

realtimepublishers.com[®]

The Definitive Guide[™] To

Securing Windows in the Enterprise

SCRIPTLOGIC

Don Jones

Chapter 6: Securing Servers and Services127

Lock it Down127

 Security for Mere Mortals.....128

 How Big Is This Problem?.....129

 Running SCW130

 The Results.....143

 Beyond the SCW.....144

Service Security144

Access Controls149

 Finding Permissions.....150

 Permissions Reporting153

 Backing Up Permissions154

Summary155

Copyright Statement

© 2005 Realtimedpublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimedpublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

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

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

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

Realtimedpublishers.com and the Realtimedpublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

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

[**Editor's Note:** This eBook was downloaded from Content Central. To download other eBooks on this topic, please visit <http://www.realtimepublishers.com/contentcentral/>.]

Chapter 6: Securing Servers and Services

The previous chapter showed you ways to make securing file servers a bit easier; this chapter will focus on security for all servers: Web servers, domain controllers, application servers, database servers, and more. Of course, all of these topics apply to file servers, too.

The crucial consideration to recognize about any server—regardless of OS, although I'll focus on Windows—is that the OS contains bugs. Some of those bugs will create security vulnerabilities. Thus, the best practice for any server's security is to try to shield those potential bugs and vulnerabilities, which is much of this chapter will focus on.

Lock it Down

Many administrators spend hours locking down their servers—removing unnecessary services and applications, blocking unnecessary TCP and UDP ports from external access, and so forth. As the Windows OS has matured, these tasks have become easier for administrators. For example, Windows 2000 (Win2K) Server includes the Internet Connection Firewall (ICF), which allows you to block access to unnecessary ports if it is difficult to disable or uninstall the software that is using these ports. As Figure 6.1 shows, Windows Server 2003 (WS2K3) Service Pack 1 (SP1) upgrades the ICF to the full Windows Firewall, providing more granular control and the ability to open ports based on tasks (such as file sharing) rather than forcing you to look up port numbers.

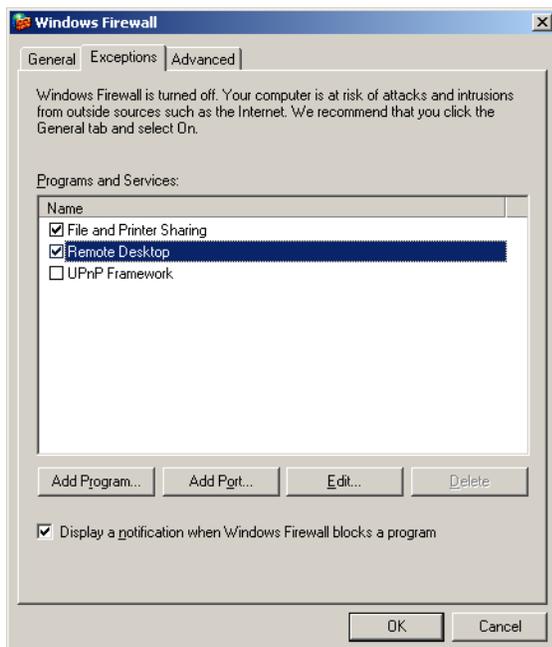


Figure 6.1: Configuring the Windows Firewall.

Unfortunately, all of Windows' new capabilities are significantly hindered by the fact that much of Windows functionality requires multiple services to be running, necessitates multiple ports be opened, and are too poorly documented for administrators to accurately lock them down. Over the years, Microsoft has amassed a series of articles (such as the one at <http://support.microsoft.com/default.aspx?scid=kb;en-us;287932> that covers Microsoft SQL Server) that attempt to detail the ports needed for various services; however, these efforts have never really come together into one comprehensive resource.

Security for Mere Mortals

Such was the case until WS2K3 SP1, which introduces the new Security Configuration Wizard. SCW is, in many respects, one of the best arguments for upgrading at least one server in your environment to WS2K3 SP1 (you must upgrade to SP1 in order to access SCW). As Figure 6.2 shows, SCW is based upon a set of XML-formatted configuration files that detail the various OS dependencies associated with specific server tasks or roles.

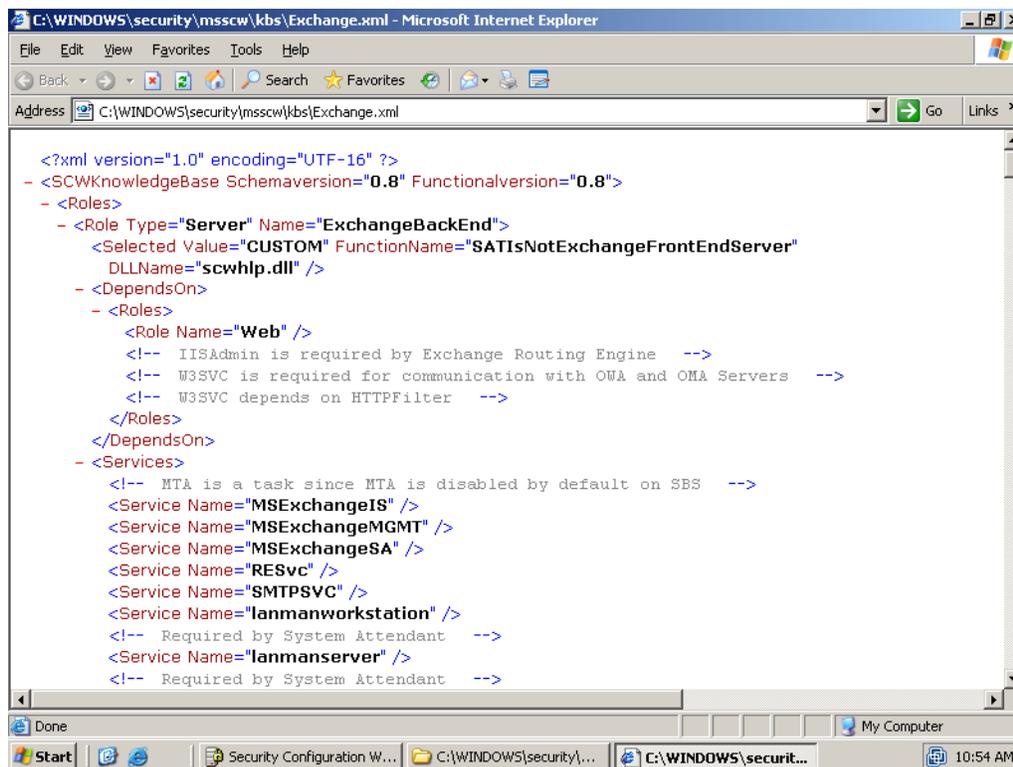


Figure 6.2: The SCW XML configuration file.

In Figure 6.2, for example, you can see a role named ExchangeBackEnd, which is an Exchange Server back-end server. This role has several dependencies: It depends on the presence of the role named Web, for example, as well as on a set of services and a series of TCP and UDP ports. The XML shipping with WS2K3 SP1 includes role definitions for most Microsoft server products as well as for common core roles such as domain controller, file server, and so forth. This XML file finally consolidates all of Microsoft's previously scattered knowledge about which applications need which services and ports in order to function properly.

Because the XML file is well-documented and fairly easy to understand, you can easily update it to include dependency information for your own internal applications, and other software vendors can provide configuration information for their applications, as well.

All of this functionality allows SCW to lock down servers more accurately, providing what some experts refer to as “security for mere mortals.” If you’re running WS2K3-based servers in your environment, you can finally—with some confidence—lock them down to prevent yet-to-be-discovered bugs and vulnerabilities from being as big a problem.

How Big Is This Problem?

It’s easy enough to determine which ports are currently opened by your Windows servers. Simply run

```
netstat -a
```

from a command-line (or run

```
netstat -a -o
```

to see ports and their associated processes). As Figure 6.3 shows, a freshly installed standalone server opens quite a number of ports.

```

C:\Documents and Settings\Administrator>netstat -a
Active Connections
Proto Local Address           Foreign Address         State
TCP    standalone:http         standalone:0            LISTENING
TCP    standalone:epmap       standalone:0            LISTENING
TCP    standalone:microsoft-ds standalone:0            LISTENING
TCP    standalone:1025        standalone:0            LISTENING
TCP    standalone:ms-sql-s    standalone:0            LISTENING
TCP    standalone:ms-olap3    standalone:0            LISTENING
TCP    standalone:ms-sql-m    standalone:0            LISTENING
TCP    standalone:netbios-ssn standalone:0            LISTENING
UDP    standalone:microsoft-ds *:*
UDP    standalone:isakmp      *:*
UDP    standalone:1027        *:*
UDP    standalone:ms-sql-m    *:*
UDP    standalone:ipsec-msft  *:*
UDP    standalone:ntp         *:*
UDP    standalone:1032        *:*
UDP    standalone:ntp         *:*
UDP    standalone:netbios-ns  *:*
UDP    standalone:netbios-dgm *:*
C:\Documents and Settings\Administrator>

```

Figure 6.3: Checking open ports on a Windows server.

It is a good practice to document which ports *should* be open on each of your servers and check each server periodically to see which ports are actually open. For that task, netstat—which is designed to be run locally—isn’t the best tool. Instead, you might find an inexpensive (or even free) IP port scanner to be more effective, as it can scan entire subnets full of computers at once, reporting on open ports. Figure 6.4 shows the interface for a free scanner called Angry IP Scanner, available from <http://www.angryziber.com/ipscan/>.

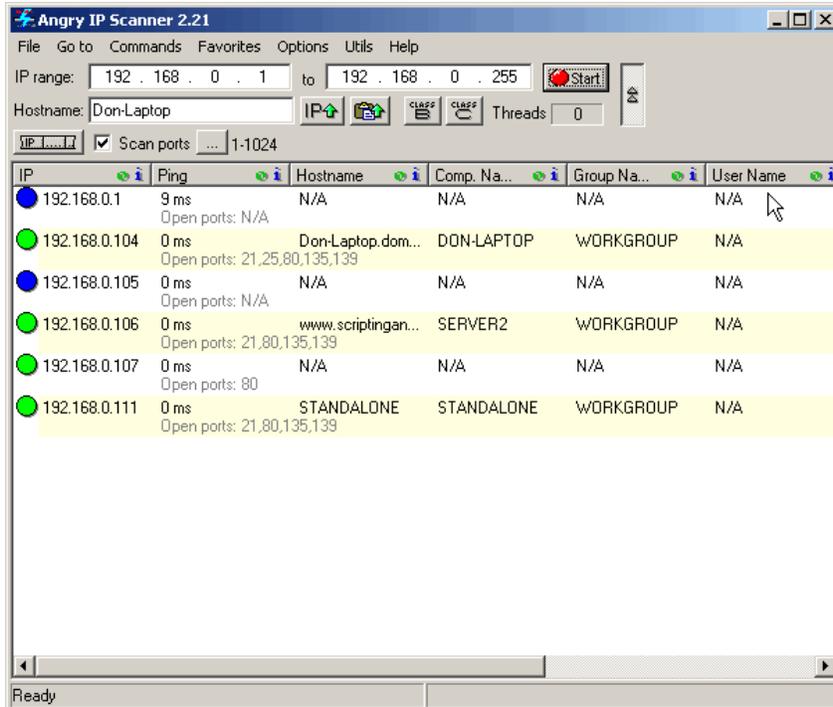


Figure 6.4: The Angry IP Scanner interface.

