Most administrators' first experience with more granular AD permissions is the Delegation of Control (DoC) Wizard, built into the AD Users and Computers console. The wizard allows you to delegate specific permissions to one or more users (bad idea) or groups (best practice), for a specific object or objects within AD. In other words, you might give the HRAdmin group permission to reset passwords on user objects within the HR OU. As Figure 4.4 shows, the wizard supports several preprogrammed permission sets, such as resetting passwords, managing Group Policy links, and so forth, or you can create your own custom tasks (permission sets).



Delegation of Control Wizard

Tasks to Delegate

You can select common tasks or customize your own.

[● Delegate the following common tasks:]

- ☐ Create, delete, and manage user accounts
- ☐ Reset user passwords and force password change at next logon
- ☐ Read all user information
- ☐ Create, delete and manage groups
- ☐ Modify the membership of a group
- ☐ Manage Group Policy links
- ☐ Generate Resultant Set of Policy (Planning)
- ☐ Generate Resultant Set of Policy (Logging)

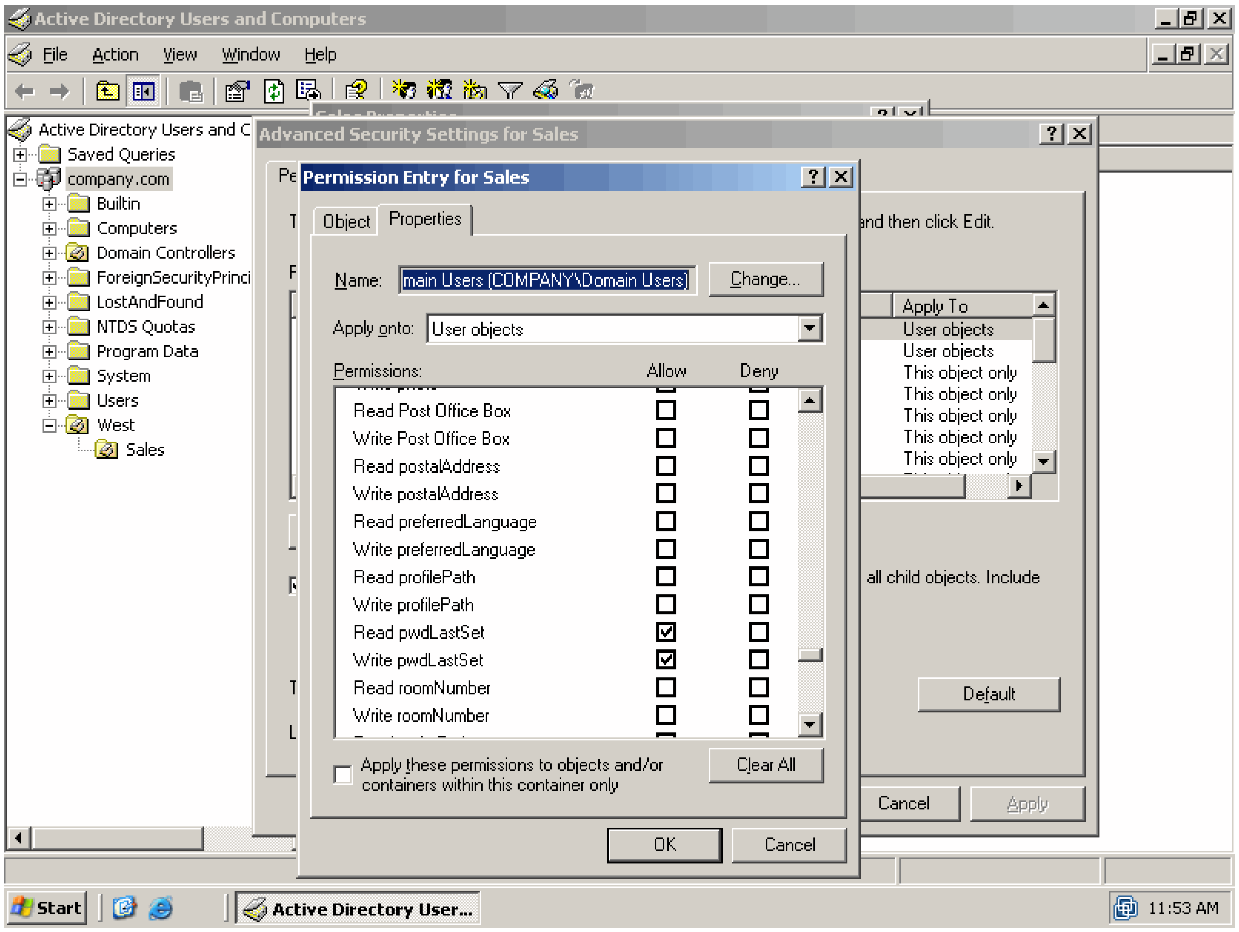
○ Create a custom task to delegate

< Back | Next > | Cancel

Figure 4.4: Running the DoC Wizard.

The *only* thing I don't like about the wizard, especially when using one of the preprogrammed permission sets, is that it's doing stuff behind the scenes. That's its job, really—to perform the permission change while hiding some of the underlying complexity. Until you fully understand that complexity, however, I think it's more valuable to make the permission changes yourself. That way you'll *know* what's happening under the hood, and you'll understand the security consequences of what you're doing more clearly.

For example, as Figure 4.5 illustrates, a single run of the wizard creates a fairly complex underlying set of permissions, requiring you to navigate through a series of dialog boxes to see what changed. The wizard didn't do anything wrong or inaccurate, but it might benefit me to know that, for example, the permission change the wizard made isn't inherited to sub-OUs; the permissions I delegated only apply to the OU I selected when running the wizard. The permission applies only to User objects, something else that the wizard glosses over a bit.



Permission Entry for Sales

Object | Properties

Name: main Users (COMPANY\Domain Users) Change...

Apply onto: User objects

Permissions:	Allow	Deny
Read Post Office Box	☐	☐
Write Post Office Box	☐	☐
Read postalAddress	☐	☐
Write postalAddress	☐	☐
Read preferredLanguage	☐	☐
Write preferredLanguage	☐	☐
Read profilePath	☐	☐
Write profilePath	☐	☐
Read pwdLastSet	☑	☐
Write pwdLastSet	☑	☐
Read roomNumber	☐	☐
Write roomNumber	☐	☐

☐ Apply these permissions to objects and/or containers within this container only Clear All

OK Cancel

Apply To
User objects
User objects
This object only
This object only
This object only
This object only
This object only

Figure 4.5: Actual permissions set after the DoC Wizard runs.

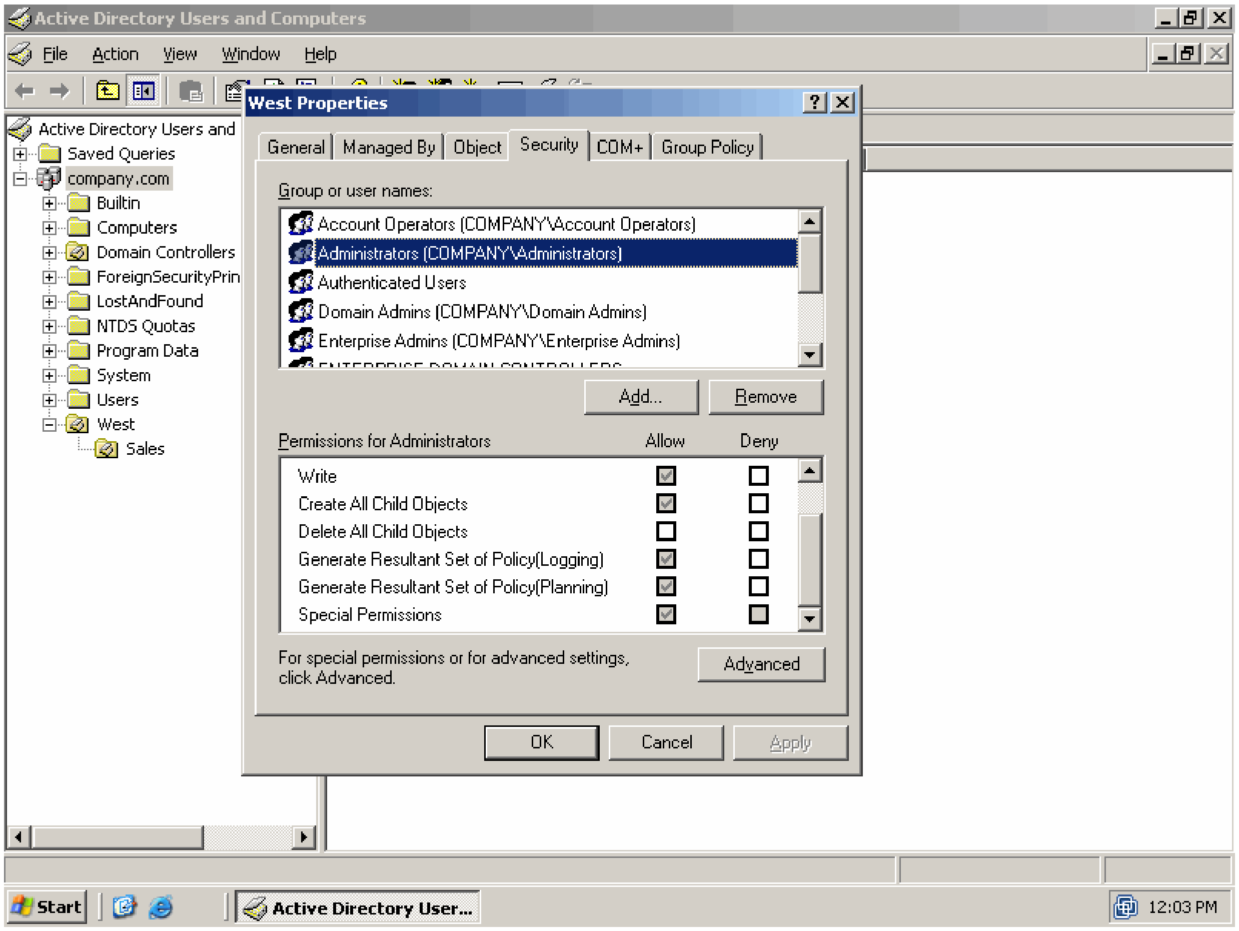
The wizard is more valuable for administrators who know how to perform its job manually, understand what the Wizard is doing under the hood, and simply use it as a timesaving device. I don't like newer or less-experienced administrators to use the wizard simply because it allows them to make significant changes with a much less thorough understanding of the result. I prefer that they manually work with AD permissions first so that they become more familiar with what's going on under the hood.

