



Deploying VPN Connections by Using Powershell and Group Policy

Microsoft Corporation

Published: August 19, 2009

Abstract

This article describes how to use Group Policy, Powershell and the Remote Access Service (RAS) application programming interfaces (APIs) to configure and deploy virtual private network (VPN) connection settings to client computers. The solution also describes how the Task Scheduler service can be used to configure scripts or programs that are run whenever a VPN connection is made to the VPN server. The advantage of this solution is that it is not platform specific, and can be used on all of the currently supported versions of Windows.



This document is provided for informational purposes only and Microsoft makes no warranties, either express or implied, in this document. Information in this document, including URL and other Internet Web site references, is subject to change without notice. The entire risk of the use or the results from the use of this document remains with the user. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, Windows, Windows Server, ActiveX, and Internet Explorer are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

[Deploying VPN Connections by Using PowerShell and Group Policy 5](#_Toc238955809)

[Solution Overview 5](#_Toc238955810)

[Solution Architecture 6](#_Toc238955811)

[Creating the GPO to Deploy the Script 7](#_Toc238955812)

[The Sample PowerShell Script and Data File 9](#_Toc238955813)

[Considerations for Using the PowerShell Script 17](#_Toc238955814)

[Signing PowerShell scripts 17](#_Toc238955815)

[Configuring post-connect actions by using Task Scheduler 18](#_Toc238955816)

# Deploying VPN Connections by Using PowerShell and Group Policy

Most organizations support some form of remote access to enable their users to access network resources when they are away from the network. Virtual private networking (VPN) over the Internet or dial-up connections can be used to provide almost the same access as when users are physically connected at the office.

After you have configured a VPN or dial-up server on your network periphery, you must deploy settings to your client computers that enable them to establish VPN connections to the server with minimal user interaction. The most common types of failures of VPN connections are due to improperly configuring the connection settings on the client computer. The user might configure an invalid VPN tunnel type, the wrong authentication type, the wrong phone number, and so forth. The configuration of the remote access server setup is typically maintained by a network administrator. Users are not usually aware of the details of the configuration, and thus are unlikely to configure a connection correctly on their own. Even if the administrator supplies clear, easy-to-follow directions to users, manually configuring the VPN settings can lead to excessive support calls to your help desk.

This guide provides a solution that enables an administrator to configure and deploy VPN connections to client computers that are ready to use, with no need for the user to configure any details.

Note

This guide discusses one technique that can be used. Its primary purpose is to illustrate how a remote access connection can be created programmatically. Another technique often used is to use the Connection Manager Administration Kit (CMAK) to create a Connection Manager profile that can be deployed to your clients. For more information about CMAK, see [Connection Manager Administration Kit](http://go.microsoft.com/fwlink/?linkid=55986) (http://go.microsoft.com/fwlink/?linkid=55986) in the Windows Server Technical Library.

# Solution Overview

This article describes a solution that uses three features of Windows:

1. [Group Policy](http://go.microsoft.com/fwlink/?linkid=55625) (http://go.microsoft.com/fwlink/?linkid=55625).

Group Policy enables administrators to push domain settings to managed computers. As a part of this solution, Group Policy is used to configure a PowerShell script as a logon script that creates a VPN connection. The PowerShell script runs every time the user logs on to the corporate network, and does not require manual intervention by the users. The network administrator defines the configuration, and then pushes them to the users by using Group Policy.

2. [PowerShell scripting](http://go.microsoft.com/fwlink/?linkid=160176) (http://go.microsoft.com/fwlink/?linkid=160176).

PowerShell embeds C# code that is run as a part of the script. This article describes a sample PowerShell script that has C# code and calls the RAS API entry points to create and configure VPN connections settings. The input to the PowerShell script is an Extensible Markup Language (XML) file containing the configuration for the VPN connection. As a network administrator, you must customize the XML file for the VPN services in your organization.

Note

PowerShell is included by default in computers that are running Windows Vista and later versions of Windows. It is not included with Windows XP, but can be downloaded and installed from the Microsoft Download Center. For more information, see [PowerShell Scripting](http://go.microsoft.com/fwlink/?linkid=160176) (http://go.microsoft.com/fwlink/?linkid=160176).