Getting to know which ports *are* open on your servers, as opposed to which ones *should* be open, can help you understand the criticality of open ports: Every open port represents a potentially undiscovered security vulnerability. Although the tasks performed by a server certainly need to be accessible to clients, ports not related to those official tasks should be locked down to help prevent them from becoming a vulnerability.

Running SCW

SCW starts as Figure 6.5 shows, by asking you which computer will be the baseline for a new policy. SCW is designed to create multiple policies, with one policy assigned to each similar class of computers. For example, if you have five Web servers that perform substantially the same tasks, one of those could serve as your baseline for the Web server category of server.

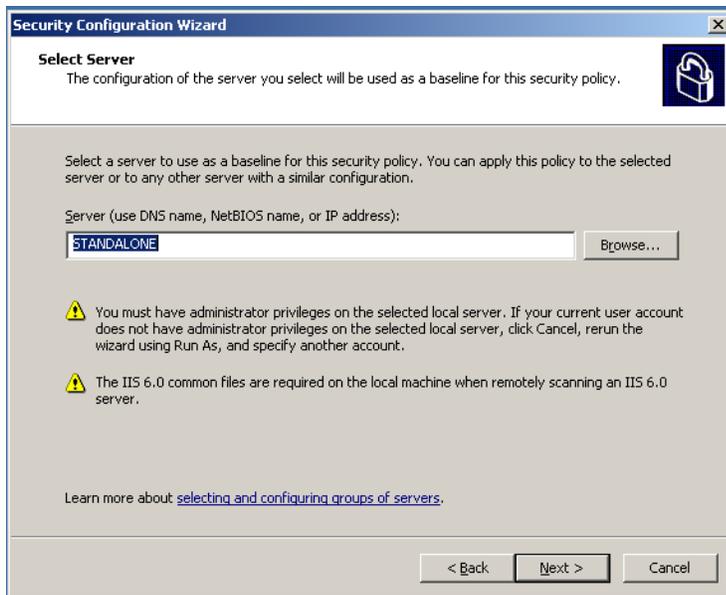


Figure 6.5: Specifying a baseline server in SCW.

SCW will scan the baseline server and present a screen similar to the one that Figure 6.6 shows, listing all the *roles* currently installed on that server. A *role* is a business-level set of tasks, such as Web server, Telnet server, SQL Server, print server, and so forth. Notice that not all of the roles are selected, meaning the software supporting the role—such as print server—is installed, but the role isn’t enabled or in use (perhaps meaning no printers are actually shared).

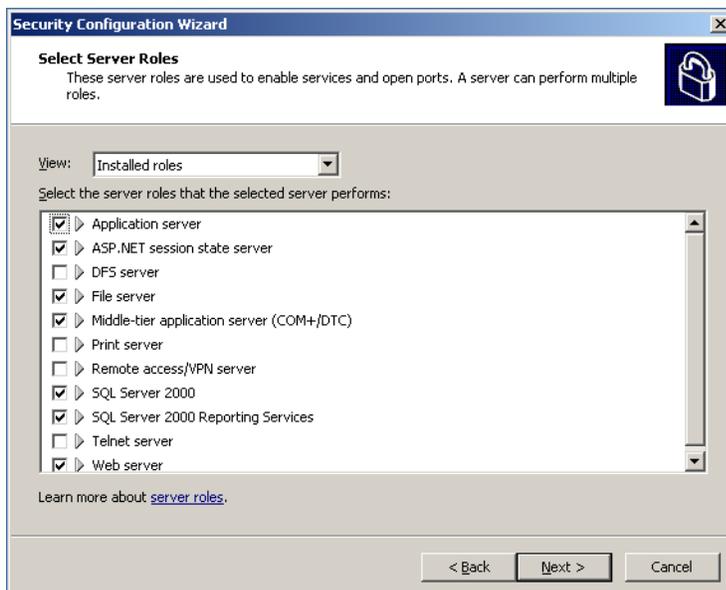


Figure 6.6: Listing currently installed roles in SCW.

By making a different selection from the View drop-down list box, you can instead view currently selected roles (as Figure 6.7 shows) or all available roles (see Figure 6.8).

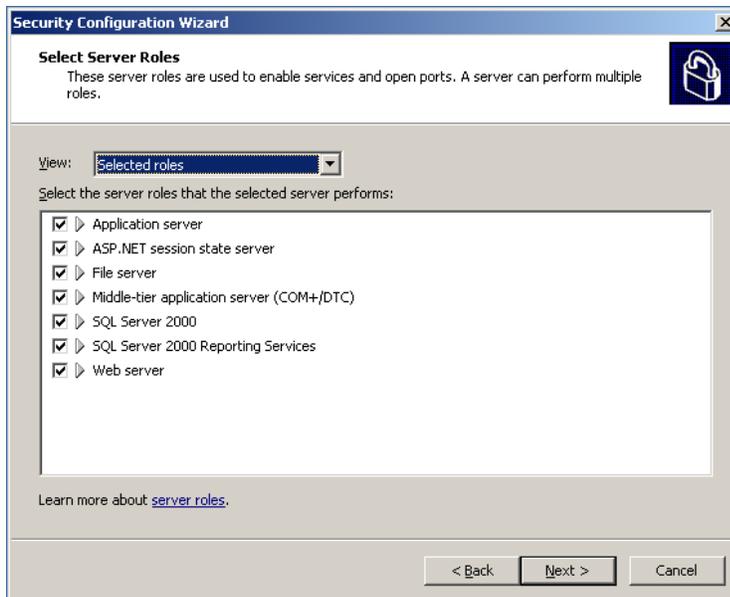


Figure 6.7: Viewing selected roles.

Viewing selected roles allows you to work with only the roles that are installed and in active use on the server; as any unselected roles aren't in use, it is most likely safe to disable or uninstall them without negatively impacting your production environment. In contrast, by viewing *all* available roles, you can potentially use SCW to prepare a server for future roles that you know it will hold, but doesn't currently. For example, if you know the server is going to become a member of a cluster in the future, you can prepare it for that role now, even though the necessary software hasn't yet been installed.

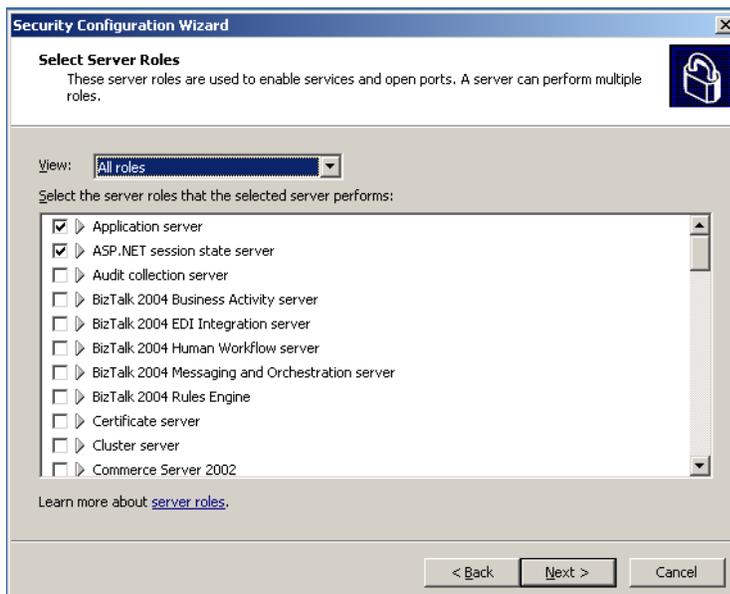


Figure 6.8: Viewing all available roles.

 The all roles view lists all roles defined in SCW's XML configuration file.

SCW has a great deal of intelligence built in with regard to role dependencies. For example, the SQL Server 2000 Reporting Services role has a dependency on the Web server role. As Figure 6.9 illustrates, if I clear the Web server role check box—thus disabling it and blocking access to it—the SQL Server 2000 Reporting Services role check box automatically clears because it can't function without the now-unavailable Web server role.

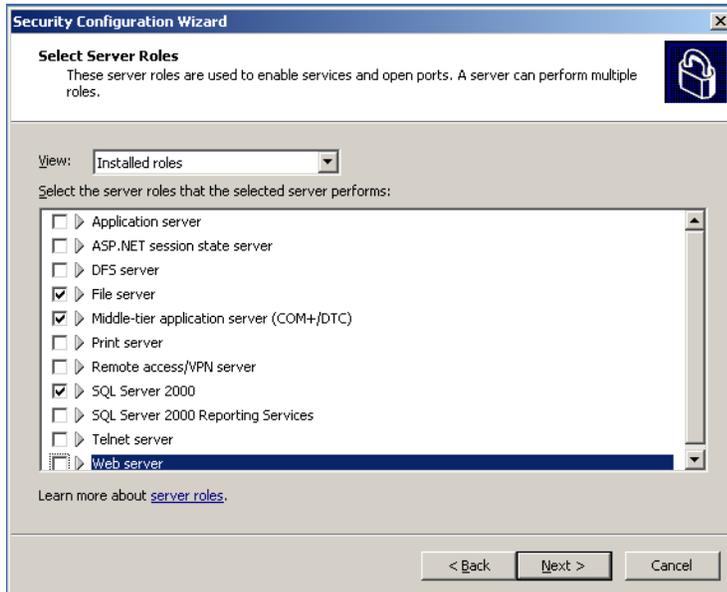


Figure 6.9: Automatic support for interdependent roles.

This support for role dependencies allows you to operate SCW with more confidence. If you disable a role because you think you don't need it, SCW will respond by disabling any dependent roles automatically. This action informs you that you did, in fact, need the role you disabled and prevents you from making a mistake.

Servers are also clients, so SCW next allows you to work with installed client features (see Figure 6.10). Again, you can choose to work with installed features, selected features, or all possible features, just as when working with server roles. In this step, you will decide whether the server will support Dynamic Host Configuration Protocol (DHCP), DNS, automatic updates, and so forth. Clear the check boxes of client software that isn't used by the server.

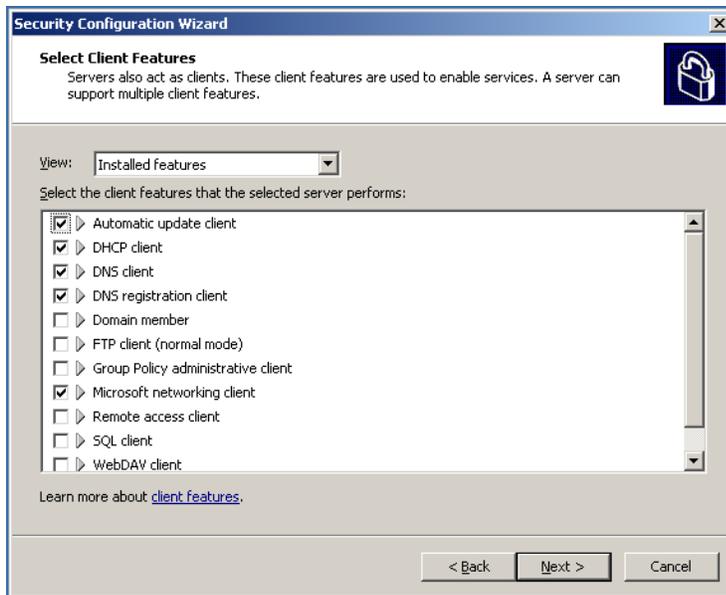


Figure 6.10: Selecting client features in SCW.

