Windows Server 2003 Networking Recipes

Robbie Allen, Laura E. Hunter,
and Bradley J. Dinerman

Apress®

Windows Server 2003 Networking Recipes

ISBN-13 (pbk): 978-1-59059-713-2

ISBN-10 (pbk): 1-59059-713-3

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Lead Editors: Jim Sumser, Jonathan Gennick
Technical Reviewers: Ed Crowley, Jonathan Hassell, William Lefkovics
Editorial Board: Steve Anglin, Ewan Buckingham, Gary Cornell, Jason Gilmore, Jonathan Gennick, Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Dominic Shakeshaft, Jim Sumser, Keir Thomas, Matt Wade
Project Manager: Richard Dal Porto
Copy Edit Manager: Nicole LeClerc
Copy Editor: Andy Carroll
Assistant Production Director: Kari Brooks-Copony
Production Editor: Ellie Fountain
Compositor: Susan Glinert
Proofreader: Elizabeth Berry
Indexer: Julie Grady
Cover Designer: Kurt Krames
Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail `orders-ny@springer-sbm.com`, or visit `http://www.springeronline.com`.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail `info@apress.com`, or visit `http://www.apress.com`.

The source code for this book is available to readers at `http://www.apress.com` in the Source Code section.

CHAPTER 3

Windows Firewall

The Windows Firewall is a feature of Windows Server 2003 Service Pack 1 that creates a protective boundary for Windows Server 2003; it monitors incoming connection attempts on the local computer and restricts information that travels between the local computer and a local area network (LAN) or the Internet. The Windows Firewall provides a way of protecting your server against malicious users or content on the Internet, or elsewhere on your network, that might try to access resources and information on your computer without your permission.

When Windows Server 2003 was first released, the Windows Firewall was known as the Internet Connection Firewall (ICF) and was a simple, host-based firewall for the Windows Server 2003 operating system. ICF was not enabled by default, and it was somewhat difficult to configure. With the release of Service Pack 1 (SP1), Windows Server 2003 now includes the same Windows Firewall that was introduced in Service Pack 2 for Windows XP. This improved firewall is much easier to configure and manage, both from the GUI as well as through Group Policy and the command line.

Note All of the recipes in this chapter assume that you are running Windows Server 2003 Service Pack 1.

Windows Firewall functions by listening for unsolicited incoming requests to your server. A good example of this is a remote client attempting to access the web service running on a physical server. You can configure exceptions based on the name of a service or application, as well as on as the TCP/UDP port it uses to communicate. Using the Windows Firewall, you can configure which applications should be permitted to access your local computer, and which connection attempts should be rejected.

Another configuration improvement in the Windows Firewall is the ability to set up multiple profiles to control how the firewall should behave in different scenarios. You can create configuration items such as firewall exceptions to apply to one or both of the following profiles:

- **Domain profile:** This profile will take effect when your computer is logged onto a domain. The Windows Server 2003 operating system uses a process called *network determination* to figure out whether a computer is currently attached to a domain. This process involves checking the last time that a computer received a Group Policy update, as well as any connection-specific DNS suffixes that have been configured and whether a SLIP or PPP connection is enabled.

- **Standard profile:** In many cases, you will probably enable certain exceptions or other firewall features that should take effect when a computer is attached to a domain—exceptions that you would not want to take effect in a less controlled environment such as a laptop connected to a hotel's public broadband Internet connection. For this reason, many administrators will configure the standard profile with more stringent firewall settings than the domain profile.

When you make a configuration change to the Windows Firewall, you can specify that the change should apply to the domain profile, the standard profile, or all profiles.

Using a Graphical User Interface

Most of the configuration items you'll see in this chapter are done through the Windows Firewall Control Panel applet, which allows you to enable or disable the Windows Firewall, configure exceptions, and set other advanced options. In addition, Recipe 3-19, which discusses viewing event log entries associated with the Windows Firewall, makes use of the free EventCombMT utility, available for download from the Microsoft website as part of the Account and Lockout Management Tools at `http://www.microsoft.com/downloads/details.aspx?FamilyID=7af2e69c-91f3-4e63-8629-b999addeOb9e&displaylang=en`. This free utility allows you to collect and view Event Viewer entries from multiple computers, as well as querying Event Viewer data for specific entries. (The Event Viewer is the MMC snap-in that provides a view of any events that have been logged by a particular computer's auditing settings.)

Using a Command-Line Interface

The primary tool that you'll use to configure the Windows Firewall at the command line is netsh. Netsh has an entire subcontext devoted to the Windows Firewall, which allows you to perform the following configurations and more from the command line:

- Enable or disable the firewall
- Create, edit, or delete program and port exceptions
- Set Internet Control Message Protocol (ICMP) and logging options

In addition, Recipe 3-19, which discusses the Windows Firewall log file, makes use of the Microsoft Log Parser. This is a free utility available for download from the Microsoft website at `http://www.microsoft.com/downloads/details.aspx?FamilyID=890cd06b-abf8-4c25-91b2-f8d975cf8c07&displaylang=en`. This free utility allows you to use a SQL-like query engine to parse information from a number of data sources, including the Windows Firewall log file, which stores information in the industry-standard W3C format.

Using a Group Policy

In addition to the Windows Firewall Control Panel applet introduced by Windows Server 2003 Service Pack 1, you now also have access to a number of options for configuring the firewall using Group Policy. Prior to Service Pack 1, the only configuration you could perform via Group Policy was to globally disable the use of ICF. With the Windows Firewall, you can now perform granular configuration of firewall options and exceptions in both the domain and standard profiles.

Depending on which profile you wish to configure, you'll find the Windows Firewall Group Policy entries in one of the following locations:

- Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile
- Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile

Using the Registry

When configuring the Windows Firewall via the Registry, you'll use one or both of the following keys, depending on which profile you wish to alter:

- `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy\DomainProfile`
- `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy\StandardProfile`

It's important to note here that these Registry keys are valid *only* if the computer you're working on is not receiving configuration data from Group Policy. If it is a Group Policy–enabled computer, these entries will be situated in a different Registry location, and any changes that you make will be overwritten the next time Group Policy refreshes itself. If your Windows Server 2003 computer is receiving its Windows Firewall configuration from Group Policy, it's best to make any changes there for the sake of consistency.

Using VBScript

With VBScript, you can manage the Windows Firewall using the following Windows Firewall interfaces:

- `INetFwOpenPort` provides access to the properties of a single port that you've opened within the Windows Firewall.
- `INetFwOpenPorts` provides access to the *collection* of ports that have been opened within the firewall.
- `INetFwService` provides access to the properties of a single service that's been enabled or disabled through the firewall.
- `INetFwServices` provides access to a *collection* of services that may be authorized to listen through the firewall.
- `INetFwRemoteAdminSettings` configures remote administration settings for utilities such as Computer Management and remote Registry editing.
- `INetFwAuthorizedApplication` allows you to configure the properties of a single application that's been authorized to receive traffic through the firewall.
- `INetFwAuthorizedApplications` provides access to the *collection* of authorized applications.
- `INetFwIcmpSettings` allows you to configure ICMP behavior through the Windows Firewall.

- `INetFwProfile` gives access to a particular firewall profile, either the domain profile, the standard profile, or the current profile that is in effect.
- `INetFwPolicy` provides access to a particular firewall policy.
- `INetFwMgr` provides access to the firewall settings for the local computer.

3-1. Enabling and Disabling the Windows Firewall

Problem

You want to enable or disable the built-in firewall on a Windows Server 2003 computer.

Solution

Using a Graphical User Interface

To enable the Windows Firewall, do the following:

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. Click Settings from the Advanced tab; this will launch the Windows Firewall Control Panel applet.
4. Click the On radio button to enable the Windows Firewall, or Off to disable the firewall.

Using a Command-Line Interface

The following command enables the Windows Firewall on a Windows Server 2003 computer for all profiles:

```
> netsh firewall set opmode mode=enable exceptions=enable profile=all
```

The following command disables the Windows Firewall on a Windows Server 2003 computer:

```
> netsh firewall set opmode mode=disable exceptions=disable profile=all
```

Using Group Policy

Tables 3-1 and 3-2 contain the Group Policy settings that enable or disable the Windows Firewall for the domain and standard profiles respectively.

Table 3-1. *Enable or Disable Windows Firewall—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Protect all network connections
Value	`Enabled` to enable the Windows Firewall for all interfaces in the domain profile. `Disabled` to turn off the Windows Firewall for all interfaces in the domain profile.

Table 3-2. *Enable or Disable Windows Firewall—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Protect all network connections
Value	`Enabled` to enable the Windows Firewall for all interfaces in the standard profile. `Disabled` to turn off the Windows Firewall for all interfaces in the standard profile.

Using the Registry

To configure the Windows Firewall on a Windows Server 2003 computer, configure the following Registry values:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>]
"EnableFirewall"=dword:1
```

Set *<Profile>* to `DomainProfile` or `StandardProfile`. Set the `dword` value to `1` to enable the Windows Firewall or `0` to disable it.

Using VBScript

This code enables the Windows Firewall for the current profile.