Manually Working with Permissions

AD permissions are complex because they're so flexible (or maybe it's the other way around). AD permissions allow you to:

- Give permissions to a group or user
- Assign permissions to a specific object, such as a user or OU
- Assign permissions to children of a container or OU (for example, you might assign permissions only to users within an OU but not the OU itself)
- Allow permissions to inherit to child objects, such as sub-OUs

Right-clicking an AD object (in the AD Users and Computers console, for example) and selecting Properties provides access to the Security tab (see Figure 4.6). Unfortunately, AD permissions are far too complex to be easily represented by this dialog box, so you'll nearly always find yourself facing grayed-out check boxes (indicating permissions that are inherited rather than being assigned directly to the object in question), as well as the Special Permissions check box.



Group or user names:

- Account Operators (COMPANY\Account Operators)
- Administrators (COMPANY\Administrators)
- Authenticated Users
- Domain Admins (COMPANY\Domain Admins)
- Enterprise Admins (COMPANY\Enterprise Admins)

Permissions for Administrators	Allow	Deny
Write	✓	☐
Create All Child Objects	✓	☐
Delete All Child Objects	☐	☐
Generate Resultant Set of Policy(Logging)	✓	☐
Generate Resultant Set of Policy(Planning)	✓	☐
Special Permissions	✓	☐

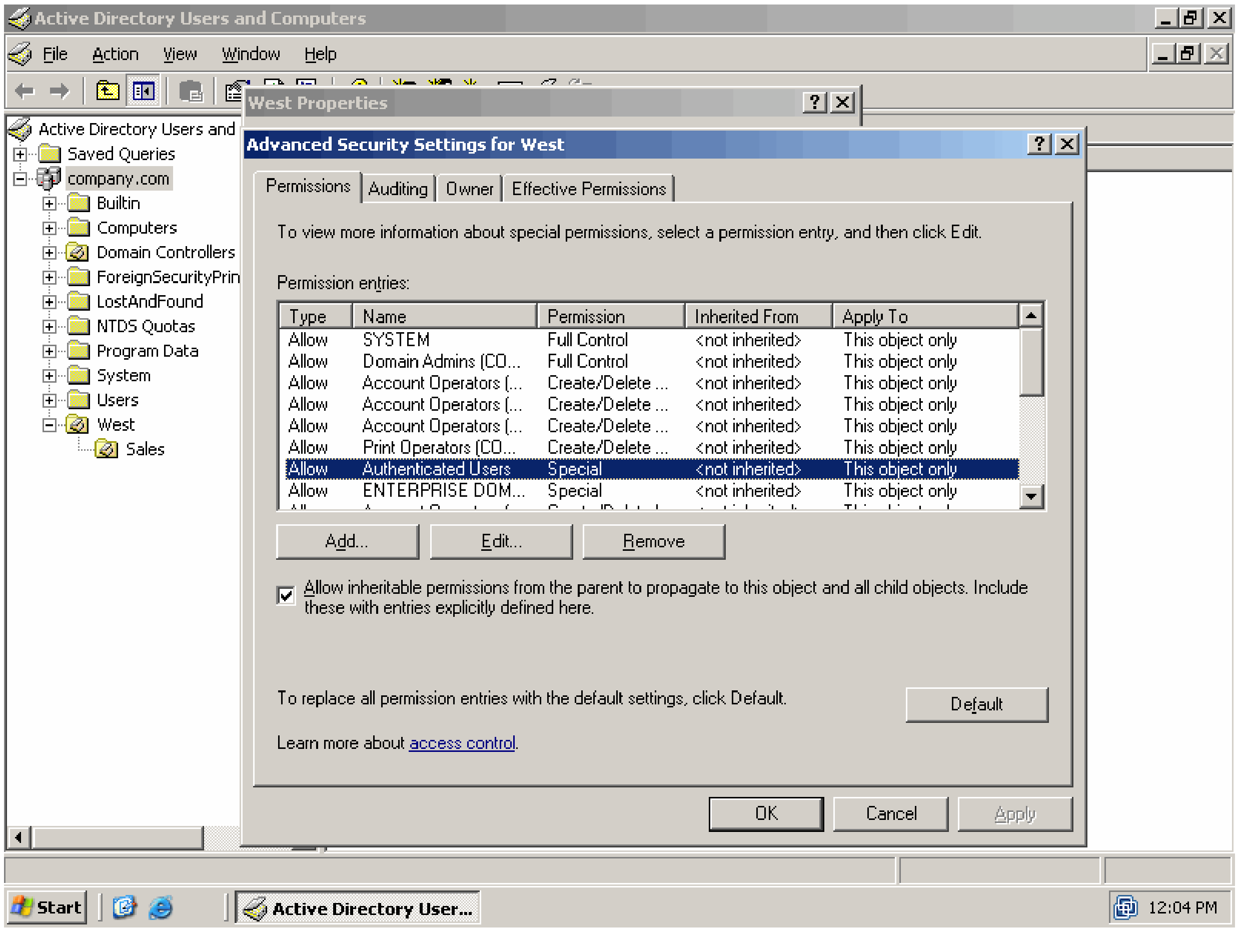
For special permissions or for advanced settings, click Advanced.

Figure 4.6: Basic AD permissions.

Dealing with Special Permissions—which nearly always exist—requires you to use the Advanced Permissions dialog box, which Figure 4.7 shows. Even then, you'll often find yourself confronted with "Special" permissions, requiring you to click the Edit button and face a third dialog box.

The Advanced Permissions dialog box (see Figure 4.7) does show some important information. It indicates where inherited permissions came from (or indicates that a permission wasn't inherited at all, but was assigned directly), and it indicates the scope of the permission: this object only, this object and all child objects, a specific type of object (such as User or Group), and so forth. You can also use the *Allow inheritable permissions from the parent to propagate to this object and all child objects* check box. When selected (the default), all permissions defined on the parent as inheritable will be inherited by this object; when cleared, no permissions will be inherited from the parent—even if said permissions were intended to be inherited. Clearing the check box effectively blocks permission inheritance, allowing you to start with a clean security slate.

This manual application of permissions lets you get *very* granular. You could, for example, allow members of a specific group to modify the membership of any other groups located in a particular OU, without giving them that ability in other OUs, any sub-OUs, and without giving them the ability to create any new group (or other) objects, and without giving them any control over groups in any parent OUs. It's flexible…and complicated.



Advanced Security Settings for West

Permissions | Auditing | Owner | Effective Permissions

To view more information about special permissions, select a permission entry, and then click Edit.

Permission entries:

Type	Name	Permission	Inherited From	Apply To
Allow	SYSTEM	Full Control	<not inherited>	This object only
Allow	Domain Admins (CO...	Full Control	<not inherited>	This object only
Allow	Account Operators (...	Create/Delete ...	<not inherited>	This object only
Allow	Account Operators (...	Create/Delete ...	<not inherited>	This object only
Allow	Account Operators (...	Create/Delete ...	<not inherited>	This object only
Allow	Print Operators (CO...	Create/Delete ...	<not inherited>	This object only
Allow	Authenticated Users	Special	<not inherited>	This object only
Allow	ENTERPRISE DOM...	Special	<not inherited>	This object only

Add... Edit... Remove

[✓] Allow inheritable permissions from the parent to propagate to this object and all child objects. Include these with entries explicitly defined here.

To replace all permission entries with the default settings, click Default. Default

Learn more about access control.

OK Cancel Apply

Figure 4.7: Advanced AD permissions.

Ultimately, however, coming up with complicated, granular permissions is the way to go. You'll distribute authority throughout your organization in a way that reflects the actual control in your organization. To use my earlier example, Suzie is the one with authority over who can be in those two HR-related security groups; AD permissions should reflect that authority whenever possible rather than forcing Suzie to call you. Since you don't have business-level authority to modify those groups, you shouldn't necessarily have AD-level permissions to do so, either.