SCW is also aware of administrative capabilities and other services, such as Alerter, audio services, and so forth. By selecting the appropriate options, you can ensure that the server is properly configured and that the software needed to administer or maintain the server will be enabled. Again, you can choose to work from a list of installed options (as Figure 6.11 shows), only those options currently in use, or all available options recognized by SCW.

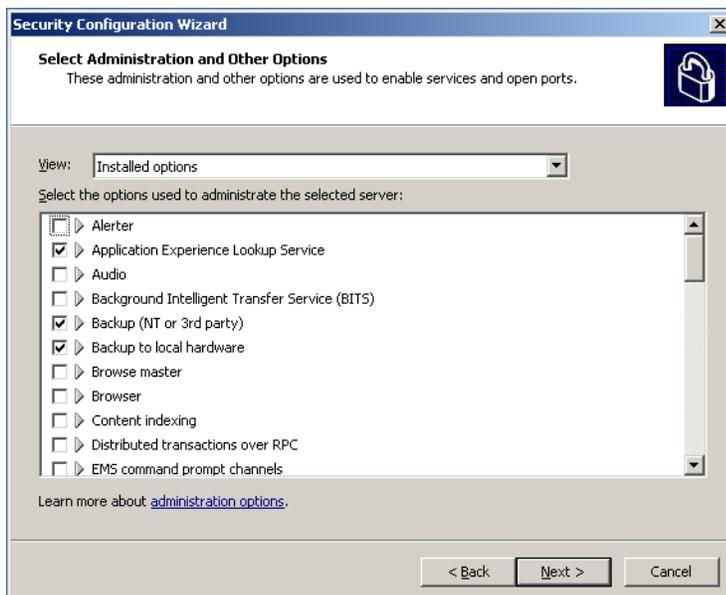


Figure 6.11: Selecting administrative and maintenance options.

SCW may also detect other services that don't fit into the categories of server roles, client abilities, or administration and maintenance options. These last services will be displayed in a list (see Figure 6.12). For this list, you will often need to conduct extra research to determine what these services are doing, and decide whether you actually need them to be running on the server.

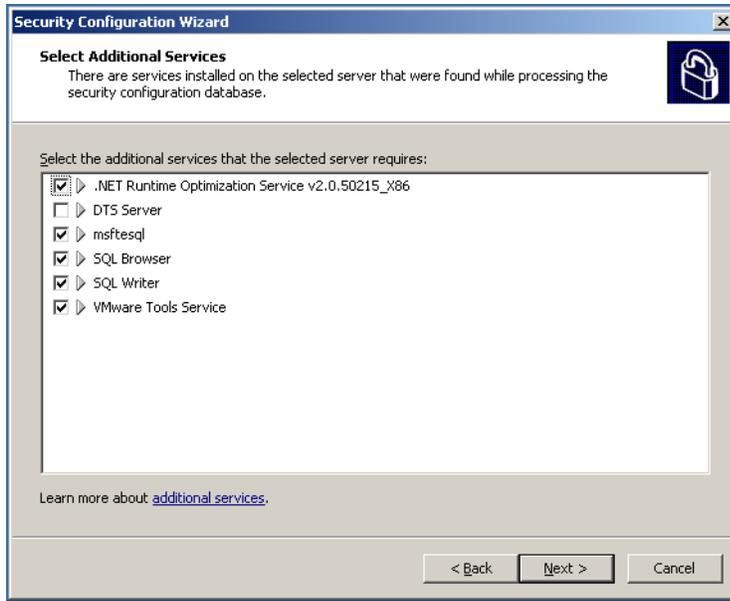


Figure 6.12: Configuring additional services in SCW.

Next, SCW will ask how you want to handle any services that haven't been explicitly selected in the prior lists. The default setting is not to change anything; however, I recommend configuring SCW's policy to disable unknown services. Doing so will handle any services currently installed or any services installed in the future, ensuring that only those roles the server is *supposed* to be fulfilling will operate. Figure 6.13 shows the selection screen.

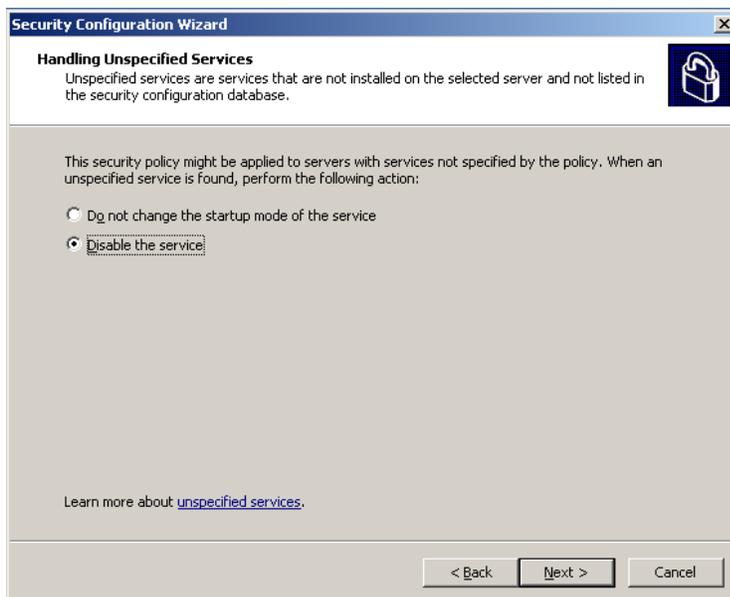


Figure 6.13: Choosing to disable unspecified services.

Remember, however, that this policy is being constructed by looking at a baseline server. Say your baseline server is a Web server, and you've configured the policy appropriately. You have four other Web servers to which you plan to apply this completed policy. If one of those servers is also acting as a domain controller (for example), selecting *Disable the service* in this step of the SCW configuration process will disable the domain controller services when the policy is applied to that server. The reason is that the baseline server wasn't a domain controller, so the domain controller services weren't specified in the policy created by SCW. Thus, you need to be sure when applying a policy to any server that the server is identical in purpose to the server that was used as the policy's baseline.

Near the end of the configuration process, SCW will display a list similar to the one that Figure 6.14 shows, detailing every service and which changes will be made by the policy. Notice the Current Startup Mode and Policy Startup Mode columns, which list the current state of each service as well as the state of each service once the policy is applied.

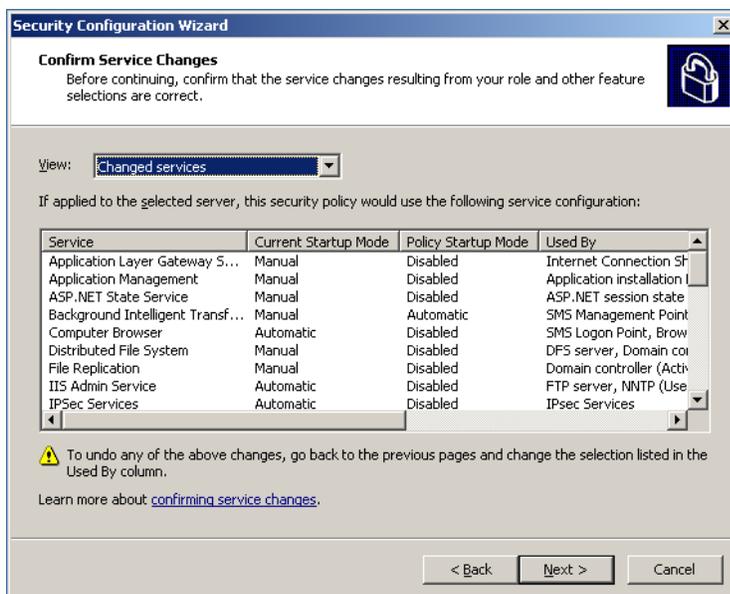


Figure 6.14: Listing the changes to each service.

Many services previously set to Manual startup will be set to Disabled once the policy is applied. This list provides you with the opportunity to perform a “sanity check” on the changes implemented by the policy. Until now, you've been dealing with roles and features, this list is the first time SCW has displayed its configuration in terms of direct changes to services. Similarly, the next screen (see Figure 6.15), provides a list of changes that will be made to TCP and UDP ports.

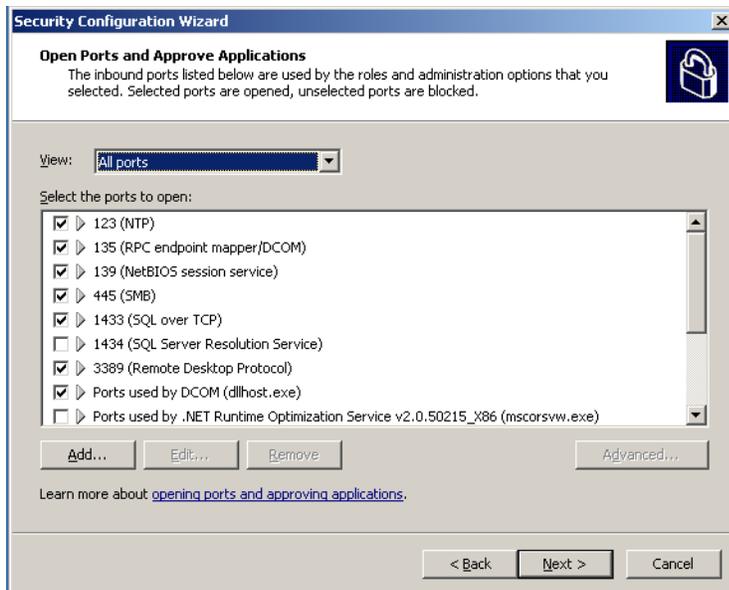


Figure 6.15: Examining port changes.

SCW will enable the Windows Firewall and will only create exceptions for ports that are selected in this list. You usually won't need to make any changes, but it is useful to document this list as the approved list of ports for the server. You can also make adjustments to the port policy. For example, double-clicking a port allows you to modify its restrictions. As Figure 6.16 illustrates, you can allow a port to be completely open, restrict access to the port to a range of IP addresses, or require IP Security (IP Sec) actions—such as encryption or authentication—to port access. For example, you might configure the policy to leave port 21 (FTP) open only to clients that can successfully establish an encrypted IPsec connection, thus protecting the FTP traffic—which transmits passwords and data in clear-text—from electronic eavesdropping.