```
Set firewall = CreateObject("HNetCfg.FwMgr")
Set firewallPolicy = firewall.LocalPolicy.CurrentProfile
firewallPolicy.FirewallEnabled = TRUE ' FALSE to disable
WScript.Echo("FirewallEnabled set to " & FirewallEnabled & "!")
```

How It Works

The original Release To Manufacturing (RTM) version of the Windows Server 2003 operating system came preloaded with the Internet Connection Firewall (ICF), which provided a simple host-based firewall. The drawback to ICF was that it was not enabled by default when Windows Server 2003 was first installed, and it was fairly unintuitive to enable and configure. Service Pack 1 for Windows Server 2003 made some significant improvements to ICF, which was renamed the Windows Firewall (WF). The Windows Firewall is now enabled when the operating system first boots, and you can make extensive configuration choices through the Windows Firewall Control Panel applet, netsh, and Group Policy.

In certain circumstances, though, you may find it necessary to disable the Windows Firewall. In most cases, this will be because your organization has already standardized on another software or hardware firewall solution, and you want to simplify the configuration of your Windows Server 2003 computers. Even if this is the case, however, the Windows Firewall is able to coexist with third-party products and might still be a useful tool to provide "defense in depth" for your Windows Server 2003 computers. This is especially true if you are relying on a hardware firewall at your network perimeter. These devices constitute the Maginot Line of network defense—if a single malware- or virus-infested PC comes online on the "safe" side of the firewall, your entire network will be at risk without another form of protection. You should also strongly consider

enabling the Windows Firewall for the standard profile even if you disable it in the domain profile. In this way, any mobile users running Windows Server 2003 on their laptops (application developers, for example) will still have firewall protection for their laptop computers when they're not attached to the domain or are otherwise unable to rely on the protection of a corporate firewall.

Using a Command-Line Interface

When using netsh to configure the Windows Firewall, much of the verbiage is optional and can be skipped to save time when entering commands on the fly. For example, this command

```
netsh firewall set opmode mode = enable exceptions = enable profile = all
```

can be shortened to

```
netsh firewall set opmode enable enable all
```

or it can even be made as short as

```
netsh fi set op en en a
```

See Also

- Recipe 3-2 for more on configuring exception processing
- Microsoft: "Understanding Windows Firewall" (`http://www.microsoft.com/windowsxp/using/security/internet/sp2_wfintro.mspx`)

3-2. Configuring Exception Processing

Problem

You want to manage rules and exceptions configured on the Windows Firewall.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the General tab, place a check mark next to Don't Allow Exceptions if you want to prevent the Windows Firewall from allowing entries configured on the Exceptions tab to pass through the firewall. To allow these entries to be processed, remove the check mark next to Don't Allow Exceptions.

Using a Command-Line Interface

The following command enables the Windows Firewall and allows exception traffic to pass:

```
> netsh firewall set opmode mode = enable exceptions = enable
```

The following command enables the Windows Firewall and prevents exception traffic from passing through the firewall:

```
> netsh firewall set opmode mode = enable exceptions = disabled
```

■ **Note** If you do not specify a profile, netsh will assume a default value of `profile=current`.

Using Group Policy

Tables 3-3 and 3-4 contain the Group Policy settings that dictate whether Windows Firewall should allow traffic configured on the Exceptions tab in the domain and standard profiles respectively.

Table 3-3. *Configure Exception Processing—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Do not allow exceptions
Value	`Enabled` to prevent WF from allowing exceptions. `Disabled` to allow exceptions.

Table 3-4. *Configure Exception Processing—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Do not allow exceptions
Value	`Enabled` to prevent WF from allowing exceptions. `Disabled` to allow exceptions.

Using the Registry

To configure an individual computer to allow exceptions to pass through the Windows Firewall, set the following Registry values:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>]
"DoNotAllowExceptions"=dword:0
```

To configure an individual computer to prevent exceptions from passing through the Windows Firewall, set the following Registry values:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\
    SharedAccess\Parameters\FirewallPolicy\<Profile>]
"DoNotAllowExceptions"=dword:1
```

Using VBScript

This code allows exceptions to pass through the Windows Firewall *for the current profile.* Use `LocalPolicy.DomainProfile` or `LocalPolicy.StandardProfile` to alter a specific profile.

```
' ------ SCRIPT CONFIGURATION ------
boolNoExceptions = FALSE ' Set to TRUE to prevent exceptions
 ' from being passed
' ------ END CONFIGURATION ---------

Set firewall = CreateObject("HNetCfg.FwMgr")
Set firewallPolicy = firewall.LocalPolicy.CurrentProfile
firewallPolicy.ExceptionsNotAllowed = boolNoExceptions
WScript.Echo("ExceptionsNotAllowed set to " & boolNoExceptions & "!")
```

How It Works

The Windows Firewall is designed to protect your computer against malicious incoming traffic; whenever the firewall detects this kind of unsolicited request, it blocks the incoming connection and drops any packets associated with it. However, there are circumstances where an application may need to receive this type of traffic in order to function. This is quite common with Instant Messenger applications and network games, as well as server applications like Internet Information Server (IIS) that need to service incoming requests for information. In order to allow this incoming traffic to pass through the firewall, you need to configure a firewall exception based on the name of the application or the TCP/UDP port on which it communicates.

Depending on the network you are connecting to, you may need more or less firewall protection. In addition to enabling or disabling the Windows Firewall, you can also specify whether or not it should allow any exceptions that you've created. This is helpful if you're connecting to the Internet from an unprotected network, like a hotel broadband connection, using your Windows Server 2003 laptop. You can also disallow exceptions in a network emergency, such as if a zero-day virus or worm is circulating the Internet and your antivirus definitions have not yet been updated to detect it. When you configure the Don't Allow Exceptions option, whether through the GUI or another method, the Windows Firewall will block *all* unsolicited incoming traffic that's sent to your computer—this includes blocking any traffic that meets the definitions of any exceptions you've configured, as well as preconfigured services and applications like Remote Assistance, Remote Desktop, and File and Printer Sharing.

See Also

- Recipes 3-3 and 3-4 for more on configuring firewall exceptions
- Microsoft TechNet: "Known Issues for Managing Exceptions" (`http://technet2.microsoft.com/WindowsServer/en/Library/b57b63fb-eb5c-4ca8-999e-e1e7e37fc0f11033.mspx`)

3-3. Creating Program Exceptions

Problem

You want to create a program exception to allow a particular executable to pass through the Windows Firewall on a Windows Server 2003 computer.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. In the Windows Firewall applet, select the Exceptions tab. To add a new program that should be allowed to traverse the firewall, select Add Program.
5. Select the executable from the prepopulated list, or click Browse to navigate to the file on your local hard drive.
6. To define the scope of the exception, click on Change Scope, and select from one of the following three options:
   - Any computer (including those on the Internet).
   - My network (local subnet).
   - Custom list. For this option, enter a single IP address using the syntax `192.168.1.151`, and/or enter a range of addresses using the network ID of the range followed by its subnet mask, such as `192.168.1.1/255.255.255.0`. Separate multiple entries using a comma.
7. Click OK when you're finished.

Using a Command-Line Interface

The following command allows `standard.exe` to pass through the Windows Firewall in the domain profile, but only for computers in the local subnet:

```
> netsh firewall add allowedprogram program = "C:\folder1\Standard.exe"
    name = Standard mode = ENABLE scope = SUBNET profile = DOMAIN
```

When enabling an exception through `netsh`, you can set `mode` to `ENABLE` or `DISABLE`; `scope` to `ALL`, `SUBNET`, or `CUSTOM`; and `profile` to `CURRENT`, `DOMAIN`, `STANDARD`, or `ALL`. If you set `scope` to `CUSTOM`, you also need to specify `addresses =` followed by a comma-separated list of IPv4 IP addresses.

Using Group Policy

Tables 3-5 and 3-6 contain the Group Policy settings that create program exceptions in the domain and standard profiles respectively.

Table 3-5. *Configure Program Exceptions—Domain Profile*

Path	Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile
Policy name	Windows Firewall: Define program exceptions
Value	Enabled to configure a list of program exceptions. Disabled to remove any exceptions previously configured by Group Policy.

Table 3-6. *Configure Program Exceptions—Standard Profile*

Path	Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile
Policy name	Windows Firewall: Define program exceptions
Value	Enabled to configure a list of program exceptions. Disabled to remove any exceptions previously configured by Group Policy.

Using the Registry

To configure an executable called `standard.exe` to pass through the Windows Firewall, set the following Registry values:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>\AuthorizedApplications\List]
"C:\folder1\Standard.exe":reg_sz:"C:\folder1\Standard.exe:*:Enabled:Standard.exe"
```

You can modify this Registry setting for your environment by adhering to the following syntax:

ProgramPath:*Scope*:Enabled|Disabled:*ApplicationName*

- `ProgramPath` allows you to enter the path and filename of the application. You can enter the path manually, or use environment variables such as `%windir%` or `%ProgramFiles%`.
- `Scope` specifies the scope of the exception. You can use * to specify the Any Computer setting, `LocalSubnet` to restrict the exception to your local network, or a single IP address or range of addresses to define a custom list. Create multiple entries by separating them with a comma, like this: `LocalSubnet,10.0.0.151,10.112.25.0/255.255.255.0,10.121.79.0/24`.
- Use `Enabled` or `Disabled` to indicate whether this program should be enabled or disabled in the exception list.
- `ApplicationName` creates a friendly name for the application exception; this is the name that will appear on the Exceptions tab in the Windows Firewall Control Panel applet.