Distributing Authority

Delegating control is often ignored because organizations believe it's too difficult to give users a user interface appropriate to their level of permission. For example, giving Suzie permission to modify her two security groups is easy; giving her a user interface to do so is more difficult. You could just give her AD Users and Computers; she wouldn't be able to *do* anything except what she's allowed to do. However, it's a poor security practice to expose people to things they have no permissions to use, so the entire AD Users and Computers console would be overkill.

One alternative is to use the Microsoft Management Console (MMC) to create a TaskPad view of the AD Users and Computers console, creating a subset of the console's functionality that reflects only what Suzie can do. Another alternative would be to write scripts or develop small custom applications that just have enough functionality to do what a particular user needs.

Companies such as NetIQ offer products designed to address this need. Its Security Administration Suite, for example, provides an in-product system of delegation and a customizable user interface that provides delegated individuals with just the user interface elements necessary to do what their permissions allow.

Whatever you decide, the point is that you *can* create limited user interfaces to reflect delegates' limited permissions and capabilities. Don't let that be a barrier in creating a more delegated security system in AD, one that more closely reflects that actual business authority in your organization.

Maintaining Consistency

One step to making AD more secure—and a major reason to adopt a good AD change management process—is consistency. Consistent configurations are more reliable simply because they're the same. For example, you may need to delegate password-reset permissions to multiple groups of people. Doing so the same way, every time, will help ensure that the delegation is done correctly. Consistency generally makes everything easier in information technology, and AD security management is no exception.

Command-Line Utilities

One way to help maintain consistency is to use scripts or command-line utilities. By performing delegation tasks, for example, using a utility or script, you'll be assured that the task is done the same way every time you do it. You don't need to worry about skipping a step in a manual process, forgetting to check a critical check box, or leaving out some other step; the script or utility will always do things consistently.

Microsoft provides a command-line utility named Directory Services Access Control Lists (DSACLs.exe), which is designed to make changes to AD permissions. The tool can be added to a batch file to make sure the file is always run consistently for various operations; you might have one batch file for each common permission- or delegation-related task that you perform. DSACLs is also a great way to apply batch corrections to incorrect security permissions within AD, as it can affect numerous objects at once, much more quickly than you could do manually.

An example of the tool’s syntax:

```
      Dsacls “OU=Peeps,DC=company,DC=com”
       /G Company\JohnD:CCDC;group;
```

This text would give user JohnD permissions over the OU Peeps. The permissions granted are Create Child and Delete Child specifically for groups; in other words, JohnD can create and delete groups within the Peeps OU.

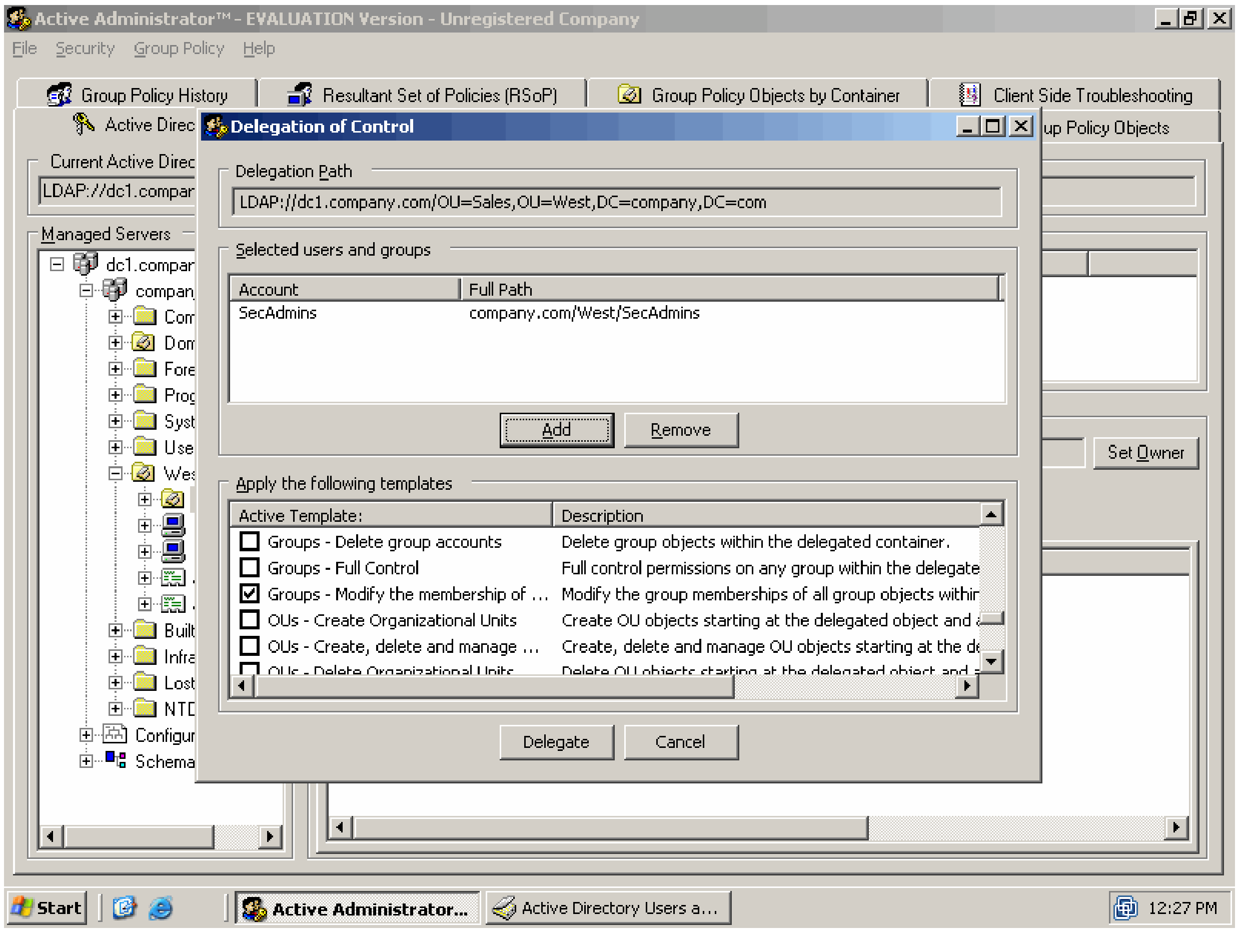
There’s a downside to using tools like this, however. First, the syntax is anything but straightforward, and you’re as likely to make a mistake running DSACLs as you are doing these tasks manually. Other command-line tools exist, but the bottom line is simply this: AD security permissions are complex, and any tools used to work with those permissions are going to be equally complex.

Template-Based Administration

Template-based administration is becoming more popular in the world of IT management, especially when it comes to security. The basic idea is that you create a bunch of templates (or rules, or policies, or whatever you want to call them) that describe how various aspects of your IT environment should be configured. You then use tools to actually apply those template settings to your environment, automatically. Oftentimes, the tools have the ability to continually enforce your templates, overwriting any unauthorized changes that are made.

An example of a product that uses this technique is ConfigureSoft Enterprise Configuration Manager (ECM). With ECM, you might define a template that requires the Messenger service to be stopped and disabled. You apply the template, using ECM, to whatever machines should be governed by that template. ECM inventories the machines and corrects their configurations to match the template. If a user or administrator manually enables the Messenger service, ECM detects that and reconfigures the machine—automatically—to comply with the template. If you ever want to enable the Messenger service, you just change your template, and ECM reconfigures everything to match.

ScriptLogic Active Administrator provides this kind of template-based administration for AD. As Figure 4.8 shows, you can create templates that assign specific permissions. In this example, I’ve opened the permissions on the West/Sales OU in a domain, and delegated the *Groups-Modify the membership of groups* security template to the SecAdmins user group. That group will now have the template’s permissions on that particular OU.