3. [Remote Access Service (RAS) application programming interfaces (APIs)](http://go.microsoft.com/fwlink/?linkid=160178) (http://go.microsoft.com/fwlink/?linkid=160178).

The remote access functionality built into Windows is accessible to programs through the RAS API set. This set of functions enables a program to create and configure remote access connections directly, without having to use the graphical Connection Manager interface.

These features are combined in the example documented in this guide to seamlessly configure VPN connection settings and deploy them for use. The solution describes how the Task Scheduler service can be used to configure scripts or run programs whenever a VPN connection is made to the VPN server. This solution can be used on any computer that is running Windows XP or a later version of Windows.

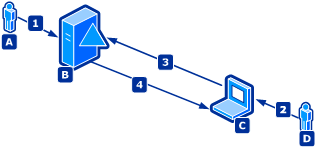
Note

Although this document only describes using the technique to create VPN connections, you could similarly use the RAS APIs in a PowerShell script to create Point-to-Point Protocol over Ethernet (PPPoE) or dial-up connections as well.

# Solution Architecture

The solution involves a network that uses an Active Directory domain to manage its client computers. Active Directory provides centralized user and computer account management, authentication of users and computers, and Group Policy to automatically configure computer and user settings as configured by the network administrator. Active Directory uses servers called domain controllers to provide these services to the client computers on the network.

The following diagram depicts the components involved in provisioning a VPN connection on a client computer.



1. The administrator (A) configures a Group Policy object (GPO) that deploys a PowerShell script to run at user logon. The GPO is stored on the domain controller (B). To create and configure a GPO, use the Group Policy Management Console (GPMC) Microsoft Management (MMC) snap-in. This GPO is configured to run a PowerShell script that creates a VPN connection, as described in the accompanying data file.

2. The user (D) logs on to the client computer (C).

3. As part of the log on process, the client computer communicates with the domain controller and identifies both itself and the logged-on user.

4. The client computer retrieves any GPOs that apply to it or the logged on user and apply the settings in the GPOs. In this case, the GPO contains the PowerShell script, which is run on the client computer, and then the VPN connection is created and configured.

Later, when the user takes the portable computer to a remote site, he or she can connect to the corporate network by using the VPN connection created earlier. The next section contains step-by-step instructions about how to create the GPO and configure it to run the script on client computers.

# Creating the GPO to Deploy the Script

This topic describes the steps involved in configuring a PowerShell script that creates a remote access connection. Note that this step is not specific to remote access and is like configuring any other PowerShell script. The example illustrates configuring the script to run at logon time. Group Policy also enables you to run scripts at user logoff, and at computer startup and shutdown.

To configure a PowerShell script as a logon script

|  |
| --- |
| 1. Share the PowerShell script and its associated data XML file on a network share that can be accessed and read by all users that are assigned the script. In this example, the files are placed in a network share called \\corpserver.contoso.com\scripts. Read and Execute permissions are granted to the Authenticated Users group.  2. Open Group Policy Management Console (GPMC), and then use it to find and edit the GPO that you want to modify. Create a new GPO if required. The example in the following figure shows a GPO named “VPN Connection Policy Object.” Ensure that the GPO is linked to either the domain or another container that contains the user accounts that you want to run the script.    3. In Group Policy Management Editor, in the navigation pane, expand User Configuration, expand Policies, expand Windows Settings, and then click Scripts (Logon/Logoff).  4. In the details pane, right-click Logon, and then click Properties.    5. On the Logon Properties dialog box, on the Scripts tab, click Add.  6. In the Add a Script dialog box, in the Script Name text box, type:  powershell.exe  In the Script Parameters text box, type the following command, substituting the appropriate network path to your copy of the files.  -noninteractive -command \\corpserver.contoso.com\scripts\Create-Conn.ps1 \\corpserver.contoso.com\scripts\VPNSettings.xml  Note  Even though the input above might wrap to more than one line when displayed here, enter it as a single line.  When you have entered the paths, click OK twice to save your settings. |

After this GPO is updated on a computer, then every time that a domain user logs on to the computer the configured PowerShell script is run.

Note

When editing a GPO on a computer that is running Windows 7 or Windows Server 2008 R2, there is a PowerShell Scripts tab on which you can directly specify a script name and parameters without having to identify the powershell.exe program. However, this option works only on client computers that are running Windows 7 or Windows Server 2008 R2. This guide uses the more general Scripts tab to be compatible with computers that are running earlier versions of Windows.