Using VBScript

This code creates a program exception for Standard.exe.

```
' ------ SCRIPT CONFIGURATION ------
Application.Name = "Standard"
Application.ProcessImageFileName = "c:\folder1\Standard.exe"
' ------ END CONFIGURATION ---------

Set firewall = CreateObject("HNetCfg.FwMgr")
Set firewallPolicy = firewall.LocalPolicy.CurrentProfile
Set Application = CreateObject("HNetCfg.FwAuthorizedApplication")

' IPVersion 2 stands for "any version of IP", this is the only correct value
Application.IPVersion = 2


' A scope of 0 allows the exception for all addresses
Application.Scope = 0

Application.Enabled = TRUE

Set Applications = firewallPolicy.AuthorizedApplications
Applications.Add(Application)
WScript.Echo("Program " & Application.Name & " added successfully!")
```

How It Works

One way to configure incoming traffic to pass through the firewall is by using a program exception. This allows a particular executable file to be accepted by the Windows Firewall, so long as exception processing is not disabled. By creating a program exception, Windows Firewall will only listen for traffic related to this exception if the program is running or active—if you close the application that requires the exception, traffic will not pass through the Windows Firewall even if it meets the criteria of the exception.

In addition to specifying the name of the application, you can also configure the *scope* of the exception. Scope options allow you to restrict which computers can communicate using the defined exception, rather than simply throwing it open to the entire world. You can choose from three options when configuring the scope of a program exception:

- **Any computer (including those on the Internet):** This setting will allow any computer on any network, including the Internet, to access your local computer by using the program exception you've defined. This is the least secure setting that you can configure, and it has the potential to expose your server to attacks from malicious users on untrusted networks.
- **My network (subnet) only:** This option will accept incoming requests only from computers that are on the same IPv4 or IPv6 subnet. For example, if your Windows Server 2003 computer has a single network connection to a private network with a network ID of `192.168.1.0` and a subnet mask of `255.255.255.0`, this exception will be valid only for computers whose IP address falls within the range of `192.168.1.1` to `192.168.1.254`.

■ **Caution** The preceding setting can be deceptive, depending on how your server is configured. If you have a network connection to a private network, and are also directly connected to the Internet via a cable modem or other high-speed link, selecting the My Network (Subnet) Only option could potentially leave the exception open to all computers on your Internet service provider's network in addition to your own.

- **Custom list:** This option allows you to specify individual IPv4 addresses, ranges of IPv4 addresses, or any combination of the two. You can specify an individual IP address by simply entering its value, such as `10.0.0.151`. You can specify a range of addresses using either of the following commonly accepted formats:
  - `10.0.0.0/255.0.0.0` indicates the `10.0.0.0` Class A network using the `255.0.0.0` subnet mask
  - `10.0.0.0/8` indicates the same network and subnet mask using classless interdomain routing (CIDR) notation, indicating that the first eight bits of the subnet mask (`255.0.0.0`) are used to indicate the network address.

■ **Note** You cannot specify a custom list for IPv6 addresses, only IPv4.

Using Group Policy

In addition to enabling the Define Program Exceptions setting in Group Policy, you'll also need to define a list of programs that should be added to the exception list via Group Policy, as follows:

1. Enable the Define Program Exceptions setting in the Group Policy Object Editor, and then click Show.
2. Click Add to create a new program exception. In the Show Contents text box, enter the program exception using the following format (each portion of the string is explained next):

   ```
   ProgramPath:Scope:Enabled|Disabled:ApplicationName
   ```

   - `ProgramPath` allows you to enter the path and filename of the application. You can enter the path manually, or use environment variables such as `%windir%` or `%ProgramFiles%`.
   - `Scope` specifies the scope of the exception. You can use * to specify the Any Computer setting, `LocalSubnet` to restrict the exception to your local network, or a single IP address or range of addresses to define a custom list. Create multiple entries by separating them with a comma, like this: `LocalSubnet,10.0.0.151,10.112.25.0/255.255.255.0,10.121.79.0/24`.
   - Use `Enabled` or `Disabled` to indicate whether this program should be enabled or disabled in the exception list.
   - `ApplicationName` creates a friendly name for the application exception; this is the name that will appear on the Exceptions tab in the Windows Firewall Control Panel applet.

■ **Note** You can create a program exception with a status of `Disabled` to prevent local administrators from enabling the program on an individual computer. The `Disabled` setting specified in Group Policy overrides any local settings.

An example of a complete Group Policy entry might look something like this:

```
C:\folder1\Standard.exe*:Enabled:Standard.exe
```

See Also

- Recipe 3-4 for information on configuring port exceptions
- Microsoft TechNet: "Configuring Scope Settings" (`http://technet2.microsoft.com/WindowsServer/en/Library/94af04b3-140e-4108-8165-6d728470d5b21033.mspx`)

3-4. Creating Port Exceptions

Problem

You want to create a port exception to allow traffic on a particular TCP or UDP port to pass through the Windows Firewall on a Windows Server 2003 computer.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. In the Windows Firewall applet, select the Exceptions tab. To add a new port that should be allowed to traverse the firewall, click Add Port.
5. Enter the name of the program or service in the Name text box. Enter the port number in the Port Number text box.
6. To define the scope of the exception, click on Change Scope, and select from one of the following three options:
   - Any computer (including those on the Internet).
   - My network (local subnet).
   - Custom list. For this option, enter a single IP address using the syntax `192.168.1.151`, and/or enter a range of addresses using the network ID of the range followed by its subnet mask, such as `192.168.1.1/255.255.255.0`. Separate multiple entries using a comma.
7. Click OK when you're finished.

Using a Command-Line Interface

The following command allows an application named FOO using TCP port 11065 to traverse the Windows Firewall. It restricts FOO to the local subnet, and configures it for the standard profile:

```
> netsh firewall add portopening protocol = TCP port = 11065
    name = FOO mode = ENABLE scope = SUBNET profile = STANDARD
```

Using Group Policy

Tables 3-7 and 3-8 contain the Group Policy settings that create port exceptions in the domain and standard profiles respectively.

Table 3-7. *Configure Port Exceptions—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Define port exceptions
Value	`Enabled` to configure a list of port exceptions. `Disabled` to remove any exceptions previously configured by Group Policy.

Table 3-8. *Configure Port Exceptions—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Define port exceptions
Value	`Enabled` to configure a list of port exceptions. `Disabled` to remove any exceptions previously configured by Group Policy.

Using the Registry

To configure an individual computer to allow an application called FOO using TCP port 11065 to traverse the Windows Firewall, set the following Registry value:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\DomainProfile\AuthorizedApplications\List]
"11065:TCP":reg_sz:"11065:TCP:*:Enabled:FOO"
```

■ **Note** To restrict FOO traffic to the local subnet, change the `"11065:TCP"` `reg_sz` value to `11065:TCP:LocalSubNet:Enabled:FOO`.

Using VBScript

This code allows an application named FOO using TCP port 11065 to traverse the Windows Firewall.

```
' ------ SCRIPT CONFIGURATION ------
 Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy.CurrentProfile
 Set Port = CreateObject("HNetCfg.FwOpenPort")
' ------ END CONFIGURATION ---------

 Port.Port = 11065
 Port.Name = "FOO"
 Port.Protocol = NET_FW_IP_PROTOCOL_TCP
 Port.Enabled = TRUE

 set Ports = Policy.GloballyOpenPorts
 addedPorts = Ports.Add(Port)
WScript.Echo "Ports configured."
```

How It Works

Like program exceptions, port exceptions allow a certain type of traffic to traverse the Windows Firewall. Unlike program exceptions, which are based on a particular executable filename, port exceptions simply allow any traffic that is destined for a particular TCP or UDP port.

Another significant difference between port exceptions and program exceptions is that a program exception will only be active for as long as the program is running; if the program exits, the exception is no longer active. Contrast this with port exceptions, which will always listen for traffic on a particular port regardless of whether the associated application is running or not.

Because program exceptions lend themselves to more granular control of the Windows Firewall, you should try to enable program exceptions wherever possible. When necessary, however, you can enable port exceptions to listen continuously for incoming traffic bound for a particular port.

Using Group Policy

When defining program exceptions in Group Policy, ports should be added to the exception list as follows:

1. Enable the Define Port Exceptions setting in the Group Policy Object Editor, and then click Show.

2. Click Add to create a new program exception. In the Show Contents text box, enter the program exception using the following format (each portion of the string is explained next):

   ```
   Port#:TCP|UDP:Scope:Enabled|Disabled:PortName
   ```

- `Port#` specifies the port number of the exception you're creating.
- Use `TCP` or `UDP` to specify the transport-level protocol being used by the port exception you're creating.
- `Scope` specifies the scope of the exception. You can use * to specify the Any Computer setting, `LocalSubnet` to restrict the exception to your local network, or a single IP address or range of IP addresses to define a custom list. Create multiple entries by separating them with a comma, like this: `LocalSubnet,10.0.0.151,10.112.25.0/255.255.255.0,10.121.79.0/24`
- Use `Enabled` or `Disabled` to indicate whether this port should be enabled or disabled in the exception list.
- `PortName` creates a user-friendly name for the port exception; this is the name that will appear on the Exceptions tab in the Windows Firewall Control Panel applet.

■ **Note** You cannot specify a custom scope for IPv6 addresses; it only supports * or `LocalSubnet`.

A complete Group Policy entry might look something like this:

```
8080:TCP:LocalSubnet:Enabled:IntranetApps
```

See Also

- Recipe 3-3 for more on creating program exceptions
- Microsoft TechNet: "Configuring Port Exceptions" (`http://technet2.microsoft.com/WindowsServer/en/Library/e53c01ac-1e0a-4693-af58-9242b884b5cd1033.mspx`)

3-5. Managing Exceptions

Problem

You want to edit or delete an existing program or port exception on the Windows Firewall.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.

3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. In the Windows Firewall applet, select the Exceptions tab. To edit an existing port or application exception, select the exception and click Edit. To remove the exception altogether, select the exception and click Delete.

■ **Note** When modifying a port exception, you can modify the name, scope, port number, and whether it uses TCP or UDP. When modifying a program exception, you can only modify its scope. If you need to change the executable name, you must delete the exception and create a new one.

Using the Registry

To modify an existing program or port exception, modify the appropriate `reg_sz` entry in the following location:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>\AuthorizedApplications\List]
```

Using a Command-Line Interface

The following command configures the Windows Firewall to permit an application named FOO using TCP port 11060. It restricts FOO to the local subnet and configures it for the domain profile:

```
> netsh firewall set portopening protocol = TCP port = 11060
    name = FOO mode = ENABLE scope = SUBNET profile = DOMAIN
```

The following command configures an existing application exception called `Standard App` to refer to an executable in the `C:\Program Files\` directory:

```
> set allowedprogram program = "C:\Program FilesProgramFiles\Standard.exe"
    name = "Standard App" mode = ENABLE
```

The following two commands delete an existing application exception in the standard profile and delete an existing port exception from all profiles:

```
> netsh firewall delete allowedprogram program "C:\Program Files\Standard.exe"
    profile = STANDARD
> netsh firewall delete portopening protocol = TCP port = 11060 profile = ALL
```

Using Group Policy

To modify an exception that you've configured using Group Policy, delete the existing exception and re-create it using the instructions in Recipe 3-3 or 3-4.

Using VBScript

This code removes an existing application exception and re-creates it with new values.

```
' ------ SCRIPT CONFIGURATION ------
 Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy
 Set Profile = Policy.GetProfileByType(1)
 strCurrentApp = "c:\program files\image\image123.exe"
 strNewApp = "c:\program files\image\imaging.exe"
' ------ END CONFIGURATION ---------

Set colApplications = Profile.AuthorizedApplications

For Each Application in colApplications
  If Application.ProcessImageFileName = strCurrentApp Then
    WScript.Echo "Removed " & Application.Name " from authorized list!"
    colApplications.Remove(Application)
  End If
Next

Set newApplication = CreateObject("HNetCfg.FwAuthorizedApplication")
newApplication.Name = "Imaging"
newApplication.IPVersion = 2
newApplication.ProcessImageFileName = strNewApp
newApplication.RemoteAddresses = "*"
newApplication.Scope = 0
newApplication.Enabled = True

colApplications.Add(objApplication)

WScript.Echo "Application " & strNewApp & " added successfully!"
```

How It Works

As an ongoing maintenance task, it's critical that you examine your Windows Firewall configuration on a regular basis to make sure that its configuration is up to date. At a minimum, you should remove any entries in the exception list that are out of date or not being used anymore, to prevent a malicious user or content from accessing your network via one of these "legitimate" entries. In addition, you can fine-tune your Windows Firewall configuration to limit the scope of your exceptions so that only legitimate users and computers can send unsolicited traffic to your Windows Server 2003 computer.

See Also

- Recipe 3-17 for more on configuring inbound connectivity
- Microsoft TechNet: "Help: Edit or Delete a Program Exception" (`http://technet2.microsoft.com/WindowsServer/en/Library/a433192f-6073-42a1-ab9b-de289d7e67951033.mspx`)
- Microsoft TechNet: "Help: Edit or Delete a Port Exception" (`http://technet2.microsoft.com/WindowsServer/en/Library/c058566f-0d3f-41e4-9ff1-08c4409a17f51033.mspx`)

3-6. Configuring Local Exceptions

Problem

You want to allow local administrators to create program or port exceptions in addition to those specified in Group Policy.

Solution

Tables 3-9 and 3-10 contain the Group Policy settings that allow local administrators to configure port or program exceptions in the domain profile.

Table 3-9. *Configure Local Program Exceptions—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow local program exceptions
Value	`Enabled` to allow local administrators to create exceptions. `Disabled` to prevent local administrators from changing firewall settings.

Table 3-10. *Configure Local Port Exceptions—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow local port exceptions
Value	`Enabled` to allow local administrators to create exceptions. `Disabled` to prevent local administrators from changing firewall settings.

Tables 3-11 and 3-12 contain the Group Policy settings that allow local administrators to configure port or program exceptions in the standard profile.

Table 3-11. *Configure Local Program Exceptions—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow local program exceptions
Value	`Enabled` to allow local administrators to create exceptions. `Disabled` to prevent local administrators from changing firewall settings.

Table 3-12. *Configure Local Port Exceptions—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow local port exceptions
Value	`Enabled` to allow local administrators to create exceptions. `Disabled` to prevent local administrators from changing firewall settings.

How It Works

In a domain environment, the most efficient and effective way to configure the Windows Firewall is through the use of Group Policy objects (GPOs). This will ensure a consistent firewall configuration for all of the computers in your enterprise. However, you may need to allow local administrators to create their own exceptions for particular applications that they've deployed locally to a single Windows Server 2003 server. You can allow this by configuring the Allow Local Program/Port Exceptions settings in Group Policy.

Enabling this setting will allow the local administrator of a Windows Server 2003 computer to create a new exception for accepting inbound traffic. If you disable this setting, you have effectively prevented even your local administrators from altering the configuration of the Windows Firewall; the only exceptions that will be permitted are those defined by Group Policy.

See Also

- Recipes 3-3 and 3-4 for more on configuring exceptions
- Microsoft TechNet: "Help: Prevent Administrators from Configuring Local Program Exceptions" (`http://technet2.microsoft.com/WindowsServer/en/Library/134fedbd-2a53-4f70-893f-b8077a9328741033.mspx`)
- Microsoft TechNet: "Help: Prevent Administrators from Configuring Local Port Exceptions (`http://technet2.microsoft.com/WindowsServer/en/Library/1969e151-92bb-4108-b684-73f670e9c0521033.mspx`)

3-7. Configuring ICMP Traffic

Problem

You want to configure how Internet Control Message Protocol (ICMP) traffic is passed through or blocked by the Windows Firewall.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.

3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Advanced tab, click the Settings button in the ICMP section.
5. In the ICMP Settings section, place a check mark next to the ICMP packets that you want to allow:
   - Allow incoming echo request
   - Allow incoming timestamp request
   - Allow incoming mask request
   - Allow incoming router request
   - Allow outgoing destination unreachable
   - Allow outgoing source quench
   - Allow outgoing parameter problem
   - Allow outgoing time exceeded
   - Allow redirect
   - Allow outgoing packet too big
6. Click OK when you're finished.

Using a Command-Line Interface

The following command allows all ICMP traffic to pass through the Windows Firewall for the standard profile:

```
> netsh firewall set icmpsetting type = ALL mode = ENABLE profile = STANDARD
```

You can set `mode` to `ENABLE` or `DISABLE`; `profile` to `CURRENT`, `DOMAIN`, `STANDARD`, or `ALL`. For `type`, you can specify one or more of the following:

`2`: Allow outbound packet too big

`3`: Allow outbound destination unreachable

`4`: Allow outbound source quench

`5`: Allow redirect

`8`: Allow inbound echo request

`9`: Allow inbound router request

`11`: Allow outbound time exceeded

`12`: Allow outbound parameter problem

`13`: Allow inbound timestamp request

17: Allow inbound mask request

ALL: All types

You can use commas or dashes to specify multiple exceptions, such as `type = 2, 13, 11` or `type = 2-5, 9`.

Using Group Policy

Tables 3-13 and 3-14 show the settings that control ICMP traffic handling in the domain and standard profiles respectively.

Table 3-13. *Configure ICMP Traffic—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow ICMP exceptions
Value	`Enabled` to allow ICMP exceptions; `Disabled` to prevent them.

Table 3-14. *Configure ICMP Traffic—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow ICMP exceptions
Value	`Enabled` to allow ICMP exceptions; `Disabled` to prevent them.

Using the Registry

To configure an individual computer to allow various types of ICMP traffic through the Windows Firewall, set the following Registry values:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>\IcmpSettings\]
"AllowInboundEchoRequest"=dword:1
"AllowInboundRouterRequest"=dword:1
"AllowInboundTimestampRequest"=dword:1
"AllowInboundMaskRequest"=dword:1
"AllowOutboundDestinationUnreachable"=dword:1
"AllowOutboundSourceQuench"=dword:1
"AllowOutboundTimeExceeded"=dword:1
"AllowOutboundParameterProblem"=dword:1
"AllowOutboundPacketTooBig"=dword:1
"AllowRedirect"=dword:1
```

■ **Note** When the Windows Firewall is enabled, all ICMP exceptions are turned off by default.

Using VBScript

This code enables all ICMP traffic to pass through the Windows Firewall.