Active Administrator™ - EVALUATION Version - Unregistered Company
File Security Group Policy Help
Group Policy History
Resultant Set of Policies (RSoP)
Group Policy Objects by Container
Client Side Troubleshooting
Delegation of Control
Add
Remove
Delegate
Cancel
Set Owner
Start
Active Administrator...
Active Directory Users a...
12:27 PM

Delegation Path

LDAP://dc1.company.com/OU=Sales,OU=West,DC=company,DC=com

Selected users and groups

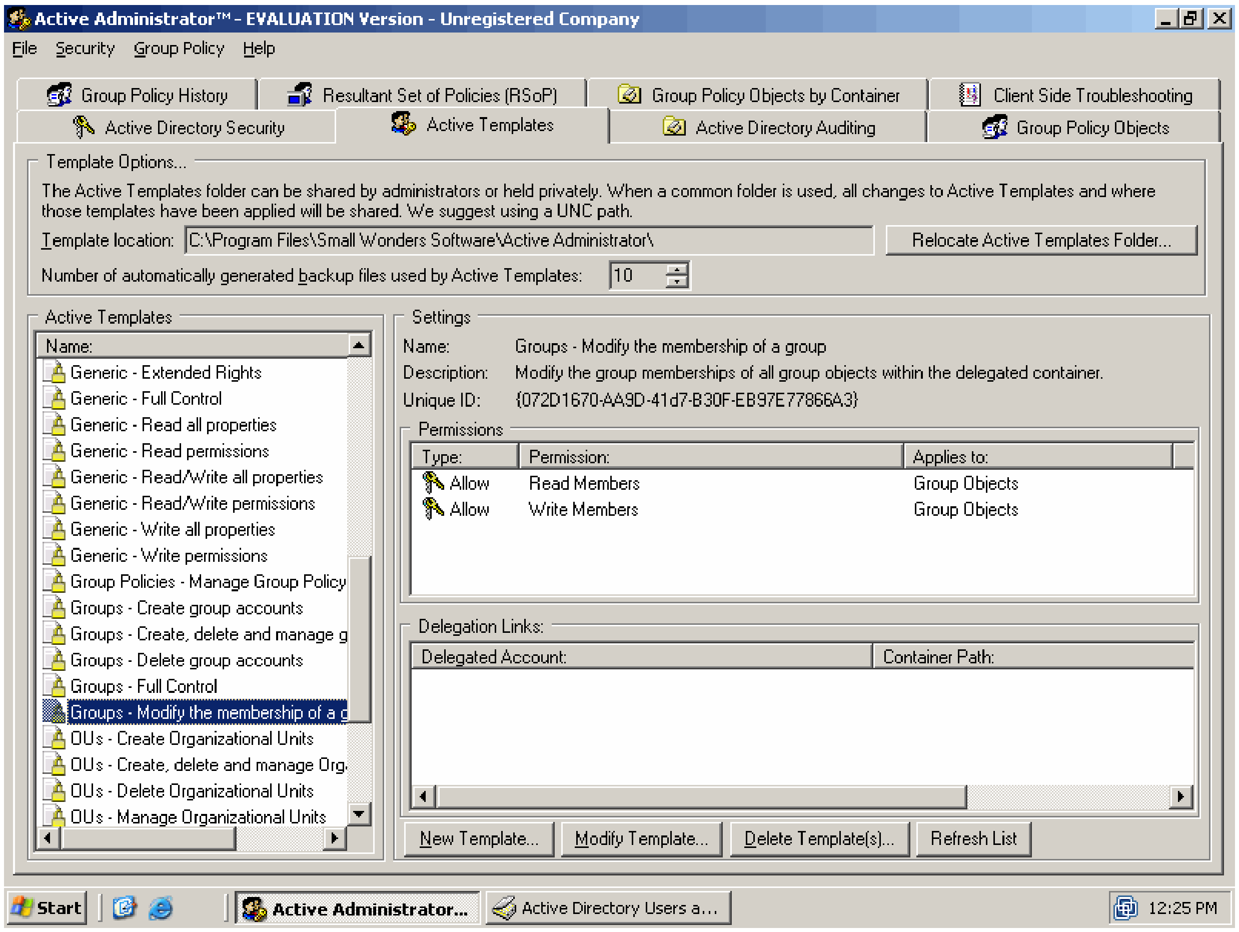
Account	Full Path
SecAdmins	company.com/West/SecAdmins

Apply the following templates

Active Template:	Description
☐ Groups - Delete group accounts	Delete group objects within the delegated container.
☐ Groups - Full Control	Full control permissions on any group within the delegate
☑ Groups - Modify the membership of ...	Modify the group memberships of all group objects withir
☐ OUs - Create Organizational Units	Create OU objects starting at the delegated object and
☐ OUs - Create, delete and manage ...	Create, delete and manage OU objects starting at the de

Figure 4.8: Creating a template of AD permissions.

This capability isn't remarkably different from AD's DoC Wizard, except that with this method you can define your own templates anytime you like. As Figure 4.9 illustrates, you have a wide range of templates to start with, and modifying them or creating your own is easy. This method allows you to create templates for whatever AD tasks are performed in your organization, making it easier to delegate permissions for those tasks to whatever users and groups require them.


Active Administrator™ - EVALUATION Version - Unregistered Company

File Security Group Policy Help

Group Policy History | Resultant Set of Policies (RSoP) | Group Policy Objects by Container | Client Side Troubleshooting
Active Directory Security | Active Templates | Active Directory Auditing | Group Policy Objects

Template Options...

The Active Templates folder can be shared by administrators or held privately. When a common folder is used, all changes to Active Templates and where those templates have been applied will be shared. We suggest using a UNC path.

Template location: C:\Program Files\Small Wonders Software\Active Administrator\ — Relocate Active Templates Folder...

Number of automatically generated backup files used by Active Templates: 10

Active Templates

Name:
- Generic - Extended Rights
- Generic - Full Control
- Generic - Read all properties
- Generic - Read permissions
- Generic - Read/Write all properties
- Generic - Read/Write permissions
- Generic - Write all properties
- Generic - Write permissions
- Group Policies - Manage Group Policy
- Groups - Create group accounts
- Groups - Create, delete and manage g
- Groups - Delete group accounts
- Groups - Full Control
- Groups - Modify the membership of a g
- OUs - Create Organizational Units
- OUs - Create, delete and manage Org
- OUs - Delete Organizational Units
- OUs - Manage Organizational Units

Settings

Name: Groups - Modify the membership of a group
Description: Modify the group memberships of all group objects within the delegated container.
Unique ID: {072D1670-AA9D-41d7-B30F-EB97E77866A3}

Permissions

Type:	Permission:	Applies to:
Allow	Read Members	Group Objects
Allow	Write Members	Group Objects

Delegation Links:

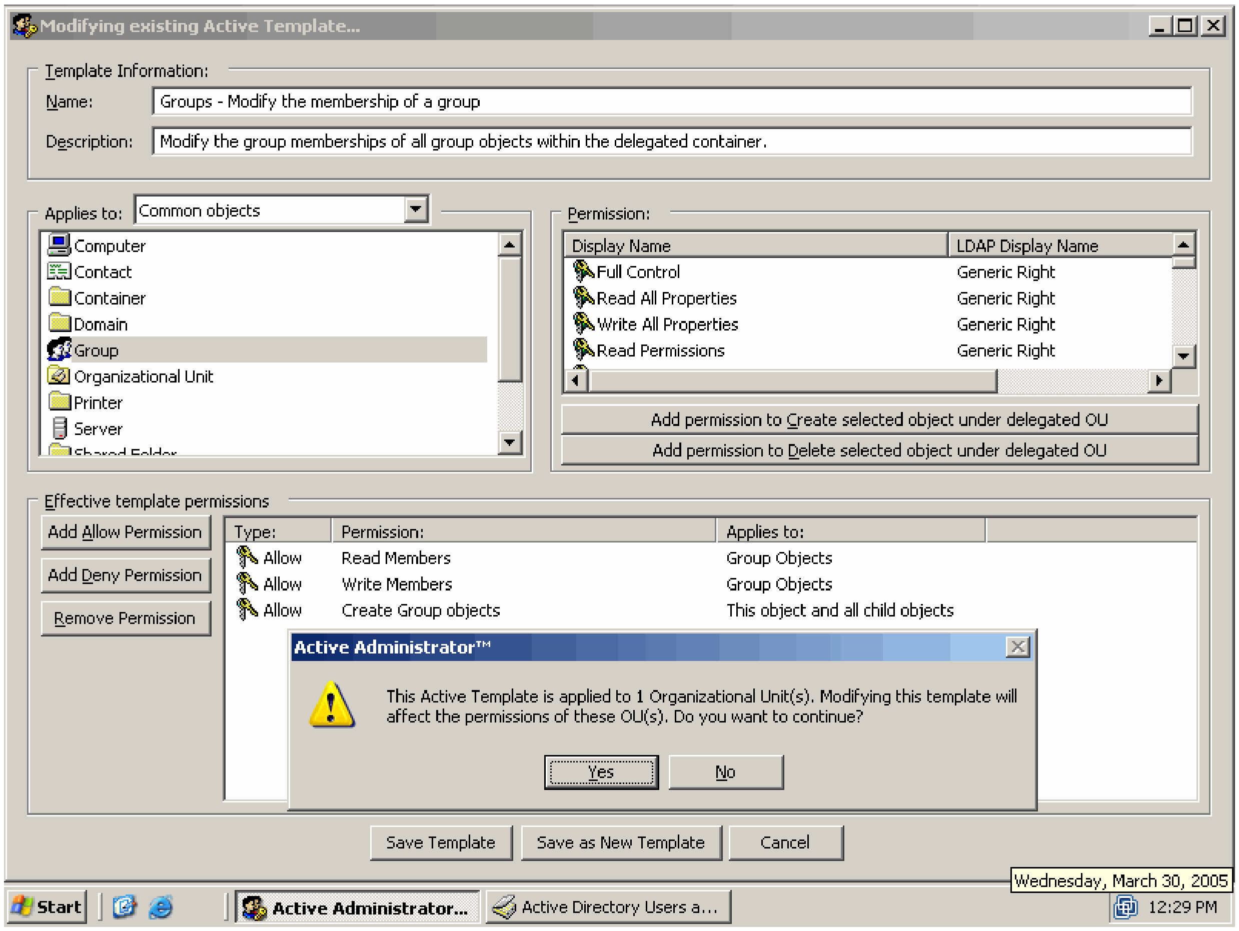
Delegated Account:	Container Path:

New Template... | Modify Template... | Delete Template(s)... | Refresh List

Start | Active Administrator... | Active Directory Users a... | 12:25 PM

Figure 4.9: Examining a security template.