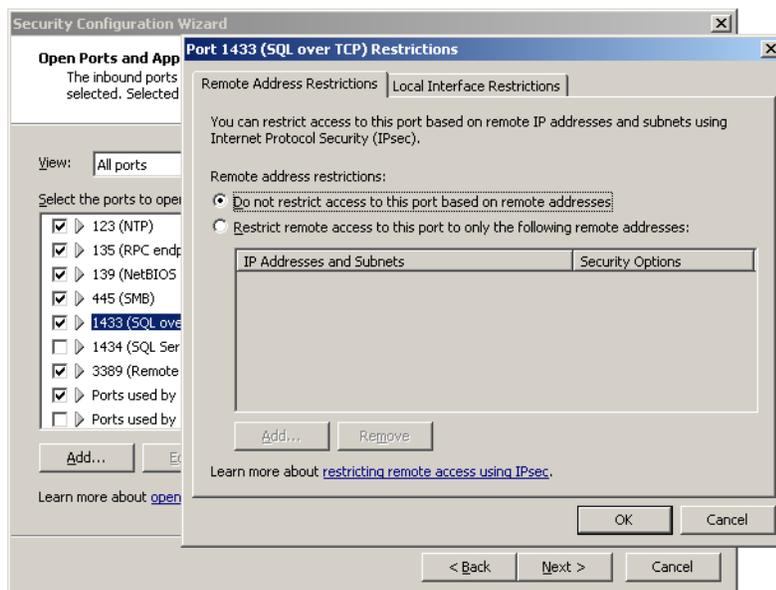


Figure 6.16: Configuring advanced port restrictions.

This additional flexibility in SCW allows you to work with complex IPSec policies in a simpler, more intuitive fashion. Many companies like the *idea* of IPSec, but quickly become intimidated when they dive into Windows' rather complex and unintuitive IPSec management interface; SCW helps make IPSec more approachable for those organizations. For example, you can restrict traffic based on a simple-to-configure drop-down box, requiring signing, encryption, or other options for incoming traffic on the specified port (see Figure 6.17).

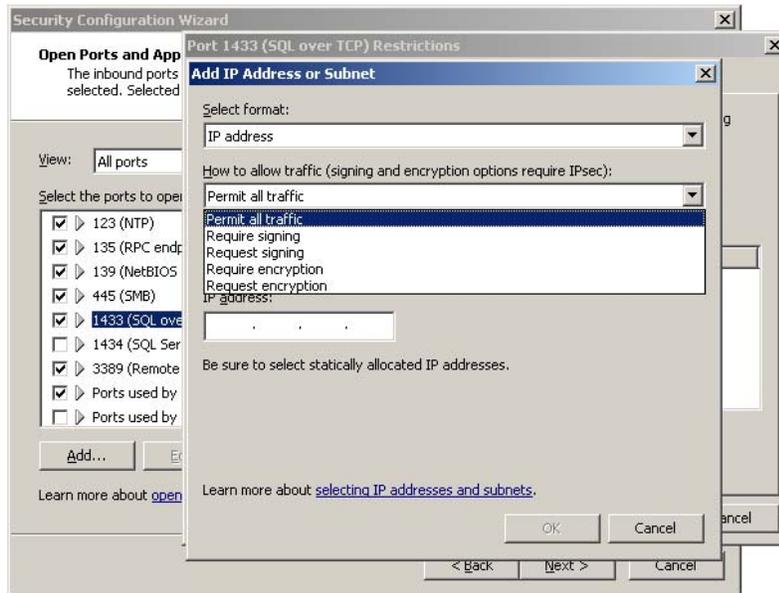


Figure 6.17: Easily configuring IPSec options within SCW.

Next, SCW tries to get a feeling for the spare processor power your server has so that it can recommend additional security options. As Figure 6.18 shows, you can specify that the server doesn't have any really old clients (*really old* being defined as Windows NT 4.0 before SP6a, Windows 95 without the Directory Services client, and so forth), and you can indicate that the server has some spare processor power. If you select both options, SCW will configure the policy to digitally sign all file and print traffic, helping to prevent traffic spoofing. This setting consumes about 20 percent extra processor power, so only select this option if your server is running at 70 percent capacity or less.

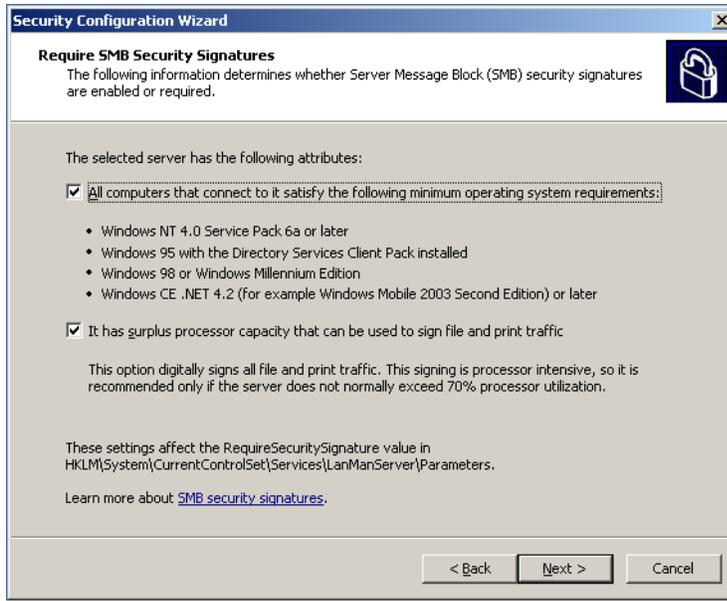


Figure 6.18: Configuring traffic signing in SCW.

SCW isn't just asking whether you want to digitally sign traffic; by asking you whether the computer has spare processor power and whether it needs to support older clients, SCW is alerting you to the *requirements* of digital signing, and helping prevent you from making a decision that isn't suitable for your environment. Similarly, as s Figure 6.19 shows, SCW will ask which types of accounts are used by the server to authenticate with remote computers. Your answer will help configure the Local Security Authority to the highest possible level of authentication.

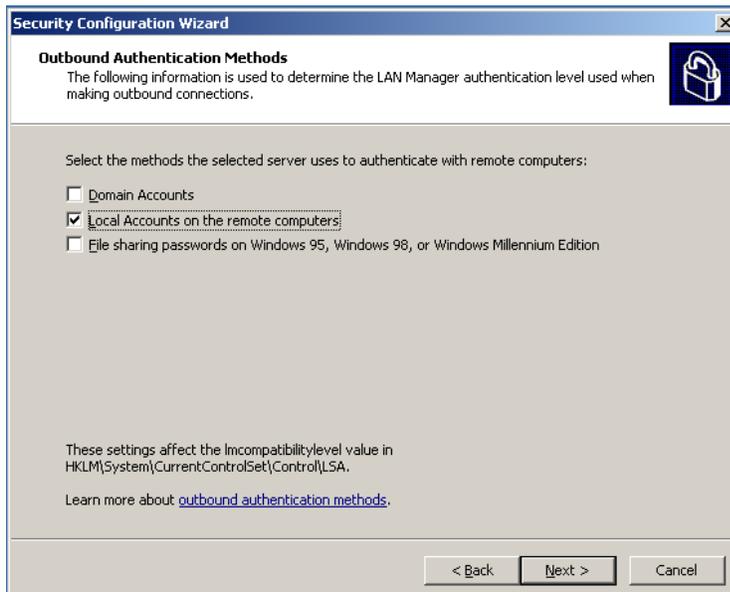


Figure 6.19: Configuring outbound authentication.

SCW isn't simply asking to which level outbound authentication should be set (which is what you would need to decide if you configured this setting in, for example, a Group Policy Object—GPO). Instead, SCW is helping you make a decision based on information you're more likely to know and understand, such as which user accounts are utilized.

 Not sure when a server would make an *outbound* connection? When the server is talking to domain controllers, for example, perhaps sending data to a network-attached printer, copying files from other servers, and so forth.

Another example of this more intuitive approach is when SCW gathers additional information to fine-tune local security (see Figure 6.20). In the previous screen, I indicated that remote local accounts are used, so SCW needs to know if those remote computers are at least running NT 4.0 SP6a and whether they synchronize their clocks (remember, Windows' Kerberos authentication requires synchronized clocks). This information helps SCW select an appropriate authentication scheme.

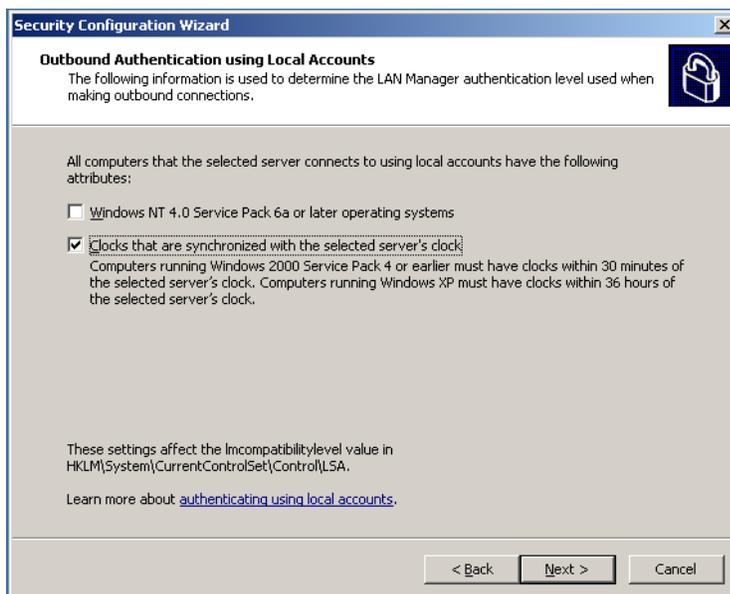


Figure 6.20: Fine-tuning the security settings.

 You might have noticed small blue text links at the bottom of many of SCW's screens. These links open a Help file to help you learn more about the settings being configured by that screen of SCW. This link is a great way to learn more about the actions that SCW is performing based on each screen's selections.

SCW isn't limited to securing ports and services, however. As Figure 6.21 shows, the prior few screens affect registry-based configuration settings.

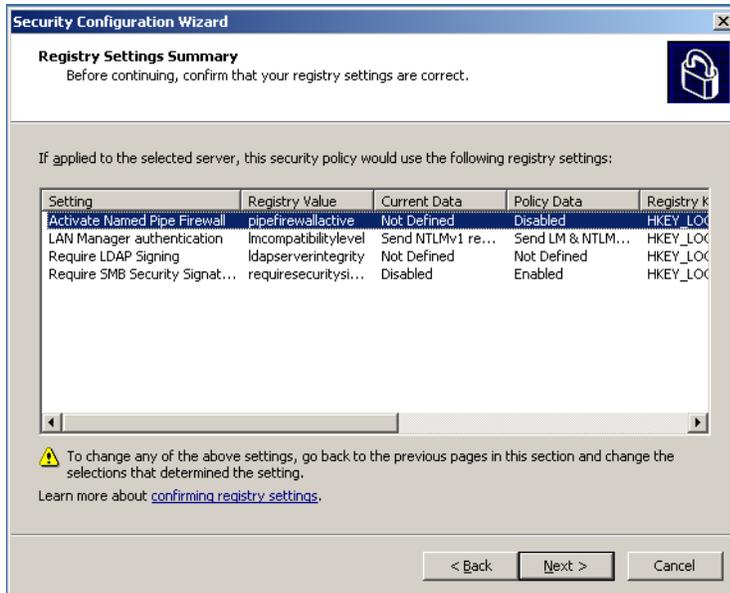


Figure 6.21: Checking registry settings in SCW.