# The Sample PowerShell Script and Data File

The version of this guide that is hosted on the [Microsoft Download Center](http://go.microsoft.com/fwlink/?linkid=160558) (http://go.microsoft.com/fwlink/?linkid=160558) includes a sample PowerShell script and XML data file. The script reads the XML file as input, and uses it to specify how the VPN connection is to be configured. You only need to modify the XML file to add new VPN connections or change existing VPN connections on your client computers.

Disclaimer

The sample script and data file described in this guide are not supported under any Microsoft standard support program or service. The sample script and data file are provided AS IS without warranty of any kind. Microsoft further disclaims all implied warranties including, without limitation, any implied warranties of merchantability or of fitness for a particular purpose. The entire risk arising out of the use or performance of the sample scripts and documentation remains with you. In no event shall Microsoft, its authors, or anyone else involved in the creation, production, or delivery of the scripts be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use of or inability to use the sample scripts or documentation, even if Microsoft has been advised of the possibility of such damages.

Caution

Test the script and its accompanying data file thoroughly in a test environment, and customize it to meet the requirements of your organization before attempting to use it in a production capacity.

This topic describes the structure of the XML data file as required by the sample PowerShell script. If you modify the script, you might have to modify the structure of the data file to match your changes.

The sample script uses the publicly documented RAS APIs and data structures to create or modify VPN connection settings. The C# code embedded in the script parses the input XML data file and loads them as settings for a VPN connection. The script then constructs a [RASENTRY](http://go.microsoft.com/fwlink/?linkid=160268) (http://go.microsoft.com/fwlink/?linkid=160268) data structure, translates it into Win32 form, and then calls the [RasSetEntryProperties](http://go.microsoft.com/fwlink/?linkid=160272) (http://go.microsoft.com/fwlink/?linkid=160272) function to create the VPN connection from the data in the RASENTRY structure.

The XML data file contains the following elements:

 An XML header

 An XML root element named RemoteAccessEntries. This contains one or more child elements named RemoteAccessEntry.

 Each RemoteAccessEntry element corresponds to a single VPN connection profile, and contains child elements that specify the settings for the connection.

The following table describes each of the elements supported in the XML data file. The settings for an individual VPN connection are mostly optional; if not present, the PowerShell script uses a default value.