```
Set Firewall = CreateObject("HNetCfg.FwMgr")
Set Policy = Firewall.LocalPolicy.CurrentProfile

Set ICMPSettings = Policy.ICMPSettings

ICMPSettings.AllowInboundEchoRequest = TRUE
ICMPSettings.AllowInboundMaskRequest = TRUE
ICMPSettings.AllowInboundRouterRequest = TRUE
ICMPSettings.AllowInboundTimestampRequest = TRUE
ICMPSettings.AllowOutboundDestinationUnreachable = TRUE
ICMPSettings.AllowOutboundPacketTooBig = TRUE
ICMPSettings.AllowOutboundParameterProblem = TRUE
ICMPSettings.AllowOutboundSourceQuench = TRUE
ICMPSettings.AllowOutboundTimeExceeded = TRUE
ICMPSettings.AllowRedirect = TRUE
WScript.Echo "Settings enabled"
```

How It Works

The Internet Control Message Protocol (ICMP) is used by TCP/IP utilities such as ping and tracert to assist in network troubleshooting and diagnostics. By default, the Windows Firewall will block any incoming ICMP traffic destined for the local computer, since ICMP can be misused as part of a number of network attacks. If you do not enable ICMP exceptions, you will not be able to use any network utilities that rely on ICMP to contact your Windows Server 2003 computer.

However, if you open TCP port 445, the Windows Firewall will automatically allow incoming ICMP echo messages, even if the Allow ICMP Exceptions setting is disabled through Group Policy. This will occur if you specifically create a port exception for TCP port 445, or if you enable the file and printer sharing or remote administration exceptions.

Using a Command-Line Interface

If you want to enable only specific ICMP message types using netsh, you'll need to specify a number associated with the message type in the `type =` parameter. The numeric values associated with the ICMP message types are as follows:

2: Allow outbound packet too big

3: Allow outbound destination unreachable

4: Allow outbound source quench

5: Allow redirect

8: Allow inbound echo request

9: Allow inbound router request

11: Allow outbound time exceeded

12: Allow outbound parameter problem

13: Allow inbound timestamp request

17: Allow inbound mask request

To enable individual ICMP message types, you'll need to issue a separate `netsh` command for each message type that you want, or use `type = ALL` to allow all ICMP messages.

Using Group Policy

In addition to enabling the Allow ICMP Exceptions Group Policy setting, you'll also need to place a check mark next to the individual ICMP message types that you wish to allow. For this setting, at least, the Group Policy user interface is essentially identical to the Windows Firewall Control Panel applet; it does not require you to manually enter a complex configuration string.

■ **Note** Enabling ICMP exceptions overrides any locally configured ICMP settings on a Windows Server 2003 computer. If you *disable* this setting in Group Policy, local administrators will be unable to define any ICMP exceptions locally.

See Also

- Microsoft TechNet: "Help: Configure ICMP Exceptions" (`http://technet2.microsoft.com/WindowsServer/en/Library/b07dd75f-ab62-475a-be2e-709f67416c201033.mspx`)
- Microsoft TechNet: "Block and Unblock ICMP Messages" (`http://technet2.microsoft.com/WindowsServer/en/Library/c65c9cdc-0613-411c-a474-b779dac4d1881033.mspx`)
- MSDN: "Configuring ICMP Settings in Windows Firewall" (`http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xpehelp/html/xeconconfiguringicmpsettingsinwindowsfirewall.asp`)

3-8. Configuring Remote Administration Through the Windows Firewall

Problem

You want to configure the Windows Firewall to allow remote administration of a Windows Server 2003 computer.

Solution

Using a Command-Line Interface

The following command enables the remote administration exception for the local subnet in the domain profile:

```
> netsh firewall set service type = REMOTEADMIN mode = ENABLE
    scope = SUBNET profile = DOMAIN
```

As with other exceptions that you enable through `netsh`, you can set `mode` to `ENABLE` or `DISABLE`; `scope` to `ALL`, `SUBNET`, or `CUSTOM`; and `profile` to `CURRENT`, `DOMAIN`, `STANDARD`, or `ALL`. If you set the `scope` to `CUSTOM`, you also need to specify `addresses =` followed by a comma-separated list of IPv4 IP addresses.

Using Group Policy

Tables 3-15 and 3-16 contain the Group Policy settings that enable remote administration through the Windows Firewall in the domain and standard profiles respectively.

Table 3-15. *Configure Remote Administration Exception—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow remote administration exception
Value	`Enabled` to allow remote administration. `Disabled` to prevent it.

Table 3-16. *Configure Remote Administration Exception—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow remote administration exception
Value	`Enabled` to allow remote administration. `Disabled` to prevent it.

Using the Registry

To configure an individual computer to allow for remote administration through the Windows Firewall, set the following Registry value:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>\RemoteAdminSettings\]
"Enabled"=dword:1
"RemoteAddresses=reg_sz:"IpAddress, IpAddress, localsubnet"
```

Using VBScript

This code enables the remote administration exception for the current profile.

```
Set Firewall = CreateObject("HNetCfg.FwMgr")
Set Policy = Firewall.LocalPolicy.CurrentProfile
Set AdminSettings = Policy.RemoteAdminSettings
AdminSettings.Enabled = TRUE
WScript.Echo "Setting enabled"
```

How It Works

In a domain environment, you'll often want to remotely administer server and workstations using tools such as Computer Management or Windows Management Instrumentation (WMI). This is because most of the administration tools you'll use need to make unsolicited incoming connections to the computer that you're trying to administer, using TCP port 445 and the svchost.exe and lsass.exe executables. As such, you'll need to open the necessary ports on the Windows Firewall to allow you to use these tools on machines in your environment.

■ **Caution** The ports and executables used by the remote administration exception are well-known attack vectors. Be sure to only open this exception selectively to trusted hosts that require access to it.

In order to enable remote administration through the Windows Firewall, you'll need to enable the appropriate setting in Group Policy, the Windows Registry, or VBScript; you cannot make this change in the Windows Firewall Control Panel applet.

In addition to enabling the remote administration exception through Group Policy, you need to specify the IPv4 addresses that are permitted to make remote administration connections. As with other Windows Firewall Group Policy settings, you can use `LocalSubnet` to specify the local subnet, * to specify all hosts, or a custom list of addresses. For IPv6 addresses, you can only specify `LocalSubnet` or *; you can't create a custom exception list.

See Also

- Recipes 3-3 and 3-4 for more on enabling program and port exceptions
- Microsoft TechNet: "Windows Firewall Settings: Remote Administration Tools" (`http://technet2.microsoft.com/WindowsServer/en/Library/62d661cc-8267-4440-aacc-55358c602a081033.mspx`)

3-9. Configuring File and Print Sharing Through the Windows Firewall

Problem

You want to configure the Windows Firewall to allow file and printer sharing.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Exceptions tab, place a check mark next to File and Printer Sharing.
5. To define the scope of the exception, click on Edit, followed by Change Scope, and select from one of the following three options:
   - Any computer (including those on the Internet).
   - My network (local subnet).
   - Custom list. For this option, enter a single IP address using the syntax `192.168.1.151`, and/or enter a range of addresses using the network ID of the range followed by its subnet mask, such as `192.168.1.1/255.255.255.0`. Separate multiple entries using a comma.
6. Click OK when you're finished.

Using a Command-Line Interface

The following command enables the file and printer sharing exception.

```
> netsh firewall set service type = fileandprint mode = ENABLE
```

Using Group Policy

Tables 3-17 and 3-18 show the settings that control the file and printer sharing exception within the Windows Firewall in the domain and standard profiles respectively.

Table 3-17. *Configure File and Printer Sharing Exception—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow file and printer sharing exception
Value	`Enabled` to allow file and printer sharing. `Disabled` to prevent it.

Table 3-18. *Configure File and Printer Sharing Exception—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow file and printer sharing exception
Value	`Enabled` to allow file and printer sharing. `Disabled` to prevent it.

Using VBScript

This code enables the file and printer sharing exception for the current firewall profile.

```
' ------ SCRIPT CONFIGURATION ------
 Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy.CurrentProfile
' ------ END CONFIGURATION ---------
Set Services = Policy.Services
Set Service = Services.Item(0)
Service.Enabled = TRUE
WScript.Echo "Setting enabled"
```

How It Works

Before the Windows Firewall was available through Service Pack 1 for Windows Server 2003, enabling file and printer sharing meant that you needed to manually open UDP ports 137 and 138, TCP ports 139 and 445, and the ICMP echo message. In the Windows Firewall, you can simply enable or disable the preconfigured file and printer sharing exception in the GUI, the Registry, or using netsh or VBScript.

Like the remote administration exception, file and printer sharing opens several well-known ports that are often used by malicious users to engage in network attacks. You should therefore configure the scope of the file and printer sharing exception carefully to ensure that only authorized users can access these ports on your Windows Server 2003 computer.

Though both the remote administration and file and printer sharing exceptions open TCP port 445, you can enable and disable them independently of each other. However, enabling the file and printer sharing exception will enable ICMP echo messages even if you've disabled the ICMP exception.

In addition to enabling the file and printer sharing exception in Group Policy, you need to specify the IPv4 addresses that are permitted to make remote administration connections. As with other Windows Firewall settings, you can use `LocalSubnet` to specify the local subnet, * to specify all hosts, or a custom list of addresses. For IPv6 addresses, you can only specify `LocalSubnet` or *; you can't create a custom exception list.

■ **Note** If you disable the file and printer sharing exception without allowing local exceptions as described in Recipe 3-5, local administrators will not be able to enable the exception on any computers that they administer.

See Also

- Recipe 3-8 to configure the remote administration exception
- Eric Cross's Networking: "Configure Windows Firewall Settings for File and Printer Sharing" (`http://ecross.mvps.org/howto/firewall.htm`)
- Microsoft TechNet: "Help: Enable or Disable the File and Printer Sharing Exception" (`http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/267c6000-957e-4fb4-8698-e41d4439fb58.mspx`)

3-10. Configuring Remote Assistance Through the Windows Firewall

Problem

You want to configure the Windows Firewall to allow Remote Assistance on a Windows Server 2003 computer.