Just as SCW allows you to review its low-level decisions in services and ports, it now allows you to review proposed changes to registry settings before continuing. This process is all part of Microsoft philosophy reflected in SCW of asking you higher-level questions, but still allowing you to review the final low-level configuration decisions.

Next, SCW will configure auditing (see Figure 6.22). The default setting is to audit successful activities, providing what Microsoft calls a *high signal-to-noise* ratio, meaning you'll get a lot of good auditing data without a lot of garbage. You can (as shown) change this setting based on your organization's requirements.

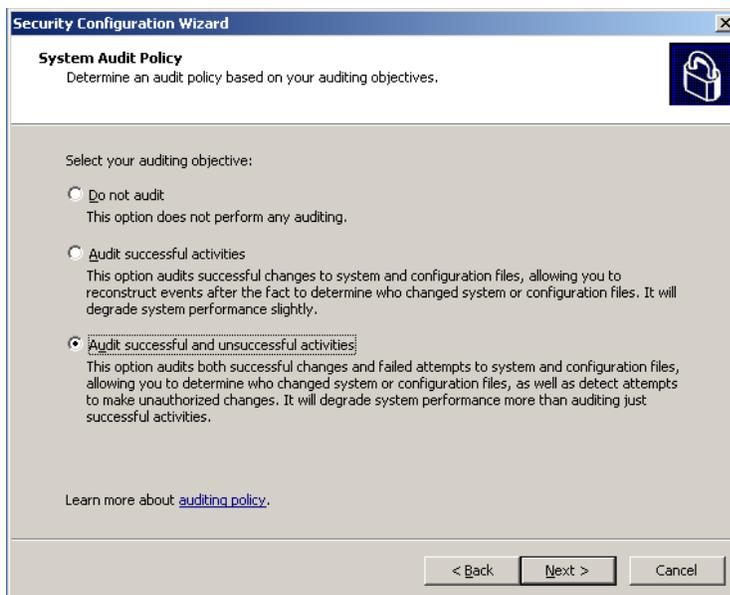


Figure 6.22: Configuring auditing.

As always, SCW takes your decision and proposes a number of configuration changes, which it then shares with you as Figure 6.23 shows.

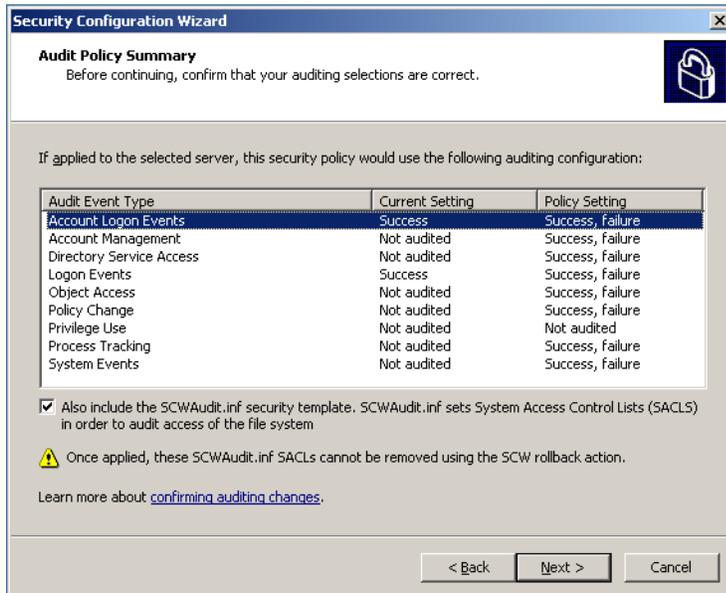


Figure 6.23: Reviewing the changes to the audit policy.

Finally, as Figure 6.24 shows, you'll be asked to save the policy in an XML file. You should provide a detailed description indicating for which type of server this policy is intended; a detailed description will help prevent confusion and error when finally applying the policy.

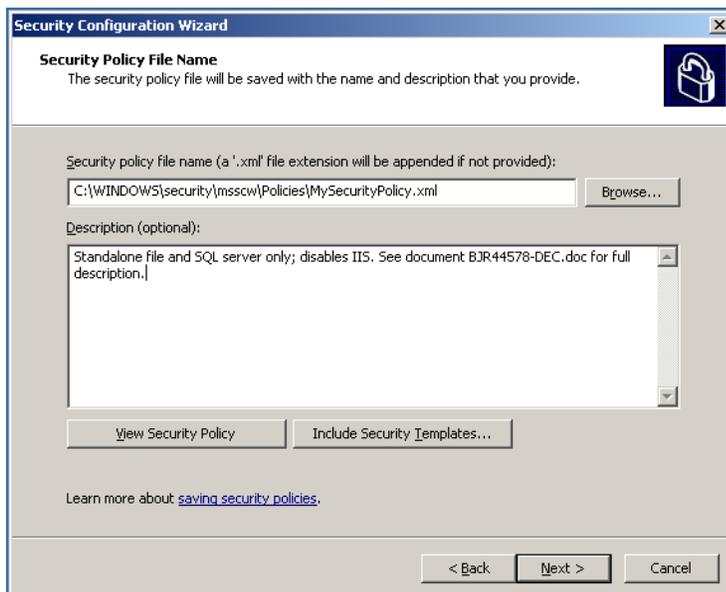


Figure 6.24: Saving the new policy.

SCW is *not* necessarily configuring the server on which it is running. Remember, you specified a baseline computer at the start of SCW, and the policy is built for *that* server. The final screen of SCW (see Figure 6.25), you can choose to simply have the policy saved to a file or you can go ahead and apply it now. You can always re-run SCW later, load in a previously saved policy, and apply it to another computer.

 Applying a policy typically requires a server restart because changes to services' startup modes won't take effect until the server is restarted and those services all stopped.

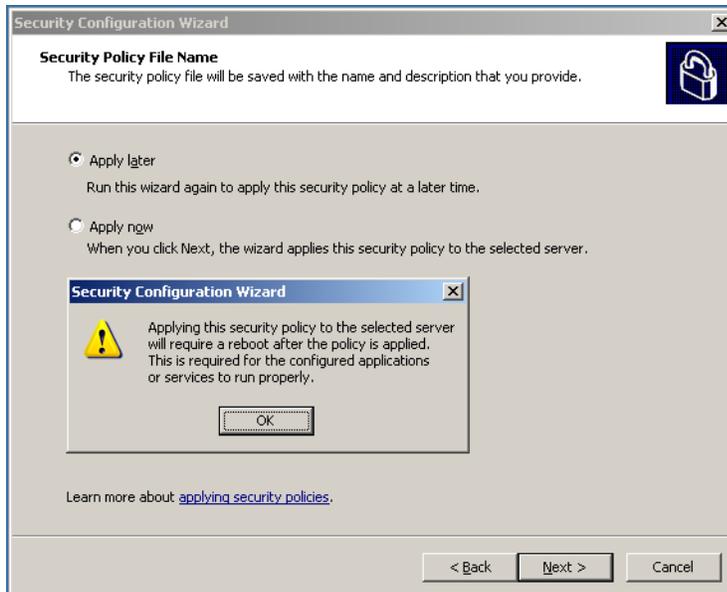


Figure 6.25: Choosing when to apply the policy.

The Results

As Figure 6.26 shows, SCW does have an effect. I chose to apply the policy immediately, and you can see by the netstat –a command output that fewer ports are now open. For easy comparison, this figure shows the ports open before SCW was run (on top) as well as after (on the bottom).

```

C:\Documents and Settings\Administrator>netstat -a
Active Connections

Proto Local Address           Foreign Address         State
TCP    standalone:http         standalone:0            LISTENING
TCP    standalone:epmap       standalone:0            LISTENING
TCP    standalone:microsoft-ds standalone:0            LISTENING
TCP    standalone:1025        standalone:0            LISTENING
TCP    standalone:ms-sql-s    standalone:0            LISTENING
TCP    standalone:ms-olap3    standalone:0            LISTENING
TCP    standalone:ms-sql-m    standalone:0            LISTENING
TCP    standalone:netbios-ssn standalone:0            LISTENING
UDP    standalone:microsoft-ds *:*
UDP    standalone:isakmp      *:*
UDP    standalone:1027        *:*
UDP    standalone:ms-sql-m    *:*
UDP    standalone:ipsec-msft *:*
UDP    standalone:ntp         *:*
UDP    standalone:1032        *:*
UDP    standalone:ntp         *:*
UDP    standalone:netbios-ns *:*
UDP    standalone:netbios-dgm *:*

C:\Documents and Settings\Administrator>

```

Figure 6.26: Comparing ports from before and after applying the SCW-created policy.

SCW provides an effective way to configure multiple servers in a more locked-down fashion, while helping you to avoid typical mistakes that arise from Windows' inherent complexity. However, although SCW offers a great start to configuring high-level security, it doesn't attempt to deal with a number of crucial details.

Beyond the SCW

SCW is primarily intended as a one-time operation, meaning you'll apply a policy to a server and then let it sit. SCW doesn't attempt to handle what you might call day-to-day security tasks, such as managing service identities and passwords; managing registry, file, and folder security; and so forth. Unfortunately, although these tasks are more immediately comprehensible than locking down ports and services, they're made more difficult by the sheer size of the task. I'll cover them in the next two sections.

Service Security

This guide has previously declared the critical need to maintain service security: ensure services aren't running under accounts that have excess authority (running every service on a server under a domain administrator account is an all-too-common scenario) and that services' passwords are changed on a regular basis. Both of these tasks are difficult to accomplish across a number of servers. For example, changing the password used by a service is easy on one server; it's time-consuming and error-prone when you're faced with a dozen or more. Scripts can help automate the process; for example, the script that Listing 6.1 shows is from the Microsoft TechNet Script Center and changes the password of a single service on single computer.

```

strComputer = "."
Set objWMIService = GetObject("winmgmts:" _
    & "{impersonationLevel=impersonate}!\\" & strComputer &
"\root\cimv2")

Set colServiceList = objWMIService.ExecQuery _
    ("Select * from Win32_Service Where StartName = '.\netsvc'")

For Each objService in colServiceList
    errReturn = objService.Change( , , , , , , "password")
Next

```

Listing 6.1: Changing the password of a single service on single computer.

It's a simply process to convert this script into one that runs against multiple computers, and many administrators use techniques such as this to automate their environments. What a script *can't* readily do, however, is ensure that it has caught every single service that needs its password updated. If you miss one, it won't start the next time it needs to, and you'll wind up with a Help desk call and upset users. Similarly, a script can easily be used to list all services running with a particular user account, *provided* you point the script to every server. Miss one, and you'll have incomplete information. In short, scripts are great for well-maintained, well-documented environments—but you may not have one of those.

 You'll find other services-related scripts at <http://www.microsoft.com/technet/scriptcenter/scripts/os/services/default.mspx> and <http://www.scriptinganswers.com>.