|  |  |  |
| --- | --- | --- |
| Element name | Meaning | Values |
| RemoteAccessEntries | This element is the root node of the XML file. It contains settings for one or more VPN connection profiles. | No values; contains only child elements.  Required |
| RemoteAccessEntry | Child node of RemoteAccessEntries. This element along with its child elements corresponds to the settings for one VPN entry to be created. There can be one or more RemoteAccessEntry elements under RemoteAccessEntries. | No values; contains only child elements.  Required |
| Name | Specifies the name of the VPN client entry as it appears in the list of VPN connections on the client computer. | A text string. It must not contain any of the following characters: \ / ; \* ? < > | "  Required |
| SharedProfile | Specifies whether the profile is for the currently logged on user only, or shared by all users of the computer.  Note  A shared VPN profile can be used to log on to a remote domain, establishing the VPN connection before the user logon process begins. | ****** True – Shared by everyone on client computer  ****** False – Available only for the currently logged on user on the client computer  Default: True |
| ConnectionType | Specifies the type of connection to the remote network. | ****** Dialup – Dial-up networking connection  ****** VPN – Virtual private network connection  ****** Direct – Direct connection  ****** Internet – Custom dialer  ****** Broadband – PPPoE connection  Default: VPN |
| Negotiate\_IPv4 | Specifies whether IPv4 must be negotiated between the VPN client and VPN server. | ****** True – Negotiate IPv4  ****** False – Do not negotiate IPv4  Default: True |
| Negotiate\_IPv6  Note  This option does not apply to computers that are running Windows XP or Windows Server 2003. | Specifies whether IPv6 must be negotiated between the VPN client and VPN server. | ****** True – Negotiate IPv6  ****** False – Do not negotiate IPv6  Default: True |
| VpnStrategy | Specifies the order in which the VPN tunnels types are tried. | ****** PPTPOnly – Try only Point-to-Point Tunneling Protocol (PPTP)  ****** L2TPOnly – Try only Layer Two Tunneling Protocol (L2TP)  ****** SSTPOnly – Try only Secure Socket Tunneling Protocol (SSTP)  ****** IKEv2Only – Try only Internet Key Exchange version 2 (IKEv2)  ****** PPTPWithSSTP – Try PPTP, and if fails, try SSTP  ****** L2TPWithSSTP – Try L2TP, and if fails, try SSTP  ****** IKEv2WithSSTP – Try IKEv2, and if fails, try SSTP  Default: IKEv2withSSTP |
| RouteIPv4TrafficOverRAS | Specifies whether the VPN connection becomes the IPv4 default gateway on the client computer for the duration of the connection. | ****** True – Add a default gateway on the VPN connection  ****** False – Do not add default gateway on the VPN connection  Default: True |
| RouteIPv6TrafficOverRAS | Specifies whether the VPN connection becomes the IPv6 default gateway on the client computer for the duration of the connection. | ****** True – Add a default gateway on the VPN connection  ****** False – Do not add a default gateway on the VPN connection  Default: True |
| ShowUsernamePassword | Specifies Specifies whether to show the Username and Password fields in the remote access connection manager interface. | ****** True – Show the Username and Password fields  ****** False – Hide the Username and Password fields  Default: True |
| ShowDomain | Specifies whether to show the Domain field in the remote access connection manager interface. | ****** True – Show the Domain field  ****** False – Hide the Domain field  Default: True |
| ShowDialProgressBar | Specifies whether to show the connection progress during the establishment of the connection. | ****** True – Show the connection progress  ****** False – Hide the connection progress  Default: True |
| RequireCHAP | Specifies whether CHAP authentication is required for the VPN connection.    We recommend that you do not use CHAP authentication. | ****** True – Negotiate CHAP  ****** False – Do not negotiate CHAP  Default: True |
| RequireMSCHAPv2 | Specifies whether Microsoft Challenge Handshake Authentication Protocol version 2 (MS CHAP v2) authentication is required for the VPN connection.    We recommend that you consider EAP instead of MS CHAP v2 because EAP is not dependent on passwords. | ****** True – Negotiate MS CHAP v2  ****** False – Do not negotiate MS CHAP v2  Default: True |
| RequireEAP | Specifies whether EAP authentication is required for the VPN connection.    We recommend the use of EAP authentication over CHAP, MS CHAP v1, or MS CHAP v2. | ****** True – Negotiate EAP  ****** False – Do not negotiate EAP  Default: False |
| RequireEncryptedPassword | Specifies whether the VPN connection requires either CHAP, MS CHAP v1, or MS CHAP v2 authentication.    We recommend the use of EAP authentication over CHAP, MS CHAP v1, or MS CHAP v2. | ****** True – Require the use of CHAP, MS CHAP v1, or MS CHAP v2 authentication  ****** False – Do not require the use of CHAP, MS CHAP v1, or MS CHAP v2 authentication  Default: True |
| RequireMsEncryptedPassword | Specifies whether the VPN connection requires MS CHAP v1 or MS CHAP v2 authentication.    We recommend the use of EAP authentication over MS CHAP v1 or MS CHAP v2. | ****** True – Require the use of MS CHAP v1 or MS CHAP v2 authentication  ****** False – Do not require MS CHAP v1 or MS CHAP v2 authentication  Default: True |
| DontCacheRASCredentialsInCredman | Specifies whether user credentials used by the VPN connection are stored in Windows Credential Manager. The user does not have to enter additional credentials for subsequent resource access if the VPN credentials are valid for access.  Credentials are stored on a per user basis. Credentials cached by one user cannot be used by another user. | ****** True – Do not cache user credentials  ****** False – Cache user credentials  Default: False |
| ReconnectIfDropped | Specifies whether the VPN connection should be reestablished if it is unexpectedly disconnected. | ****** True – Reconnect if VPN connection drops  ****** False – Do not reconnect if VPN connection drops  Default: True |
| ProxySettings | Specifies the proxy settings for the current VPN connection. It has child elements each specifying a part of configuration for Internet Explorer. | No values; contains only child elements. |
| UseManualProxy (child element of ProxySettings) | Specifies whether the proxy setting in Internet Explorer is configured by this connection profile. | ****** True – Use manual proxy specified in the ManualProxyServer element  ****** False – Do not change the proxy setting in Internet Explorer  Default: False |
| UseAutoProxy (child element of ProxySettings) | Specifies whether the proxy setting is configured automatically. | ****** True – Use automatic proxy detection  ****** False – Do not use automatic proxy detection  Default: False |
| UseAutoConfigurationScript (child element of ProxySettings) | Specifies whether to use a proxy auto-configuration script. | ****** True – Use proxy auto-configuration script  ****** False – Do not use proxy auto-configuration script  Default: False |
| ManualProxyServer (child element of ProxySettings) | Specifies the manual proxy server name for the VPN connection in Internet Explorer. | A text string that identifies the fully qualified domain name (FQDN) or IP address of the proxy server to use. |
| ProxyOverride (child element of ProxySettings) | Specifies Domain Name System (DNS) names or addresses for which the proxy should not be used. | A comma-separated list of resolvable DNS names or IP addresses. |
| ByPassProxyForLocal (child element of ProxySettings) | Specifies whether the proxy server is bypassed for local subnet addresses. | ****** True – Bypass proxy for local subnet addresses  ****** False – Do not bypass the proxy server for local subnet addresses |
| AutoConfigurationScript (child element of ProxySettings) | Specifies the Web Proxy Auto Discovery (WPAD) configuration script. | The universal naming convention (UNC) file path to the auto-configuration script. |
| Destination | Specifies a destination IP address or phone number for the VPN connection. It has one child element, DestinationAddress, which contains the destination information. You can specify one or more Destination elements, each with its own DestinationAddress child element. | No values; contains only child elements. |
| DestinationAddress (child element of Destination) | Specifies the IP address in case of VPN connections, phone number in case of dial-up connections and service names in case of PPPoE connections. | ****** An IPv4 or IPv6 address of a VPN server on the network  ****** The phone number of a dial-up server  ****** The service name provided by your PPPoE service provider |