Solution

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Exceptions tab, place a check mark next to Remote Assistance.
5. To define the scope of the exception, click on Edit, followed by Change Scope, and select from one of the following three options:
   - Any computer (including those on the Internet).
   - My network (local subnet).
   - Custom list. For this option, enter a single IP address using the syntax `192.168.1.151`, or enter a range of addresses using the network ID of the range followed by its subnet mask, such as `192.168.1.1/255.255.255.0`. Separate multiple entries using a comma.
6. Click OK when you're finished.

Using a Command-Line Interface

The following command enables Remote Assistance requests to pass through the Windows Firewall

```
> netsh firewall set service type = remotedesktop mode = ENABLE
```

Using Group Policy

Tables 3-19 and 3-20 show the settings that control the Remote Assistance exception within the Windows Firewall in the domain and standard profiles respectively.

Table 3-19. *Configure Remote Assistance Exception—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow remote desktop exception
Value	`Enabled` to allow incoming remote desktop traffic. `Disabled` to prevent it.

Table 3-20. *Configure Remote Assistance Exception—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow remote desktop exception
Value	`Enabled` to allow incoming remote desktop traffic. `Disabled` to prevent it.

Using VBScript

This code allows Remote Assistance to traverse the Windows Firewall.

```
Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy.CurrentProfile
 Set Port = CreateObject("HNetCfg.FwOpenPort")

 Port.Port = 3389
 Port.Name = "Remote Desktop"
 Port.Protocol = NET_FW_IP_PROTOCOL_TCP
 Port.Enabled = TRUE

 set Ports = Policy.GloballyOpenPorts
 addedPorts = Ports.Add(Port)
WScript.Echo "Ports configured."
```

How It Works

Remote Assistance is a feature in Windows XP and Windows Server 2003 that lets you share control of your computer with another user. Unlike a Terminal Services session, this grants you direct access to view and manipulate a user's desktop to offer them support. Creating a Remote Assistance connection requires the permission of the user to whose computer you're trying to connect.

To enable Remote Assistance connections prior to Windows Server 2003 Service Pack 1, you needed to manually configure the following program exceptions:

- `%WINDIR%\PCHealth\HelpCtr\Binaries\Helpsvc.exe`
- `%WINDIR%\PCHealth\HelpCtr\Binaries\Helpctr.exe`
- `%WINDIR%\SYSTEM32\Sessmgr.exe`

Additionally, you needed to enable access to TCP port 135.

On Windows Server 2003 computers with Service Pack 1 installed, you simply need to enable the preconfigured Allow Remote Desktop Exception setting; there is not a separate exception to enable for Remote Assistance. As with other exceptions, you should restrict the scope of this exception to protect your systems against attacks targeted at the well-known Remote Procedure Call (RPC) port, TCP 135.

See Also

- Recipes 3-3 and 3-4 for more on configuring program and port exceptions
- Microsoft TechNet: "Configuring System Service Exceptions" (`http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/Operations/9a29df7b-235a-42fd-9c25-13f6be94ad9a.mspx`)

3-11. Configuring UPnP Through the Windows Firewall

Problem

You want to configure the Windows Firewall to allow Universal Plug and Play (UPnP) on a Windows Server 2003 computer.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Exceptions tab, place a check mark next to UPnP Framework.
5. To define the scope of the exception, click on Edit, followed by Change Scope, and select from one of the following three options:
   - Any computer (including those on the Internet).
   - My network (local subnet).
   - Custom list. For this option, enter a single IP address using the syntax `192.168.1.151`, or enter a range of addresses using the network ID of the range followed by its subnet mask, such as `192.168.1.1/255.255.255.0`. Separate multiple entries using a comma.
6. Click OK when you're finished.

Using a Command-Line Interface

The following command enables UPnP to pass through the Windows Firewall:

```
> netsh firewall set service type = upnp mode = ENABLE
```

Using Group Policy

Tables 3-21 and 3-22 show the settings that control the UPnP exception within the Windows Firewall in the domain and standard profiles respectively.

Table 3-21. *Configure UPnP Exception—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow UPnP framework exception
Value	`Enabled` to allow incoming UPnP traffic. `Disabled` to prevent it.

Table 3-22. *Configure UPnP Exception—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow UPnP framework exception
Value	`Enabled` to allow incoming UPnP traffic. `Disabled` to prevent it.

Using VBScript

This code allows UPnP traffic to traverse the Windows Firewall.

```
 Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy.CurrentProfile
 Set TcpPort = CreateObject("HNetCfg.FwOpenPort")
 Set UdpPort = CreateObject("HNetCfg.FwOpenPort")

 TcpPort.Port = 2869
 TcpPort.Name = "UPnP Framework"
 TcpPort.Protocol = NET_FW_IP_PROTOCOL_TCP
 TcpPort.Enabled = TRUE

 UdpPort.Port = 1900
 UdpPort.Name = "UPnP Framework"
 UdpPort.Protocol = NEW_FW_IP_PROTOCOL_UDP
 UdpPort.Enabled = True

 set Ports = Policy.GloballyOpenPorts
 addedPorts = Ports.Add(TcpPort)
 addedPorts = Ports.Add(UdpPort)
WScript.Echo "Ports configured"
```

How It Works

The Universal Plug and Play (UPnP) framework extends the functionality of Plug-and-Play devices to allow device discovery and driver installations over a local area network or the Internet. Using UPnP, a device can automatically connect to a network and discover other network devices, allowing them to communicate directly without requiring administrator configuration or intervention.

UPnP requires the ability to make unsolicited connections on TCP port 2869 and UDP port 1900. If you choose to enable UPnP on your network, be sure to restrict the scope of the exception to only authorized computers, using comma-separated IP addresses or the `LocalSubnet` parameter.

See Also

- Recipe 3-4 for more on configuring port exceptions
- Microsoft KB 886257: "How Windows Firewall Affects the UPnP Framework in Windows XP Service Pack 2"

3-12. Configuring Firewall Notifications

Problem

You want to control how the Windows Firewall notifies the user of a Windows Server 2003 computer about events relating to the Windows Firewall.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Exceptions tab, place a check mark next to Display a Notification When Windows Firewall Blocks a Program to enable Windows Firewall notification. Remove this check box to disable notifications.

Using a Command-Line Interface

The following command enables the Windows Firewall to notify the local user of any programs it blocks in both the standard and domain profiles:

```
> netsh firewall set notifications mode = ENABLE profile = ALL
```

Using Group Policy

Tables 3-23 and 3-24 show the settings that control program notifications within the Windows Firewall in the domain and standard profiles respectively.

Table 3-23. *Configure UPnP Exception—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Prohibit notifications
Value	`Enabled` to disallow notifications; `Disabled` to allow notifications.

Table 3-24. *Configure UPnP Exception—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Prohibit notifications
Value	`Enabled` to disallow notifications; `Disabled` to allow notifications.

Using the Registry

To configure an individual computer to allow the Windows Firewall to notify the local user when it blocks a particular program, set the following Registry value:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>\]
"DisableNotifications"=dword:0
```

Using VBScript

This code enables firewall notifications.

```
 Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy.CurrentProfile
 Policy.NotificationsDisabled = FALSE ' set this to TRUE to disable notifications
WScript.Echo "Notifications enabled"
```

How It Works

When most applications request an open port, Windows Firewall adds the program to the program exceptions list with the default status value of `Disabled`. If you enable this policy setting, notifications are not displayed, and the status value for the program exception remains `Disabled` until manually changed.

If you disable or do not configure this policy setting, Windows Firewall displays notification messages.

If the user is not a local administrator, the message informs them that they might need to contact a network administrator, which can alert the network administrator about possible malicious programs on the network.

If the user is a local administrator, and you have either enabled the Windows Firewall: Allow Local Program Exceptions setting or you have not configured the Windows Firewall: Define Program Exceptions setting, the notification message allows the user to specify whether to enable the application. If you disable the Windows Firewall: Define Program Exceptions setting, the user will not be notified unless the policy is enabled locally.

See Also

- Recipe 3-19 for more on auditing Windows Firewall events
- Microsoft TechNet: "Managing Windows Firewall Notifications" (`http://technet2.microsoft.com/WindowsServer/en/Library/b3440a22-ae9c-45a3-8a61-da3f8a2c791f1033.mspx`)
- Microsoft TechNet: "Known Issues for Managing Windows Firewall Notifications" (`http://technet2.microsoft.com/WindowsServer/en/Library/2e3c1981-39fb-4979-bd16-c38ec6bf29fb1033.mspx`)

3-13. Allowing IPSec Traffic

Problem

You want to allow IPSec traffic to pass through the Windows Firewall on a Windows Server 2003 computer.

Solution

Using Group Policy

Table 3-25 contains the Group Policy setting that allows IPSec traffic to bypass the Windows Firewall.

Table 3-25. *Configure IPSec Traffic Exception*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections`
Policy name	Windows Firewall: Allow authenticated IPSec bypass
Value	`Enabled` to allow authenticated IPSec traffic to bypass the Windows Firewall. `Disabled` to prevent it.

How It Works

By default, the Windows Firewall will allow Internet Key Exchange (IKE) packets to pass through the firewall on UDP ports 500 and 4500. You have the additional option to allow *all* IPSec-protected traffic to bypass the Windows Firewall. To configure this, you can use the Allow Authenticated IPSec Bypass Group Policy setting to allow incoming, unsolicited IPSec traffic to bypass the Windows Firewall.

In order for this feature to work effectively, you need to specify which computers should be allowed to communicate this way. You'll do this in the Group Policy setting listed in Table 3-25 by listing a Security Descriptor Definition Language (SDDL) string that contains a list of the computers or groups of computers that should be exempt from the Windows Firewall blocking rules. An SDDL string is formatted as follows:

```
O:DAG:DAD:(A;;RCGW;;;<SID>)
```

In this syntax, `<SID>` refers to the Security Identifier (SID) of the computer or group of computers to which this policy should apply.

You can obtain the SID of an object within Active Directory by using the getsid.exe utility from Windows Support Tools. To obtain the SID for a group of computers called `DOMAINPCS`, for example, you would use the following syntax:

```
getsid \\<domain_controller> DOMAINPCS \\<domain_controller> DOMAINPCS
```

Getsid will return a numeric SID that looks something like this:

```
S-1-5-21-3475798998-36396922571-9412747344-3157
```

You would then enter the following SDDL string into Group Policy in the `Define IPSec Peers to be exempted from firewall policy` text box:

```
O:DAG:DAD:(A;;RCGW;;; S-1-5-21-3475798998-36396922571-9412747344-3157)
```

To enter multiple computers or groups of computers, use a single SDDL string with the following syntax:

```
O:DAG:DAD:(A;;RCGW;;;<SID1>) (A;;RCGW;;;<SID2>) (A;;RCGW;;;<SID3>)...
```

■ **Caution** If you configure an SDDL in Group Policy and subsequently disable the Allow Authenticated IPSec Bypass setting, the SDDL will be deleted.

See Also

Microsoft TechNet: "Help: Allow IPSec Traffic to Bypass Windows Firewall" (`http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/c7dd775d-00e9-4957-beba-3d82f1d829ad.mspx`)

3-14. Controlling Broadcast and Multicast Traffic

Problem

You want to configure how the Windows Firewall responds to broadcast and multicast traffic on a Windows Server 2003 computer.

Solution

Using a Command-Line Interface

The following command enables the local computer to respond to broadcast or multicast traffic while using the domain profile:

```
> netsh firewall set multicastbroadcastresponse mode = ENABLE profile = DOMAIN
```

Using Group Policy

Tables 3-26 and 3-27 show the settings that control multicast and broadcast traffic behavior in the domain and standard profiles respectively.

Table 3-26. *Configure Multicast and Broadcast Traffic—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Prohibit unicast response to multicast or broadcast requests
Value	`Enabled` to disallow responses to broadcast/multicast traffic; `Disabled` to allow the local computer to respond.

Table 3-27. *Configure Multicast and Broadcast Traffic—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Prohibit unicast response to multicast or broadcast requests
Value	`Enabled` to disallow responses to broadcast/multicast traffic; `Disabled` to allow the local computer to respond.

Using the Registry

To configure an individual computer not to respond to multicast or broadcast traffic, set the following Registry value:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\
    Parameters\FirewallPolicy\<Profile>\]
"DisableUnicastResponsesToMulticastBroadcast"=dword:1
```

Using VBScript

This code disables unicast responses to broadcast or multicast traffic.

```
 Set Firewall = CreateObject("HNetCfg.FwMgr")
 Set Policy = Firewall.LocalPolicy.CurrentProfile
Policy. UnicastResponsestoMulticastBroadcastDisabled = TRUE
' set this to FALSE to enable broadcast/multicast responses
WScript.Echo "Unicast disabled"
```

How It Works

When a Windows Server 2003 computer sends a multicast or broadcast packet, it will drop any unicast packets sent in response to the broadcast that are not received within three seconds. By enabling the Prohibit Unicast Response to Multicast or Broadcast Requests setting, Windows Firewall will drop any packet received in response to broadcast or multicast traffic. (The default setting for this behavior is Not Configured.)

The exception to this setting is that Windows Firewall will permit traffic associated with a DHCP lease request. However, this setting can interfere with broadcast-based NetBIOS name resolution, since responses to a NetBIOS broadcast will be dropped, preventing the computer from resolving a NetBIOS name to an IP address using broadcasts. If your network uses WINS for NetBIOS name resolution, this problem will be alleviated.

See Also

Microsoft TechNet: "Computer Names Do Not Resolve When Used in a UNC Path" (`http://technet2.microsoft.com/WindowsServer/en/Library/b8b1438d-8871-406d-a366-75f1f3a815e91033.mspx`)

3-15. Resetting the Windows Firewall

Problem

You want to reset the Windows Firewall to its default protection settings.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.

4. From the Advanced tab, click the Restore Defaults button in the Default Settings section. Click Yes to acknowledge the warning message stating that this will delete all settings of Windows Firewall that you have made since Windows was installed.
5. Click OK when you're finished.

Using a Command-Line Interface

The following command resets the Windows Firewall to its default settings:

```
> netsh firewall reset
```

Using VBScript

This code restores the Windows Firewall to its default settings.

```
Set Firewall = CreateObject("HNetCfg.FwMgr")
Firewall.RestoreDefaults()
WScript.Echo "Default settings restored."
```

How It Works

As an ongoing administrative task, you should monitor and audit your firewall settings to ensure that any existing settings, such as port and program exceptions, are still applicable to your current environment. This ensures that your network is not exposed to any unnecessary risks caused by unused or out-of-date firewall exceptions being enabled.

By using the `RestoreDefaults()` method, you can erase all user-defined program and port exceptions that you've created, and restore the Windows Firewall to its default setting of Enabled with appropriate exceptions permitted.

■ **Note** As a precaution, you should take a Registry backup before performing this procedure.

See Also

- The `RestoreDefault()` method in the Windows Firewall
- Microsoft TechNet: "Restore Windows Firewall Default Settings" (`http://technet2.microsoft.com/WindowsServer/en/Library/22623f4a-4699-4a59-8365-97783d117bd81033.mspx`)

3-16. Configuring Per-Interface Protection

Problem

You want to enable or disable the Windows Firewall for individual network interface cards (NICs) installed in a Windows Server 2003 computer.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Advanced tab, remove the check mark next to the interface name that should not be protected by the Windows Firewall in the Network Connections Settings section. To re-enable Windows Firewall protection for a particular interface, place a check mark next to it.
5. Click OK when you're finished.

■ **Note** This option is only available if you have more than one NIC installed in your Windows Server 2003 computer. For a single-homed computer, you can enable or disable the Windows Firewall using the steps in Recipe 3-1.

Using a Command-Line Interface

The following command disables the Windows Firewall for the NIC associated with the `Local Area Connection`.

```
> netsh firewall set opmode mode = DISABLE interface = "Local Area Connection"
```

■ **Note** If you are enabling or disabling the Windows Firewall for an individual interface, you cannot specify either the `profile` or the `exceptions` switch.

How It Works

When you are working with a multi-homed Windows Server 2003 computer, you may need to enable or disable the Windows Firewall on a per-interface basis, rather than globally for the entire server. You may wish to enable unfettered connectivity on a private network, for example, or you may have a hardware-based firewall protecting the NIC attached to a public network such as the Internet. Even if this is the case, however, the Windows Firewall is able to coexist with third-party products, and might still be a useful tool to provide "defense in depth" for your Windows Server 2003 computers, since the Windows Firewall will take effect before any third-party applications you've installed.

When working from the command line, you cannot use the `profile=` or `exceptions=` parameters in conjunction with the `interface=` parameter. To configure exceptions for an individual interface, you can use the `netsh firewall set portopening` command, which we will discuss in the next recipe.

See Also

- Recipe 3-1 for more on enabling and disabling the Windows Firewall
- Microsoft TechNet: "Help: Understanding Windows Firewall Exceptions" (`http://technet2.microsoft.com/WindowsServer/en/Library/7a19b261-840a-449e-b2b3-38b136d7bd591033.mspx`)

3-17. Enabling Per-Interface Inbound Connectivity

Problem

You want to configure the Windows Firewall to allow external users to connect to the local computer.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Advanced tab, select the interface that should be listening for inbound traffic in the Network Connections Settings section. Click Settings to configure the applications that this interface should be listening for connections on.

5. The Windows Firewall is preconfigured to enable inbound traffic for any of the following applications:

   - FTP Server
   - Internet Mail Access Protocol (IMAP) version 3
   - Internet Mail Access Protocol (IMAP) version 4
   - Internet Mail Server (SMTP)
   - Post Office Protocol version 3 (POP3)
   - Remote Desktop
   - Secure Web Server (HTTPS)
   - Telnet Server
   - Web Server (HTTP)

   Place a check mark next to the protocol(s) that you want to enable for inbound traffic.

6. To create a new port for inbound connectivity, click on Add and follow these steps:

   a. Enter the name of the service or application in the Description of Service text box.

   b. Enter the DNS name or IP address of the computer that's hosting this service in the Name text box.

   c. Select the appropriate radio button next to TCP or UDP.

   d. Enter the external port number on which this application or service listens in the External Port Number for This Service text box.

   e. Enter the internal port number (if you are translating port numbers on your internal network, for example) in the Internal Port Number for This Service text box.