Commercial tools are often a better way to work with services in larger environments. For example, Lieberman Software's Service Account Manager (see Figure 6.27), provides a single view of all services on all systems, allowing you to make changes and check services more easily.

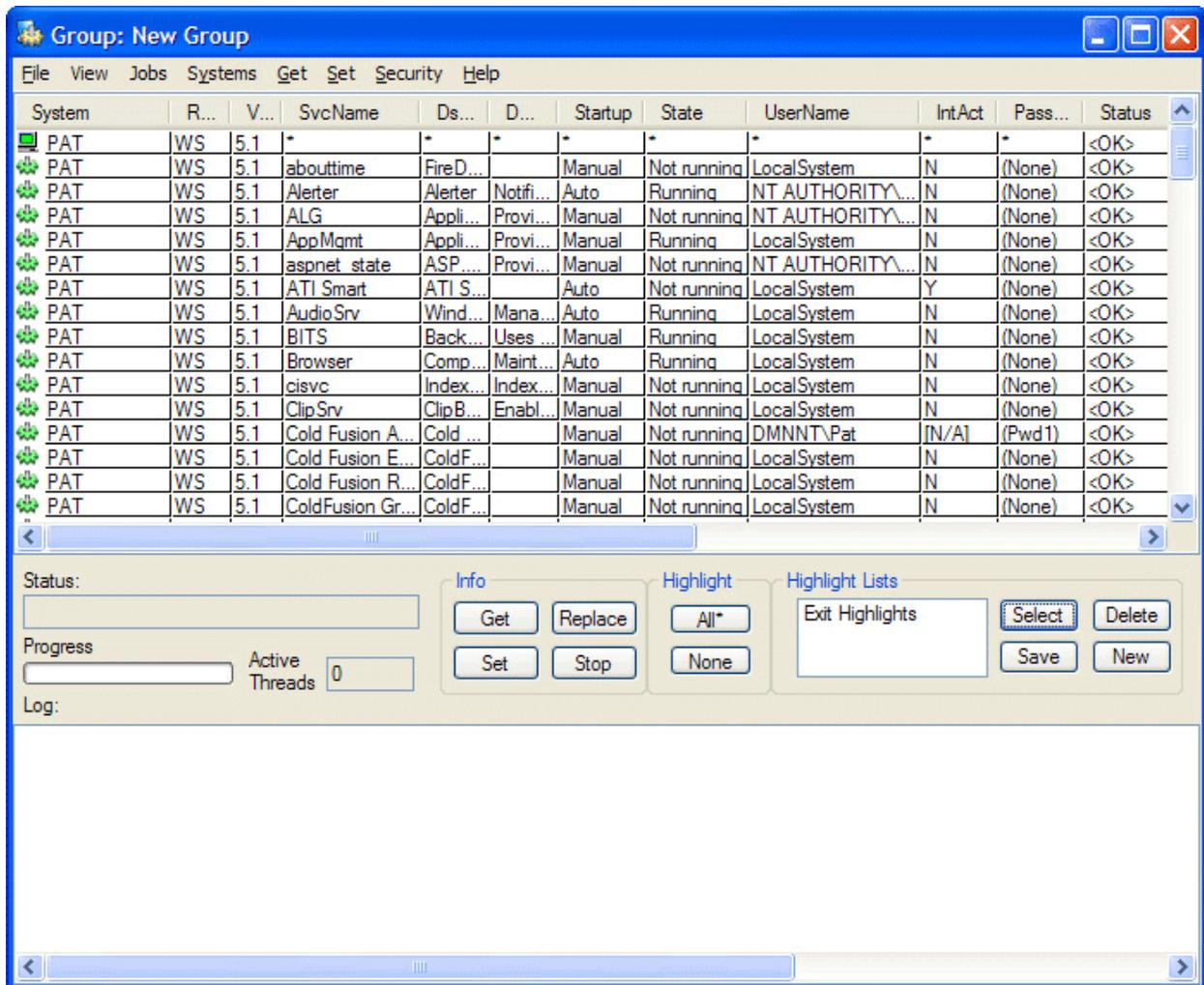


Figure 6.27: Security Account Manager.

Another product, ScriptLogic Service Explorer, provides similar capabilities. As Figure 6.28 shows, you can easily view all the services on any remote system, make changes to services' configurations, and so forth. Service Explorer also handles the management of Scheduled Tasks, which also often run under a user's security credentials.

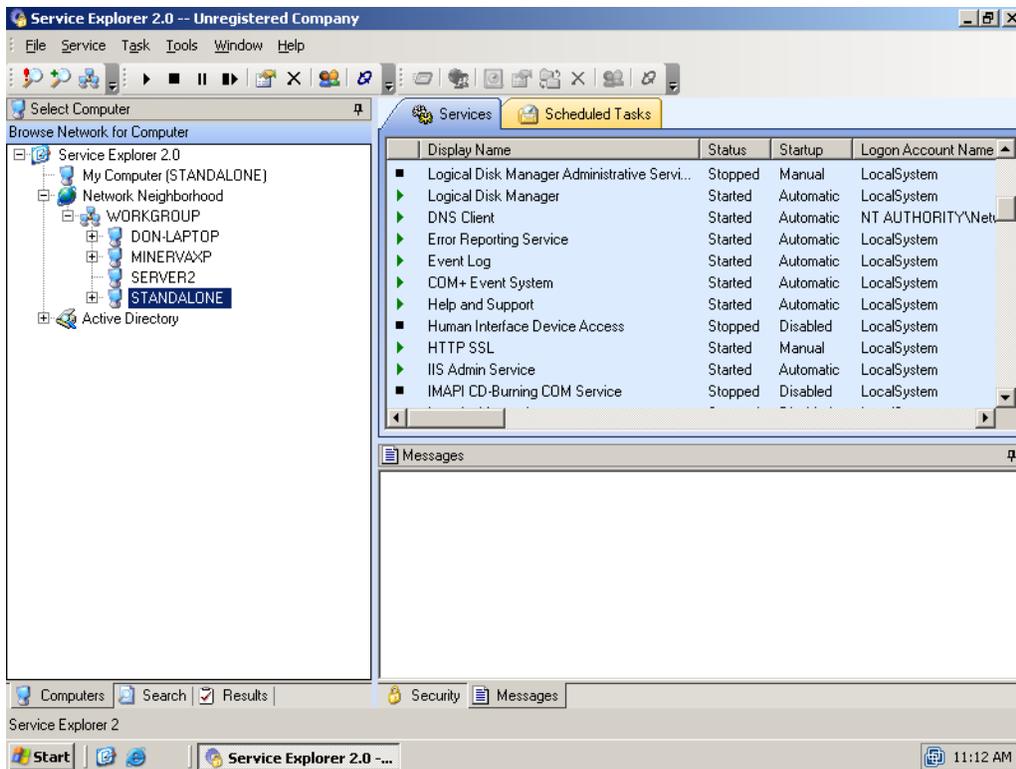


Figure 6.28: Listing services with Service Explorer.

Effective in large environments is the product's built-in search capabilities. For example, as Figure 6.29 shows, you can right-click any service and add it to current search criteria (the "add to search" option is at the bottom of the context menu). This feature makes it easy to quickly find *all* servers running that particular service, making it easier to then change the password or account used by the service, or even to change its startup type or other configuration parameters.

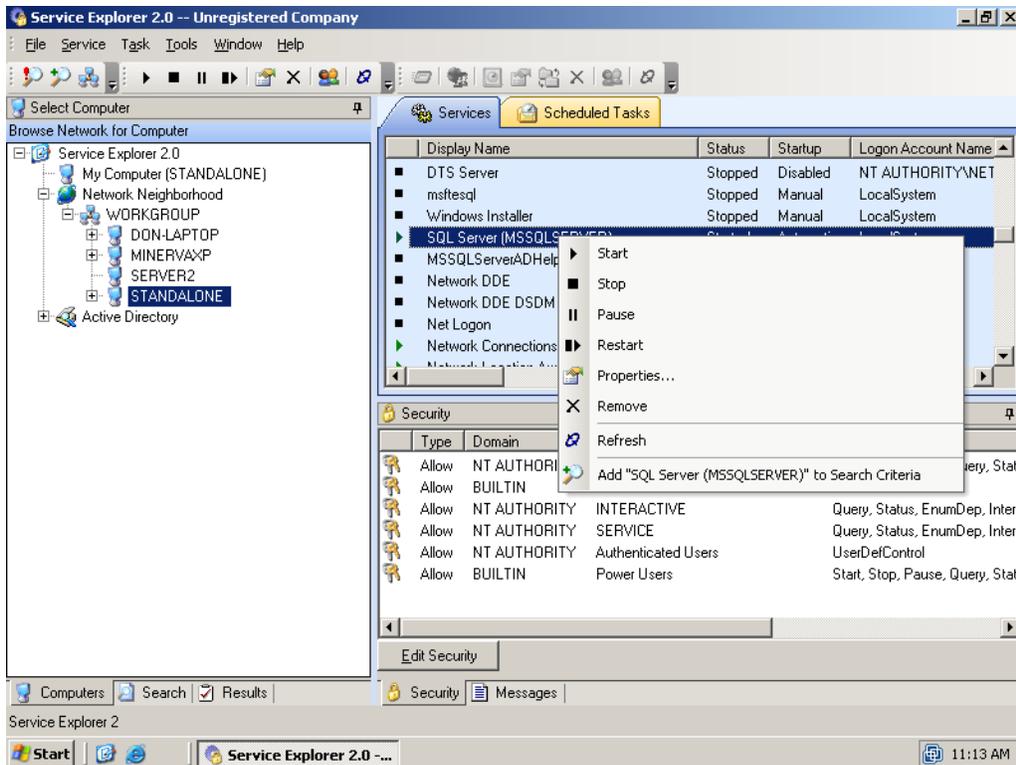


Figure 6.29: Adding a service to search criteria.

Searches can be performed against any scope, such as a manually constructed list of servers, your entire network neighborhood, a workgroup, or even—probably the most useful scope—an entire domain. Search capabilities are broad: you can specify search criteria (shown in the window’s upper-left corner) that includes service names, account names, and so forth. As Figure 6.30 shows, the search process will even include detailed error messages indicating servers that couldn’t be reached, ones for which you haven’t specified valid logon credentials, and so forth.