Where Active Administrator really shines, though, is when you later change a template, as Figure 4.10 shows. In this example, I've modified the template I previously delegated to the SecAdmins group. I've added the ability to create new group objects. When I save the template, Active Administrator notes that it is already applied to one OU, and that saving the modified template will affect the permissions on that OU. In other words, simply by changing the template, I'm affecting AD permissions wherever this template was used.


Modifying existing Active Template...

Template Information:

Name: Groups - Modify the membership of a group

Description: Modify the group memberships of all group objects within the delegated container.

Applies to: Common objects

Computer
Contact
Container
Domain
Group
Organizational Unit
Printer
Server
Shared Folder

Permission:

Display Name	LDAP Display Name
Full Control	Generic Right
Read All Properties	Generic Right
Write All Properties	Generic Right
Read Permissions	Generic Right

Add permission to Create selected object under delegated OU

Add permission to Delete selected object under delegated OU

Effective template permissions

Add Allow Permission
Add Deny Permission
Remove Permission

Type:	Permission:	Applies to:
Allow	Read Members	Group Objects
Allow	Write Members	Group Objects
Allow	Create Group objects	This object and all child objects

Active Administrator™

This Active Template is applied to 1 Organizational Unit(s). Modifying this template will affect the permissions of these OU(s). Do you want to continue?

Yes No

Save Template   Save as New Template   Cancel

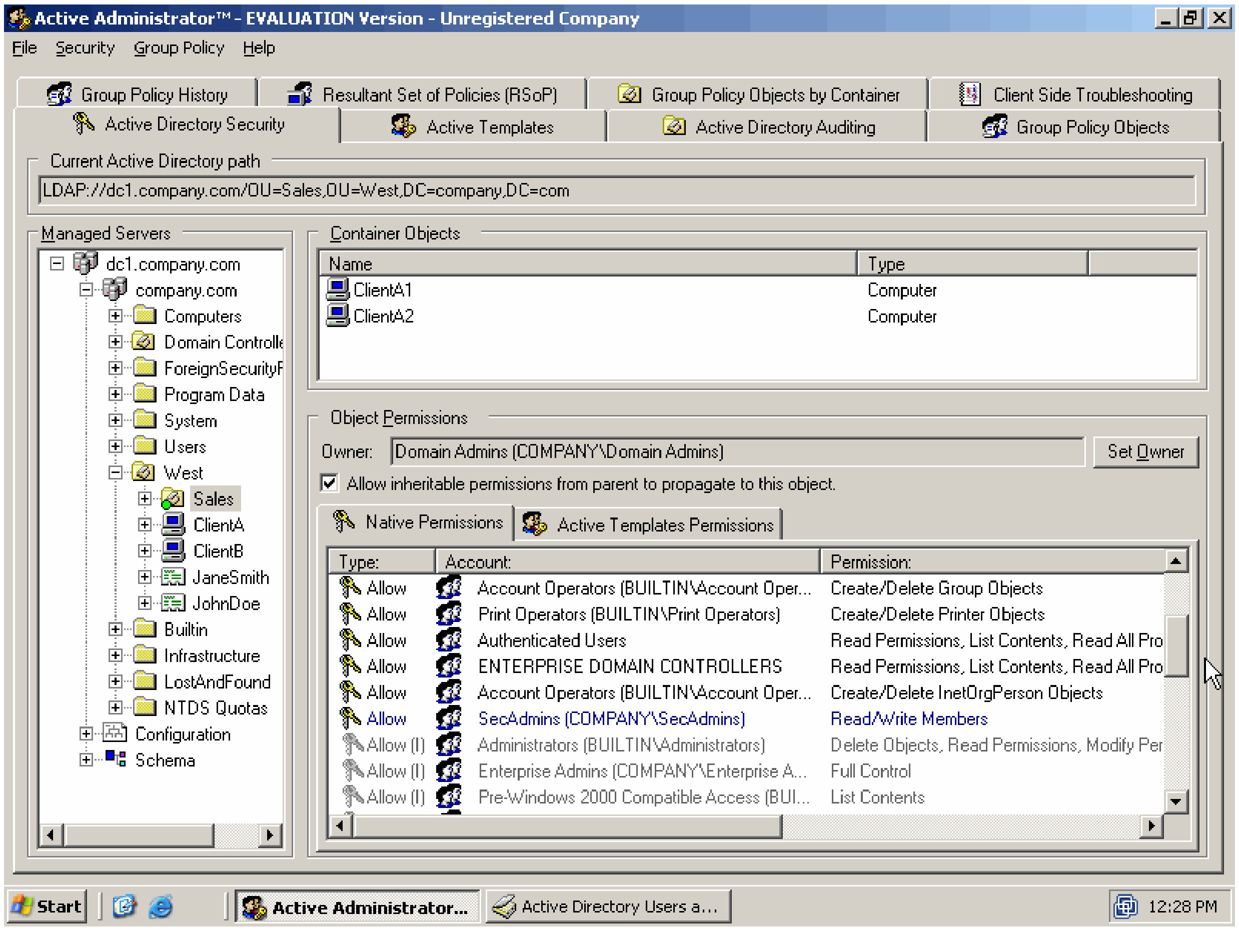
Start   Active Administrator...   Active Directory Users a...   Wednesday, March 30, 2005   12:29 PM

Figure 4.10: Changing a template changes its application to existing objects.

This solution is a powerful, template-based means of managing AD security. You might create templates that reflect the capabilities certain groups or users should have within your environment; as those capabilities change, you simply modify the templates and permissions are updated accordingly. You can begin to move away from direct (or *native*) AD permissions assignment and begin to work entirely with template-based permissions, improving the efficiency of your overall administration and making delegation of permissions much easier.

Security Reporting

In addition to maintaining consistency, security reporting is a key factor in secure AD administration. As I previously mentioned, AD permissions are too complex to easily view in a graphical user interface (GUI). To address this problem, you can again turn to a third-party tool that offers a clear interface to view the native permissions on any AD object, as Figure 4.11 shows.



Current Active Directory path: LDAP://dc1.company.com/OU=Sales,OU=West,DC=company,DC=com

Container Objects

Name	Type
ClientA1	Computer
ClientA2	Computer

Object Permissions

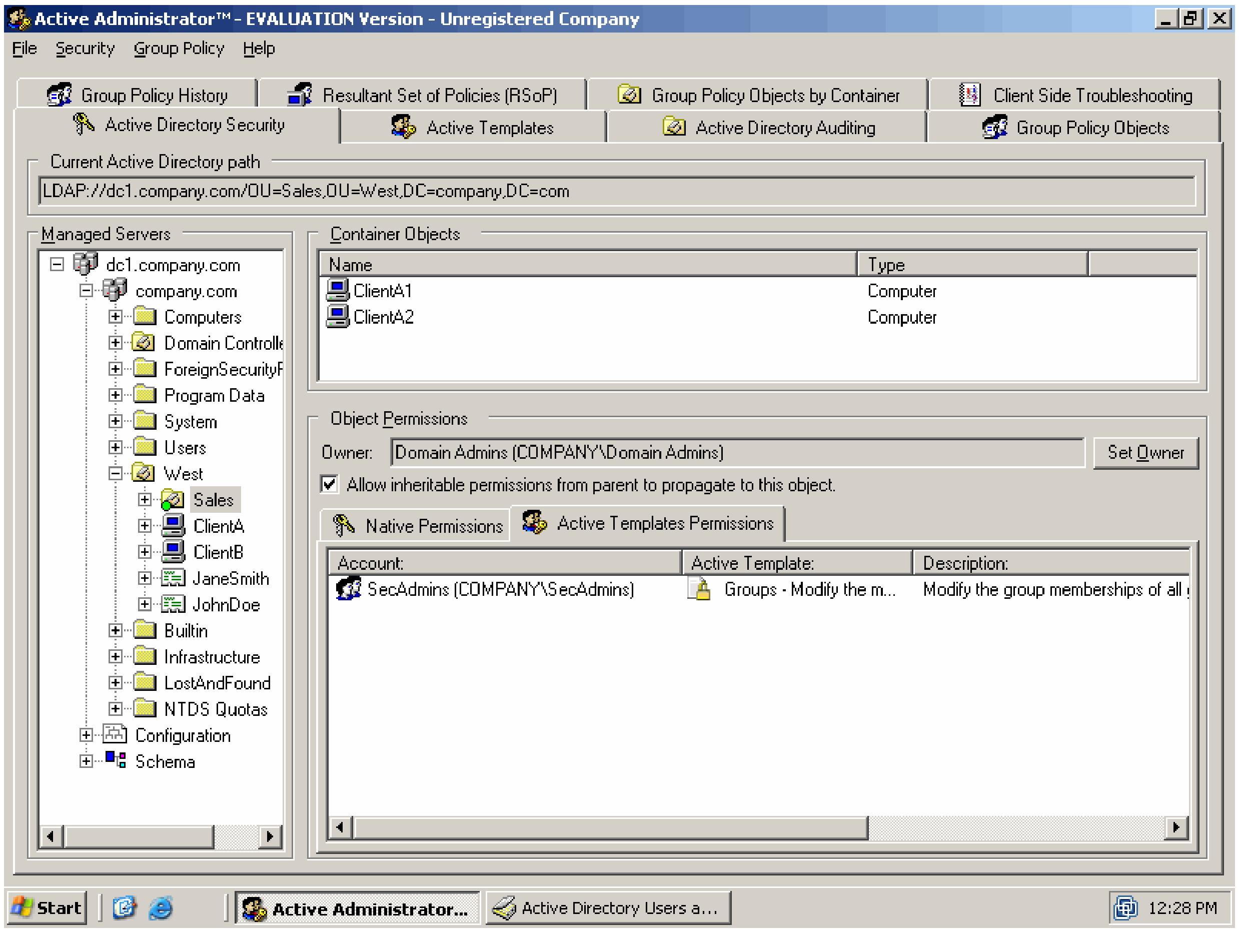
Owner: Domain Admins (COMPANY\Domain Admins)

[✓] Allow inheritable permissions from parent to propagate to this object.