7. Click OK when you are finished.

Using a Command-Line Interface

The following command enables inbound connectivity on TCP port 80 for the `Local Area Network` interface:

```
> netsh firewall set portopening protocol = TCP port = 80 name = WEB
    mode = ENABLE interface = "Local Area Network"
```

■ **Note** You can also enable per-interface ICMP exceptions by using the `netsh firewall set icmpsettings` command with the `interface=` parameter.

How It Works

When you are working with a multi-homed server—a server containing more than one network interface card (NIC) and attached to more than one network segment—you may have Windows Firewall exceptions that are only applicable to a particular interface.

For example, you may have a Windows Server 2003 that is attached to a private network as well as the Internet, on which you've installed the Internet Information Server (IIS) to host your company's public website. In this case, the NIC that is connected to the Internet should be configured to listen for unsolicited incoming requests on the HTTP port, TCP port 80. However, you may wish to restrict access to this port to the public-facing NIC only, while the NIC attached to your private network should not accept unsolicited HTTP traffic. In this case, you'll need to configure only the Internet-connected NIC with an exception for the HTTP port; this exception should not apply to the NIC attached to the private network.

Because this is a common reason to enable per-connection Windows Firewall settings, the Windows Firewall Control Panel applet allows you to easily select several preconfigured services for per-connection exceptions, including the FTP service, telnet server service, IMAP, SMTP, and HTTP/HTTPS for Web traffic.

See Also

- Recipe 3-7 for more on configuring ICMP exceptions
- Microsoft TechNet: "Help: Add a System Service to the Windows Firewall Exceptions List" (`http://technet2.microsoft.com/WindowsServer/en/Library/34cfba8e-d564-4a8c-9f4c-58120bed441d1033.mspx`)

3-18. Configuring Firewall Logging

Problem

You want to control how the Windows Firewall logs information on a Windows Server 2003 computer.

Solution

Using a Graphical User Interface

1. Open the Network Connections applet.
2. Double-click on the Local Area Connection icon.
3. From the Advanced tab, click Settings. This will launch the Windows Firewall Control Panel applet.
4. From the Advanced tab, click the Settings button in the Security Logging section.

5. In the Logging Options section, place a check mark next to one or both of the following settings:
   - Log dropped packets
   - Log successful connections
6. In the Log File Options section, specify the filename and directory path of the log file in the Name text box. Specify the maximum size of the file in the Size Limit (KB) window.
7. Click OK when you're finished.

Using a Command-Line Interface

The following command enables Windows firewall logging for dropped packets with a maximum file size of 8,192 bytes:

```
> netsh firewall set logging c:\logs\wfirewall.log droppedpackets = ENABLE
    maxfilesize = 8192
```

Using Group Policy

Tables 3-28 and 3-29 contain the Group Policy settings that dictate whether Windows Firewall should log firewall activity for the domain and standard profiles respectively.

Table 3-28. *Configure Firewall Logging—Domain Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Domain Profile`
Policy name	Windows Firewall: Allow logging
Value	`Enabled` to turn on logging. `Disabled` to turn off logging.

Table 3-29. *Configure Firewall Logging—Standard Profile*

Path	`Computer Configuration\Administrative Templates\Network\Network Connections\Windows Firewall\Standard Profile`
Policy name	Windows Firewall: Allow logging
Value	`Enabled` to turn on logging. `Disabled` to turn off logging.

How It Works

When you enable logging, the Windows Firewall creates a text file containing information about any packets that it drops or accepts. By default, the file is stored as `c:\windows\pfirewall.log` with a maximum file size of `4096` bytes (4MB). Once the file reaches its maximum size, Windows will begin logging to a new file called `pfirewall.log.1`.

The log file itself is recorded in the W3C Extended Log File Format. This is an industry standard format that will allow you to analyze the log file in a simple text editor like Notepad, or to import log information into a database for analysis with third-party tools. The file begins with a header that lists the fields that are recorded, as follows:

```
#Version: 1.5
#Software: Microsoft Windows Firewall
#Time Format: Local
#Fields: date time action protocol src-ip dst-ip src-port dst-port size
    tcpflags tcpsyn tcpack tcpwin icmptype icmpcode info path
```

After the header, the body of the log file begins. Each line in the body records a packet that was either dropped or allowed to pass through the firewall. Each field in the body of the file corresponds to the title listed in the header; a dash (-) indicates that there was no information to record in that field. For example, a log file entry that records a dropped packet would resemble the following:

```
2005-07-15 11:26:36 DROP UDP 10.1.7.30 255.255.255.255
    1522 14000 104 - - - - - - - RECEIVE
```

■ **Note** You can also analyze the Windows Firewall log file with the Microsoft Log Parser, which we'll discuss further in Recipe 3-19.

See Also

- Recipe 3-19 for more on auditing Windows Firewall events.
- Microsoft TechNet: "W3C Extended Log File Format (IIS 6.0)" (`http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/676400bc-8969-4aa7-851a-9319490a9bbb.mspx`).
- Microsoft TechNet: "Log Parser 2.2" (`http://www.microsoft.com/technet/scriptcenter/tools/logparser/default.mspx`). This article discusses the free Microsoft Log Parser tool.

3-19. Auditing Windows Firewall Events

Problem

You want to view and manage the Windows event log entries created by the Windows Firewall.

Solution

Using a Graphical User Interface

1. Open the EventCombMT Utility. Be sure that your domain name is listed in the Domain text box.
2. Right-click in the Select to Search/Right-click to Add text box, and select Add Single Server.
3. Type the name of your server in the Server Name text box and select Add Server, or click Browse to select the server from My Network Places. Click Close when you've selected the name of the server.
4. In the Choose Log Files to Search section, place a check mark next to Security.
5. In the Event Types section, place a check mark next to the following event types:
   - Error
   - Informational
   - Warning
   - Success Audit
   - Failure Audit
   - Success
6. Enter the following event IDs in the From and To text boxes:
   - 848
   - 861
7. Click on Search to begin querying for event log entries that match these criteria. By default, the results will be stored in a comma-separated values (CSV) file called `<ComputerName>-Security_LOG.txt`.

Using Group Policy

Tables 3-30 and 3-31 contain the Group Policy settings that enable the Windows Event viewer to track events related to the Windows Firewall. As with other Windows auditing events, auditing `Success` events means that the Event Viewer will create an entry if someone attempts to perform a particular action and is able to do so. Auditing `Failure` events will create an Event Viewer entry if someone attempts an action and is unsuccessful.

Table 3-30. *Audit Process Tracking Settings*

Path	`Computer Configuration\Windows Settings\Security Settings\Local Policies\Security Options`
Policy name	Audit Process Tracking
Value	`Success`, `Failure`

Table 3-31. *Audit Policy Change Settings*

Path	`Computer Configuration\Windows Settings\Security Settings\Local Policies\Security Options`
Policy name	Audit Policy Change
Value	`Success, Failure`

■ **Note** If you are not in an Active Directory environment, you can enable auditing on an individual Windows Server 2003 computer by using the Local Security Policy MMC snap-in available in the Administrative Tools folder.

Using a Command-Line Interface

The following command uses the logparser.exe command-line utility to parse the `Security` log for events relating to the Windows Firewall:

```
> LogParser "SELECT TimeGenerated, SourceName,
EventCategoryName, Message INTO report.txt FROM Security WHERE
EventID > 847 AND EventID < 862" -resolveSIDs:ON
```

■ **Note** Notice that logparser.exe doesn't support the "greater than or equal to" or "less than or equal to" operators (<= and >=), so instead we're searching for event IDs that are greater than 847 and less than 862.

How It Works

One of the greatest challenges of system administration is the maintenance and analysis of the auditing data that is generated by a network or domain full of Windows computers. In the spirit of the "tree falling in the forest" question, many administrators find themselves wondering if a security breach or configuration error has actually taken place somewhere on their network and they simply haven't spotted the event log entry that will alert them to it.

To help combat this, Microsoft has released a number of free utilities to help you to analyze and monitor event log data. One of these tools, EventCombMT, has been around since the days of Windows NT 4.0 and is used to collect Event Viewer entries from numerous computers into a single location to allow you to monitor and view them more efficiently. EventCombMT is now available within a larger bundle of free tools called the Account Lockout and Management Tools, available for download from `http://www.microsoft.com/downloads/details.aspx?FamilyID=7AF2E69C-91F3-4E63-8629-B999ADDE0B9E&displaylang=en`.

Another free tool that hasn't received quite the same publicity as EventCombMT is the Microsoft Log Parser. This was first released as a free Resource Kit utility with very little supporting documentation, but it has developed a large grassroots following with strong Internet community support. In a nutshell, the Microsoft Log Parser allows you to use a SQL-like query engine to extract data from any number of common log file sources, including Windows event logs,

IIS logs, firewall logs in the W3C standard log file format, and even metadata stored by the operating system about files and folders on a Windows hard drive. In this case, the Windows Firewall creates a log file in the W3C file format that can be easily queried by the Microsoft Log Parser.

While you cannot parse a log file or the Event Viewer using Group Policy, you can (and should) use a Group Policy object (GPO) to enable event auditing for your Windows computers wherever possible. This ensures that all of your Windows Server 2003 computers are recording the same types of data, to ensure that no security-related events go unnoticed because they are not being audited.

See Also

- Microsoft TechNet: "Events and Errors Message Center" (`http://www.microsoft.com/technet/support/ee/ee_advanced.aspx`).
- EventID.net (`www.eventid.net`): This site provides detailed information on Windows Event Viewer entries.

The Unofficial Log Parser Support Site (`www.logparser.com`).