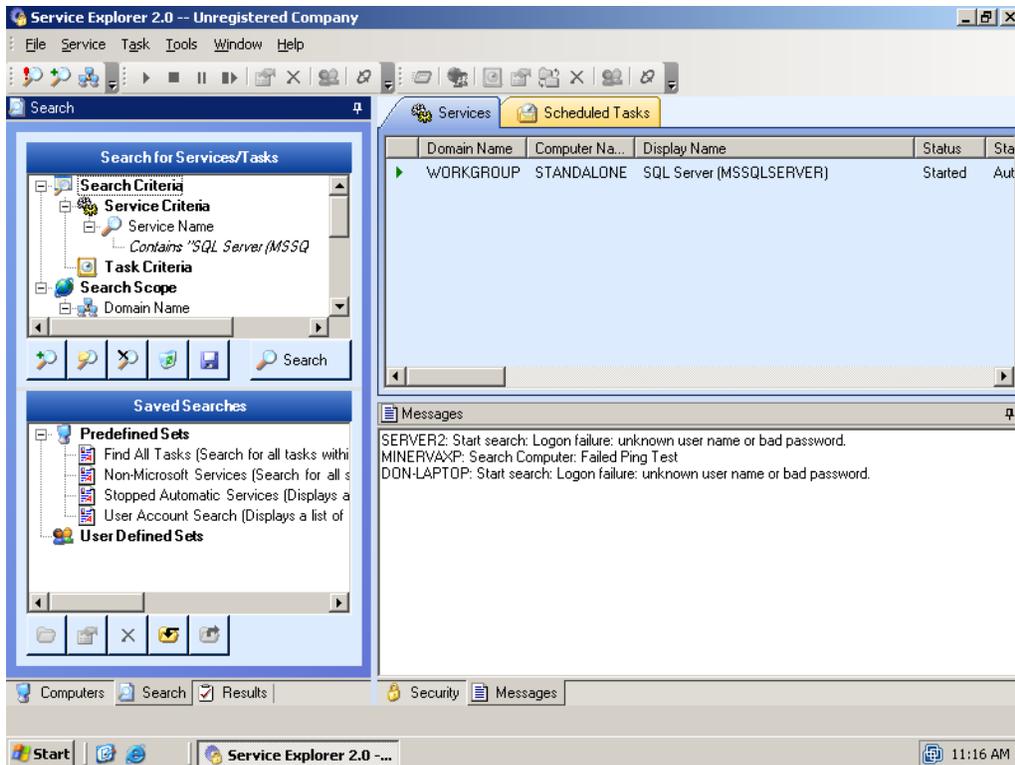


Figure 6.30: Searching includes detailed messages to help you spot any problems.

Products such as Service Explorer and Service Account Manager make it easier to manage services and their security across a large number of servers. You can easily change service passwords on a regular basis, reconfigure services to run under the right user accounts, locate services using a particular user account, and so forth. As many of these tasks—especially updating services’ passwords—go ignored in most environments, these tools are effectively enabling a much higher level of server security than you likely already have.

Access Controls

The previous chapter spent a lot of time on techniques to better manage file and share permissions. Of course, access control isn’t limited to file servers; all servers have files and folders that need to be secured. They have registries, too, which are an often-overlooked area of security that can lead to significant security compromises. Keep in mind that most Windows software stores all its configuration settings in the registry; a compromised registry can lead to seriously compromised applications and servers. With that in mind, let’s explore access control as a general topic, focusing on tools and techniques that can address both files and folders as well as the registry.

Finding Permissions

Windows' built-in security tools—including its graphical access control list (ACL) editor and command-line tools such as `Cacls.exe`—provide effective basic capabilities for working with access controls. However, these tools lack any kind of search capability. For example, if you want to find all files on which the special Everyone account has permissions, doing so is difficult without finding a third-party tool. Fortunately, third-party tools are readily available to handle this task. For example, Figure 6.31 shows such a tool being used to construct a search. As shown, the permissions from an existing folder are displayed, and the Everyone group—which has permissions on that folder—is selected. Clicking Begin Search starts the search for other files and folders with that group assigned.

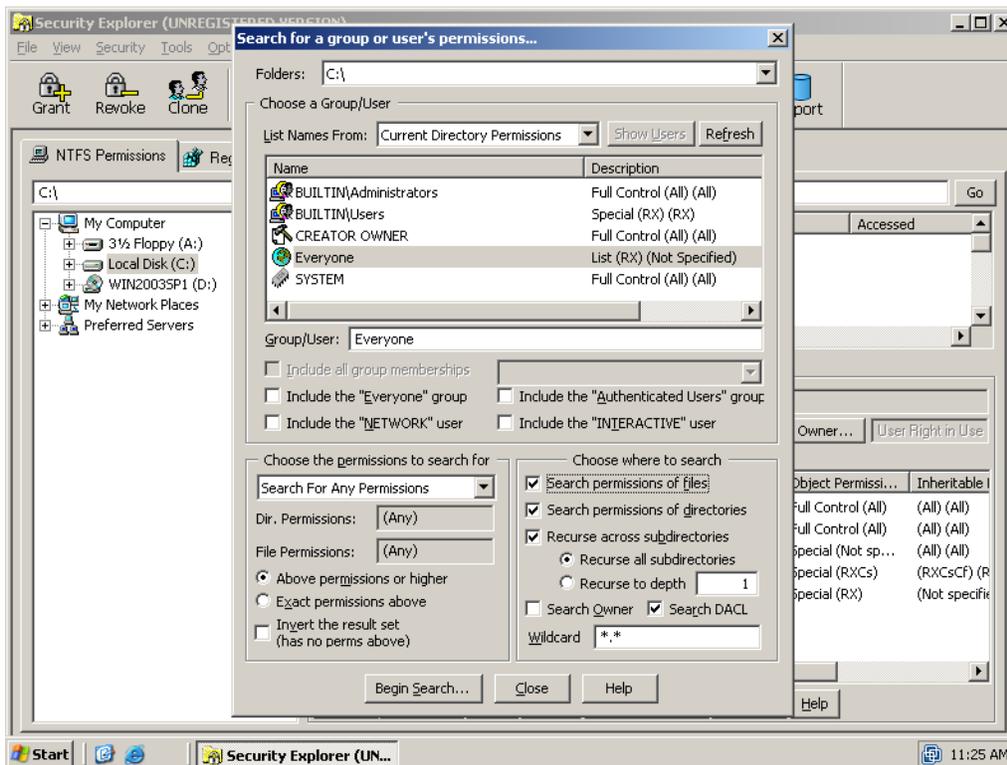


Figure 6.31: Searching for the built-in Everyone group.

The results of the search, which Figure 6.32 shows, quickly identify files and folders where that selected group has permissions, and shows you which permissions are assigned. For example, `D:(RX) F:(RX)` indicates that the group has Read and Execute permissions on a folder and on files within that folder.

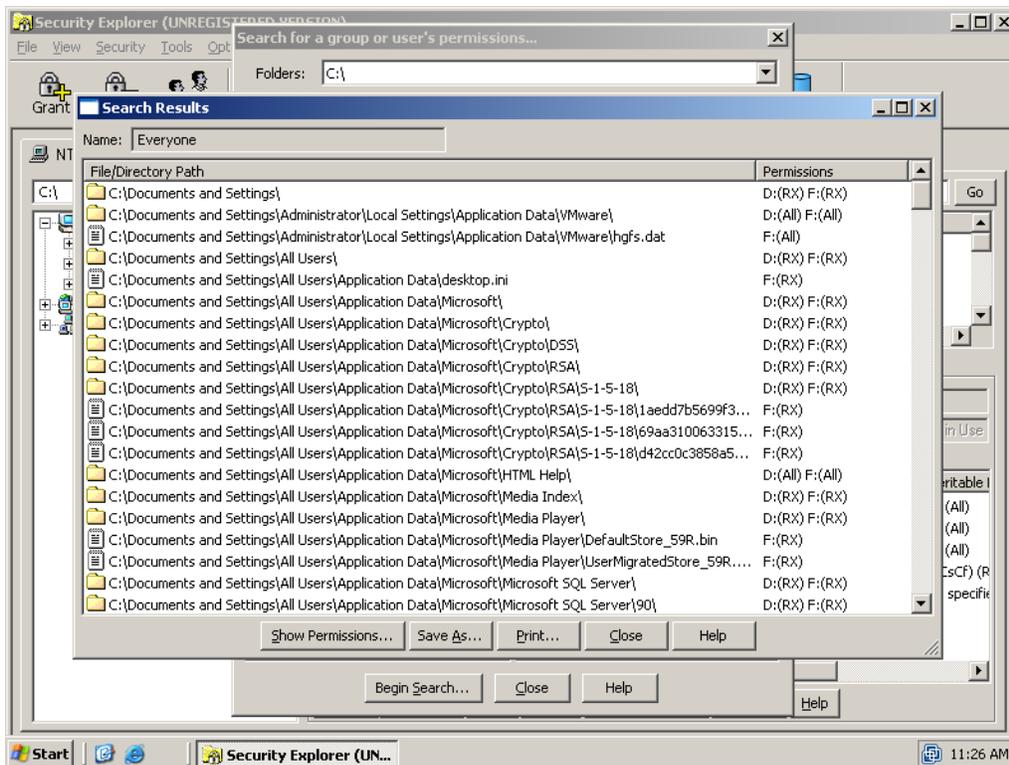


Figure 6.32: Permissions search results.

If you find permissions you don't like, third-party tools make it easier to get rid of them. As Figure 6.33 shows, for example, you can easily get rid of the Everyone group throughout a directory tree. I've selected the special Everyone group and indicated that I want to Revoke All Permissions for the user. This action will—and here's where Windows' security terminology can become confusing—remove any “allow” and “deny” permissions, effectively erasing the Everyone group from the ACLs.

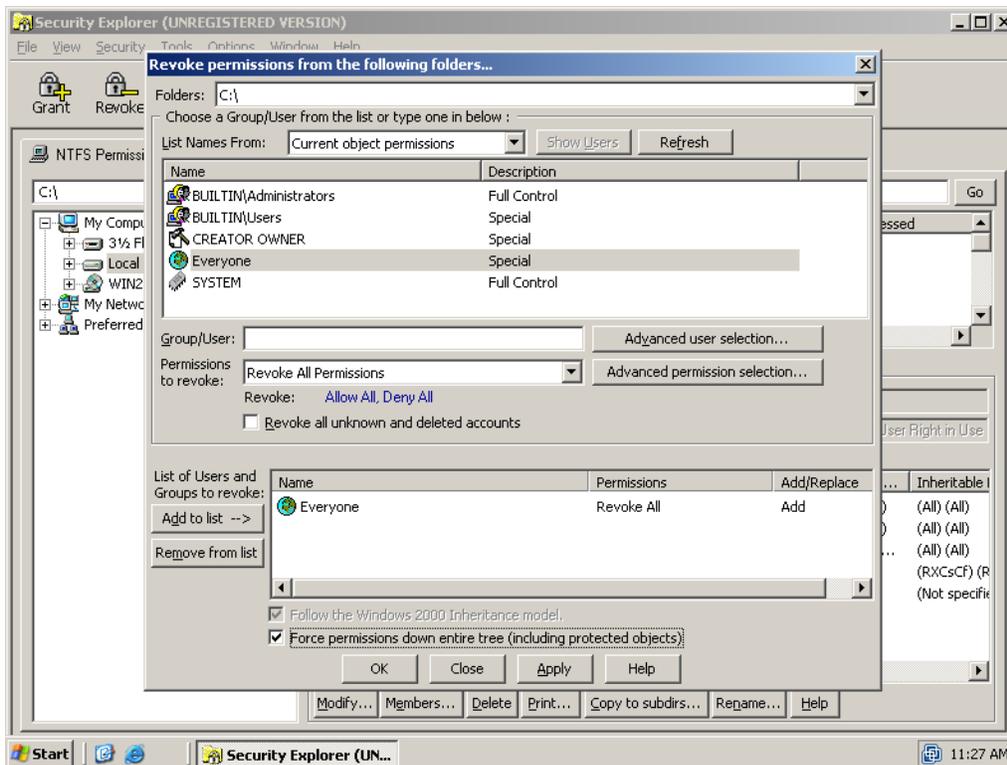


Figure 6.33: Revoking permissions for the Everyone user.