Type:	Account:	Permission:
Allow	Account Operators (BUILTIN\Account Oper...	Create/Delete Group Objects
Allow	Print Operators (BUILTIN\Print Operators)	Create/Delete Printer Objects
Allow	Authenticated Users	Read Permissions, List Contents, Read All Pro
Allow	ENTERPRISE DOMAIN CONTROLLERS	Read Permissions, List Contents, Read All Pro
Allow	Account Operators (BUILTIN\Account Oper...	Create/Delete InetOrgPerson Objects
Allow	SecAdmins (COMPANY\SecAdmins)	Read/Write Members
Allow (I)	Administrators (BUILTIN\Administrators)	Delete Objects, Read Permissions, Modify Per
Allow (I)	Enterprise Admins (COMPANY\Enterprise A...	Full Control
Allow (I)	Pre-Windows 2000 Compatible Access (BUI...	List Contents

Figure 4.11: Viewing permissions in Active Administrator.

As this figure shows, inherited permissions are tagged with an (I) indicator and listed in gray; permissions that are the result of a template application are shown in bright blue. This screen makes it much easier to view and report on AD permissions because the permissions are displayed in a very effective manner. Figure 4.12 shows that template applications can be viewed separately, allowing you to quickly note which security templates are in effect on the AD object you're examining.



LDAP://dc1.company.com/OU=Sales,OU=West,DC=company,DC=com

Name	Type
ClientA1	Computer
ClientA2	Computer

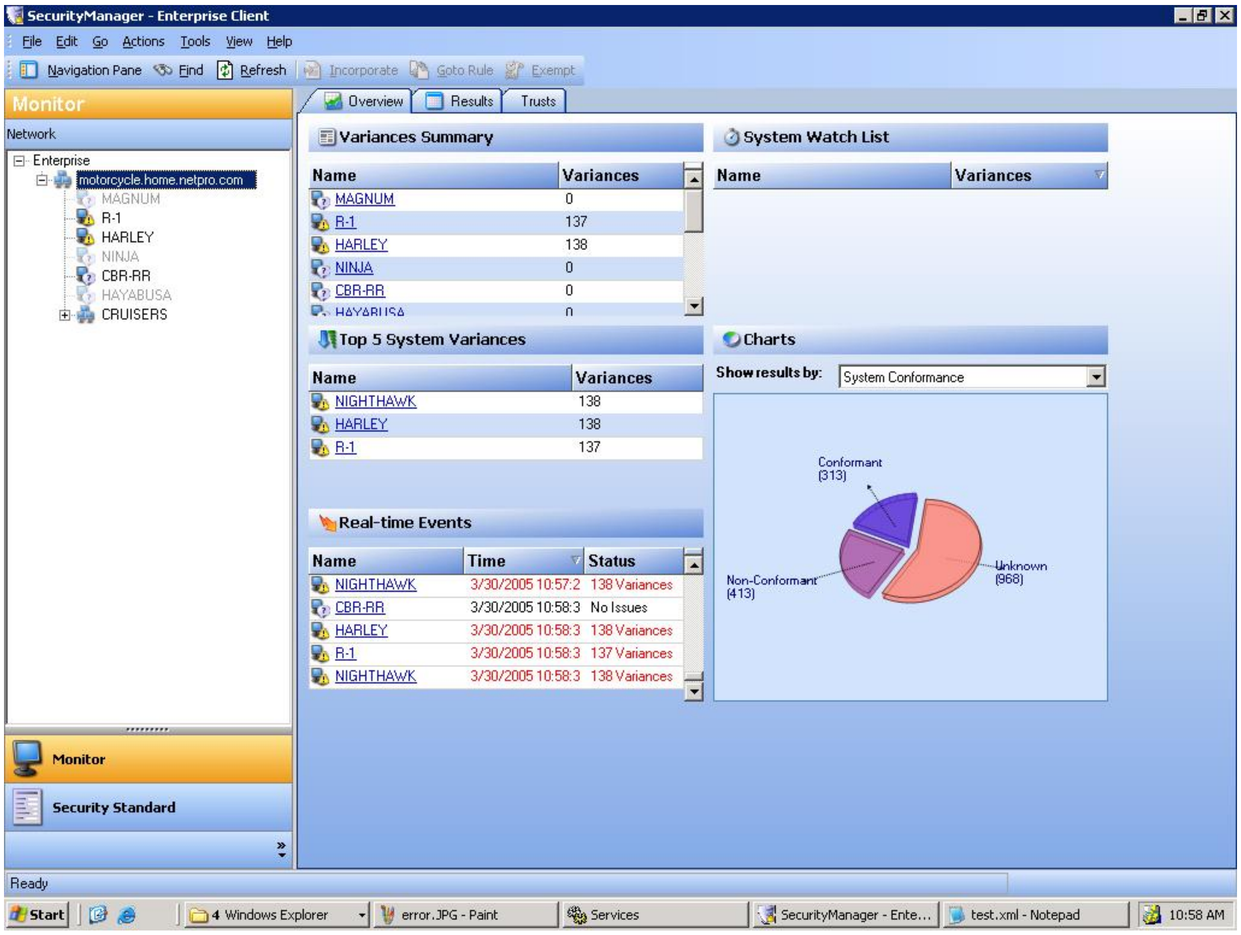
Owner: Domain Admins (COMPANY\Domain Admins)

☑ Allow inheritable permissions from parent to propagate to this object.

Account:	Active Template:	Description:
SecAdmins (COMPANY\SecAdmins)	Groups - Modify the m...	Modify the group memberships of all

Figure 4.12: Examining template application.

Another valuable reporting tool is NetPro SecurityManager. It can be used to get a quick snapshot of your security situation (see Figure 4.13). The domain examined in the screen shot has 413 non-conformant issues, with those variances broken down per system in the Variances Summary section. This tool, preloaded with several best-practice configuration settings, allows you to quickly spot potential security issues and deal with them appropriately.



Variances Summary

Name	Variances
MAGNUM	0
R-1	137
HARLEY	138
NINJA	0
CBR-RR	0
HAYABUSA	0

System Watch List

Name	Variances

Top 5 System Variances

Name	Variances
NIGHTHAWK	138
HARLEY	138
R-1	137

Real-time Events

Name	Time	Status
NIGHTHAWK	3/30/2005 10:57:2	138 Variances
CBR-RR	3/30/2005 10:58:3	No Issues
HARLEY	3/30/2005 10:58:3	138 Variances
R-1	3/30/2005 10:58:3	137 Variances
NIGHTHAWK	3/30/2005 10:58:3	138 Variances

Figure 4.13: Security snapshot from SecurityManager.

The purpose of any AD reporting tool is to simply help aggregate and filter the immense amount of security information available under AD. Whether the tool is consolidating and filtering AD auditing events, AD security settings, or a broader range of AD configuration settings, reporting allows you to view, in one place, a larger amount of information than AD's native tools allow you to do. Other useful AD reporting tools include:

- Imanami SmartR—This flexible tool allows you to produce a variety of custom reports. You can quickly find users based on their last login date, see who owns particular groups, and so forth.
- Javelina Software AD Toolkit—This tool provides several built-in reporting capabilities, many of which are useful for cleaning up security problems (such as inactive computer accounts and disabled accounts), as well as offers group membership reports and more.
- Quest Enterprise Directory Reporter—This tool is a comprehensive reporting solution intended to help you determine which users have administrative rights, what accounts are unused, how many users have been created in the past month, and so on.
- ScriptLogic Active Administrator—In addition to providing configuration and maintenance capabilities, this tool provides reporting on your existing AD configuration.

These and other reporting tools can help you gain better insight into your AD security situation, spot potential problems, and more easily find areas that need your attention.

Overlooked (and Ignored) Security Problems

Perhaps one of the best things you can do for AD, security-wise, is keeping up-to-date on Windows service packs and patches. That's hardly news—patch management is a hot topic these days. But most organizations also have some easily overlooked and ignored security problems that can be a lot more dangerous than being behind on your patches.