Because server permissions are a bit simpler on non-file servers (typically, a single set of permissions suffices for entire volumes), you can use third-party tools to fix the permissions on a server fairly quickly. Plus, certain tools manage registry permissions as well (see Figure 6.34, which shows an almost-identical screen to Figure 6.33 in which you can correct registry permissions for an entire registry hive or key).

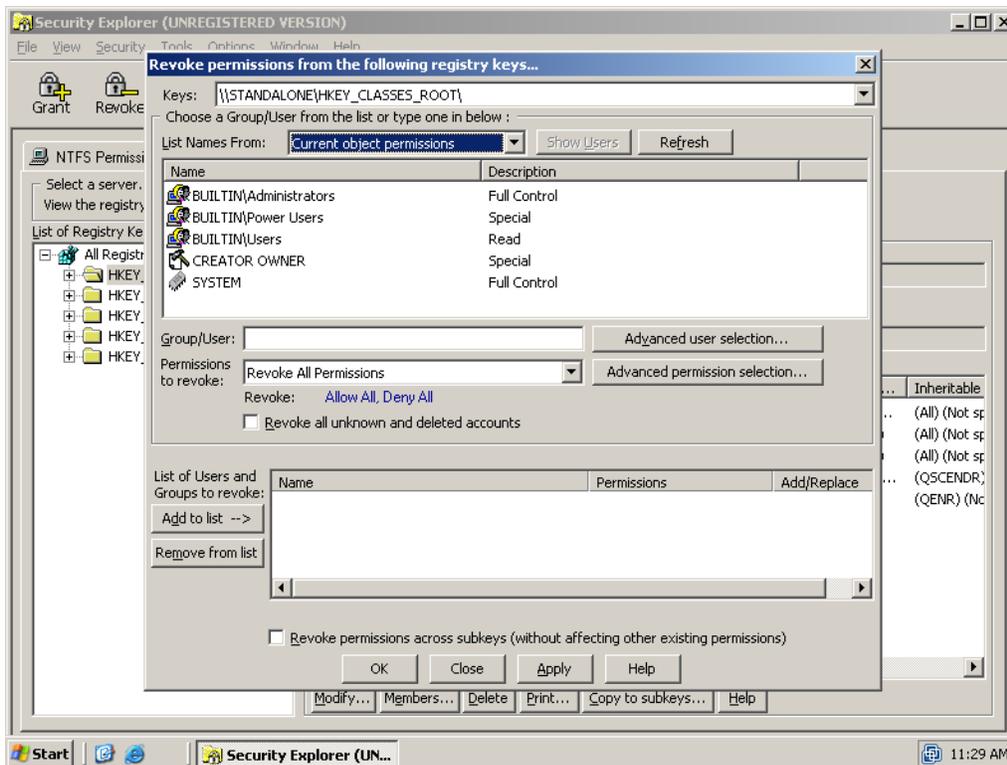


Figure 6.34: Correcting registry permissions.

Many enterprise environments focus on file and folder security on file servers but don't worry about it much on their other servers. After all, users aren't usually given access to the file system on non-file servers, so why bother? In addition, managing file, folder, and registry permissions can be time-consuming. However, with the right tool, permissions can be set up much more effectively with much less effort; taking the time to do so also provides an additional layer of defense in case the wrong person *does* gain access to the file system or the registry.

Permissions Reporting

Being able to report on permissions is a useful capability, especially in today's audit-heavy world of legislative compliance issues. Third-party tools offer the capability to export permissions to an Open Database Connectivity (ODBC) database, such as Microsoft SQL Server. Once in the database, you can create customized reports using standard reporting tools such as Crystal Reports or SQL Server Reporting Services. Figure 6.35 shows the export process.

 ScriptLogic makes another product, Enterprise Security Reporter, which is better-suited for large-scale permissions reporting. Other companies such as NetIQ and Ecora also offer security-reporting solutions.

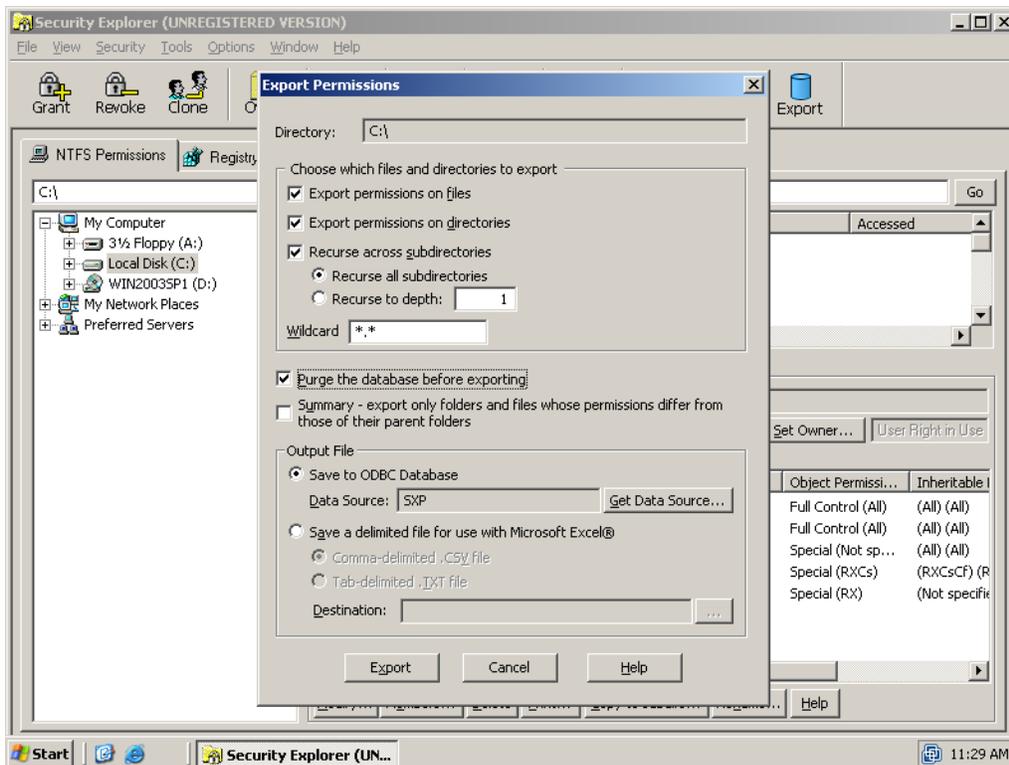


Figure 6.35: Exporting permissions to a database.

Backing Up Permissions

Any backups you make should also back up your permission sets. Windows Backup and most third-party backup utilities are all capable of backing up permissions along with files, and usually handle the registry and its permissions, as well.

Sometimes, however, you might want to restore *only* the permissions on a file, folder, or registry key. Most backup software can only do so if you're also restoring the file, but you may have occasions where you want to restore the permissions—because, perhaps another administrator configured them incorrectly—but *not* the file, folder, or registry key (perhaps it has changed since the last backup was made). Conveniently, effective third-party tools provide this capability by allowing you to back up and restore *only* permissions (see Figure 6.36).

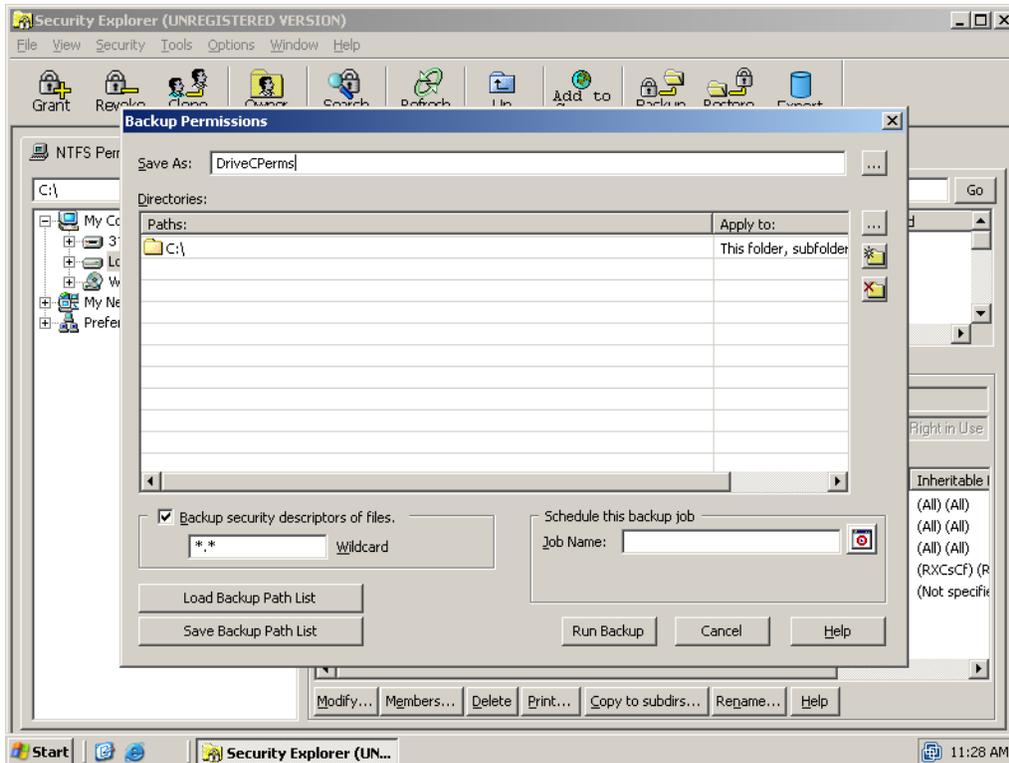


Figure 6.36: Backing up permissions on the C folder.

Summary

Server security boils down to a few basic areas:

- Access controls
- Service management
- Lockdown (of services, applications, and ports)
- Authentication security

Unfortunately, these areas are either large and difficult to manage or complex and difficult to accurately configure (or both). By using the tools that this chapter introduces, however, you'll be able to more easily deal with mass-security reconfigurations, service management, and server lockdown.

Although locking down unnecessary services is a great step to take toward server security, unauthorized or unnecessary software—not just services—is one of the biggest problems facing today's enterprises. The next chapter will talk about software management, software maintenance, and software filtering, techniques guaranteed to help make your Windows enterprise more secure.

Content Central

[Content Central](#) is your complete source for IT learning. Whether you need the most current information for managing your Windows enterprise, implementing security measures on your network, learning about new development tools for Windows and Linux, or deploying new enterprise software solutions, [Content Central](#) offers the latest instruction on the topics that are most important to the IT professional. Browse our extensive collection of eBooks and video guides and start building your own personal IT library today!

Download Additional eBooks!

If you found this eBook to be informative, then please visit Content Central and download other eBooks on this topic. If you are not already a registered user of Content Central, please take a moment to register in order to gain free access to other great IT eBooks and video guides. Please visit: <http://www.realtimepublishers.com/contentcentral/>.