Unnecessary Security Principles

Any security account—whether it's a user or a computer—shouldn't exist unless it's being actively used for a defined purpose. Old user accounts should be disabled for a fixed period of time and deleted as soon as is practicable. Any security principle (such as a user or computer) can potentially be used as an attack point. Pretty much by definition, unused accounts' passwords aren't changed regularly, making them somewhat more susceptible to attack.

Built-in Accounts and Groups

I recently attended a lecture on the use of built-in accounts and groups, where the speaker proposed that built-in accounts and groups should *never* be used and should in fact have all of their built-in permissions removed. The argument was an extension of the common practice of renaming the built-in Administrator account. The theory there is that the Administrator (and Guest, for that matter) account name is well-known and is the first point of attack because an attacker only has to guess the password. The speaker's argument was that the security identifiers (SIDs) for all built-in accounts and groups are well-known, and changing the account name doesn't change the SID. Because the SID is well-known (or, in the case of a domain, can be guessed relatively easily), the accounts should be disabled, the groups should be emptied, and both should have their built-in permissions and rights removed. You should create your own accounts that have any permissions and rights you need because the SIDs of those accounts will be difficult for an attacker to predetermine.

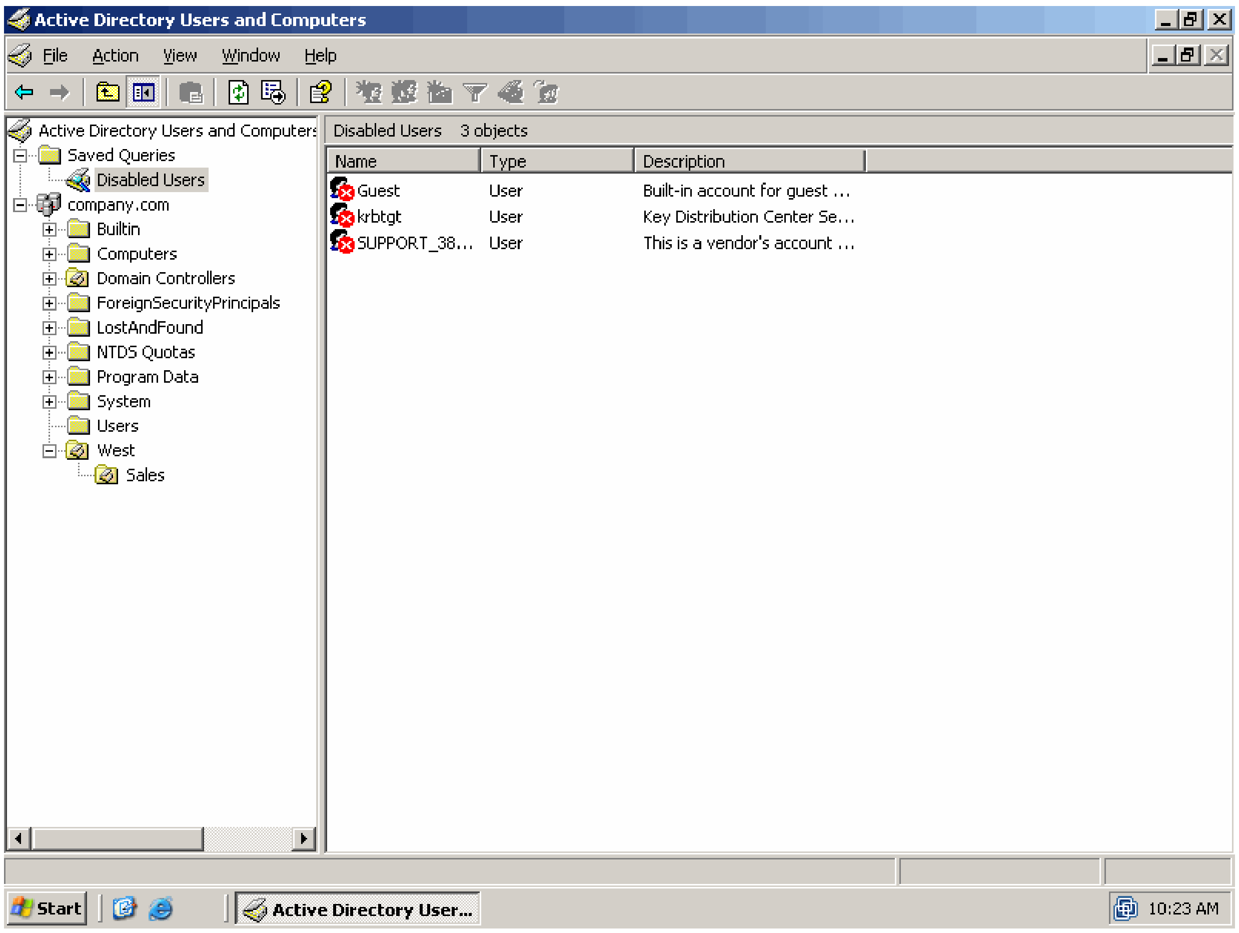
I haven't worked with any organizations that are implementing the speaker's suggestion, but it's an interesting theory. Although "security through obscurity" is no security at all, making it difficult for an attacker to know or learn anything about your environment is definitely *part* of a good overall security strategy.

Many reporting tools can help spot unused accounts. Windows itself keeps a "last logged in" date for all domain accounts, making it relatively easy to determine which accounts aren't in use. Be warned, however, that prior to WS2K3, AD didn't replicate that "last logged in" account attribute. Thus, a reporting tool must check with every domain controller to find the most recent "last logged in" date. In an all-WS2K3 domain, however (which is operating at the highest domain functional level), that attribute is replicated, so you can get an accurate last login date for any account from any one domain controller.

Disabled Accounts

Like unused accounts, disabled accounts represent an unnecessary security risk. Accounts can too easily be re-enabled, and it's likely they'll have an old password and obviously aren't being regularly used or checked. Disabled accounts are fine for a short period of time while, for example, you take ownership of the former user's files and other resources, but disabled accounts should be scheduled for deletion as soon as possible.

In this case, any WS2K3 domain controller (even if your domain is not entirely WS2K3) can provide you with a list of disabled accounts. Simply use the "Saved Queries" functionality of the WS2K3 AD Users and Computers console. As Figure 4.14 shows, you can easily construct a query that shows all disabled user accounts, allowing you to review whether those accounts can be deleted.



Disabled Users 3 objects

Name	Type	Description
Guest	User	Built-in account for guest ...
krbtgt	User	Key Distribution Center Se...
SUPPORT_38...	User	This is a vendor's account ...

Figure 4.14: Viewing disabled user accounts.

The Saved Queries function can also provide you with "last logged in" dates, but will only be accurate in an all-WS2K3 domain.

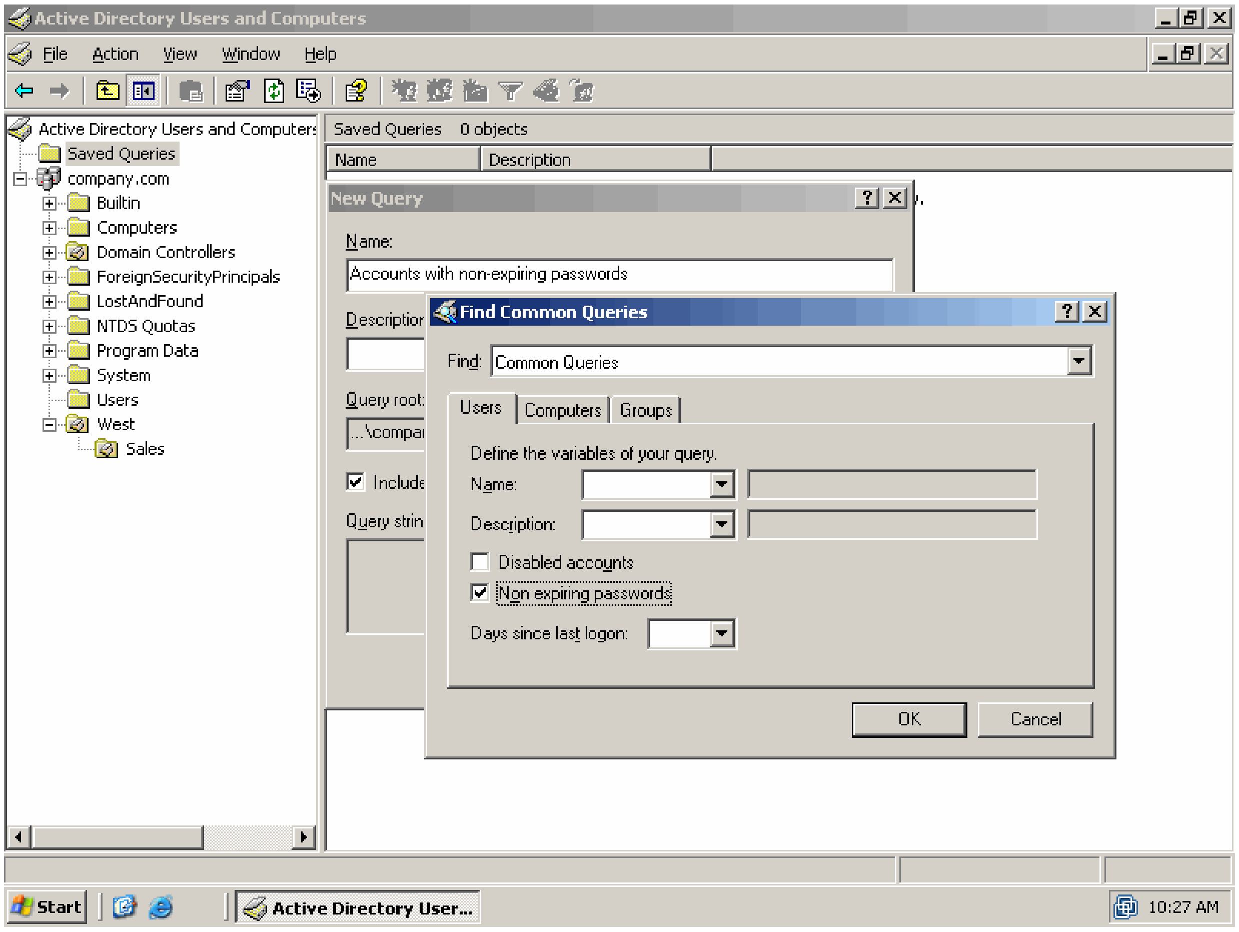
If a disabled account must remain in the domain, be sure to periodically change its password to something long and complex (a pass *phrase*, rather than a pass *word*, for example). Doing so will help prevent the account from being compromised in the event it becomes re-enabled. You should also document accounts that must remain in the domain and will be disabled so that you can periodically check them to ensure they *remain* disabled.

Don't forget about computer accounts, which can also be disabled and which represent as much a security risk as user accounts.

Accounts with Non-Expiring or Old Passwords

AD Users and Computers' Saved Queries function can also help find accounts with non-expiring passwords. You can use it to quickly define a query that spots non-expiring passwords, which are a major security risk (see Figure 4.15).

Most non-expiring passwords are for service accounts. Chapter 6 will address ways in which service accounts can be made more secure, including making it easier to regularly change their passwords.


Name: Accounts with non-expiring passwords

Find: Common Queries

Define the variables of your query.

Disabled accounts [ ]

Non expiring passwords [✓]

Figure 4.15: Querying for non-expiring user accounts.

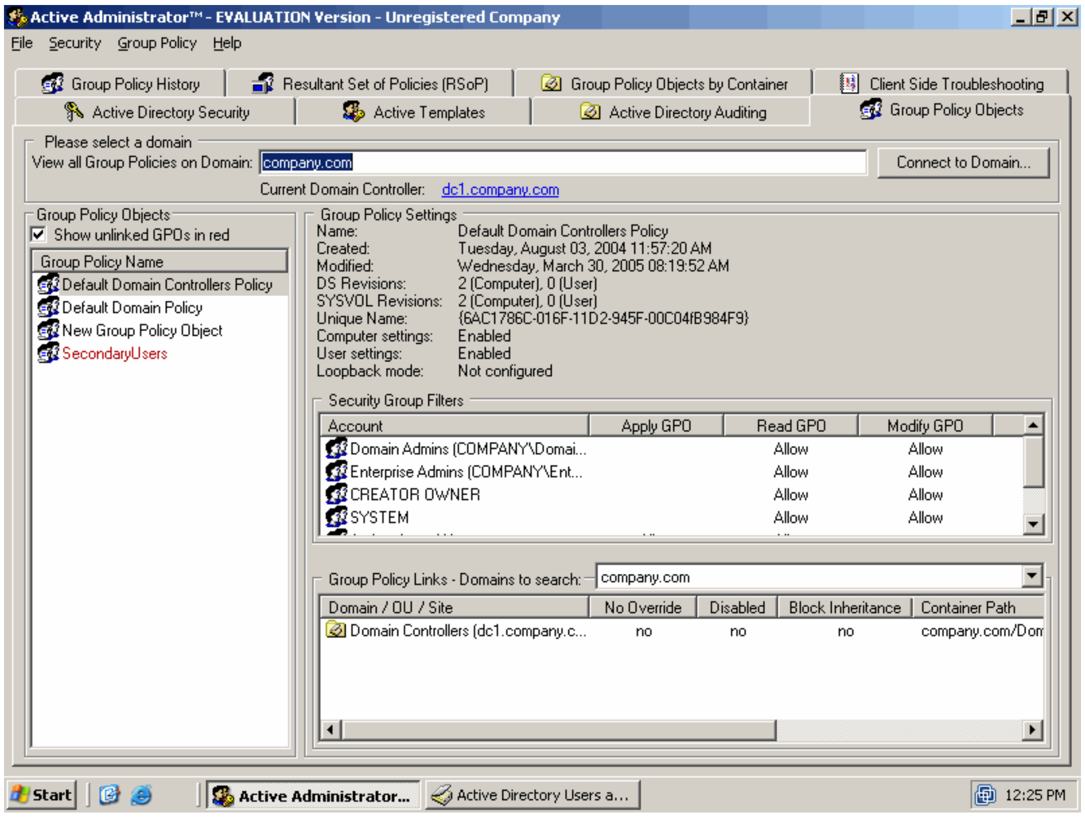
To query for *old* passwords, you'll probably need to use a third-party reporting tool, such as one of the ones mentioned earlier. If you have a maximum password age set in your domain, locating accounts with passwords older than the maximum age is a good way to spot accounts that aren't being regularly used. Contact the person supposedly using the account, if possible, and find out why the account isn't being used or why the password hasn't changed.

Don't forget about computer accounts! While computer account passwords are automatically changed by the computer owning the account, finding accounts with non-expiring passwords—something an administrator can configure—will help spot accounts that could potentially be a security risk.

Unlinked GPOs

Another potential security problem—more from the a "security of your operations" standpoint than a "loss of data" standpoint—is unlinked GPOs in the domain. These don't often occur, but when you unlink a GPO from the last container to which it was linked, you can leave the underlying GPO in place. The result is an "orphan" GPO, one which sits on every domain controller and contains configuration settings that could potentially disrupt your environment were someone to relink or mislink it. Although this occurrence isn't common, an overriding part of a good security philosophy is to not have anything lying around unless it's actually being used, so you should get rid of orphaned GPOs.

Fortunately, ScriptLogic Active Administrator makes it easy to spot them. As Figure 4.16 shows, the tool can quickly locate any GPOs that exist but aren't being utilized; in this example, the SecondaryUsers GPO is highlighted in red, indicating that it isn't in use.


Active Administrator™ - EVALUATION Version - Unregistered Company
File
Security
Group Policy
Help
Group Policy History
Resultant Set of Policies (RSoP)
Group Policy Objects by Container
Client Side Troubleshooting
Active Directory Security
Active Templates
Active Directory Auditing
Group Policy Objects
Connect to Domain...
Start
Active Administrator...
Active Directory Users a...
12:25 PM

Please select a domain
View all Group Policies on Domain: company.com
Current Domain Controller: dc1.company.com

Group Policy Objects
☑ Show unlinked GPOs in red

Group Policy Name
Default Domain Controllers Policy
Default Domain Policy
New Group Policy Object
SecondaryUsers

Group Policy Settings

Name:	Default Domain Controllers Policy
Created:	Tuesday, August 03, 2004 11:57:20 AM
Modified:	Wednesday, March 30, 2005 08:19:52 AM
DS Revisions:	2 (Computer), 0 (User)
SYSVOL Revisions:	2 (Computer), 0 (User)
Unique Name:	{6AC1786C-016F-11D2-945F-00C04fB984F9}
Computer settings:	Enabled
User settings:	Enabled
Loopback mode:	Not configured

Security Group Filters

Account	Apply GPO	Read GPO	Modify GPO
Domain Admins (COMPANY\Domai...		Allow	Allow
Enterprise Admins (COMPANY\Ent...		Allow	Allow
CREATOR OWNER		Allow	Allow
SYSTEM		Allow	Allow

Group Policy Links - Domains to search: company.com

Domain / OU / Site	No Override	Disabled	Block Inheritance	Container Path
Domain Controllers (dc1.company.c...	no	no	no	company.com/Dom

Figure 4.16: Spotting unlinked GPOs.

Summary

AD comes out of the box configured as a fairly secure system. Most companies do the right thing on their initial deployment and have a pretty secure initial state. Over time, however, the sheer complexity and flexibility of AD sets in and can easily create a less-than-ideal security situation. The key is to remember a few easily overlooked security risks, such as poorly understood permissions inheritance, unused security principles, and so forth. Another key is to use third-party tools to supplement AD's lackluster built-in security tools. The third-party tools introduced in this chapter can make permissions management easier, highlight security problems you may not have known existed, and provide you with more readable and usable reports on the status of your directory's security.

The next chapter will step away from AD a bit and talk about file server security. File servers are probably the most abundant type of Windows server in any environment, and they have several easily overlooked security problems that, if left uncorrected, can undermine your security efforts across the enterprise.