# Considerations for Using the PowerShell Script

This topic addresses some of the additional steps to consider before deploying this as a solution in a production environment.

## Signing PowerShell scripts

PowerShell scripts must be signed to run on client computers that have a default PowerShell configuration. Running unsigned scripts requires a configuration change. We recommend that you change the signing requirement only on test computers on which you develop and test your PowerShell scripts. Do not remove the signing requirement on production client computers because it increases the potential attack surface of the computer to malicious scripts.

By default, the PowerShell scripting engine runs only scripts that are signed by using a certificate that was issued by a trusted certification authority (CA). If you have a CA in your organization, such as one running Active Directory® Certificate Services (AD CS), then you can configure it to automatically add the root CA certificate to the Trusted Root Certification Authorities store on your client computers automatically. Then you can issue a code signing certificate to the user who is tasked with signing the scripts used in your organization. By only signing tested and approved scripts, and by preventing client computers from running any unsigned scripts you help to maintain the security of the network.

For more information about signing PowerShell scripts, see [Windows PowerShell: Sign Here Please](http://go.microsoft.com/fwlink/?linkid=160357) (http://go.microsoft.com/fwlink/?linkid=160357) in TechNet Magazine. For more information about code signing in general, see [Introduction to Code Signing](http://go.microsoft.com/fwlink/?linkid=59273) (http://go.microsoft.com/fwlink/?linkid=59273) on MSDN. For more information about setting up your own CA, see [Active Directory Certificate Services](http://go.microsoft.com/fwlink/?linkid=136444) (http://go.microsoft.com/fwlink/?linkid=136444) in the TechNet Library.

## Configuring post-connect actions by using Task Scheduler

Once you have your VPN connections deployed to all of your client computers, you might want to no longer run the PowerShell script as a logon script. Instead, if you are running Windows Vista or Windows 7, then you can configure a scheduled task that runs every time the user connects by using the VPN. This task can run the script to update the details of the VPN connections to ensure that they are current.

When a user connects by using a VPN connection, the remote access software automatically logs events that indicate the connection stages and connect or failure status. For example, after a VPN is successfully connected, the remote access service logs Event ID 20225 – “RAS Connection Establishment” in the Application event log.

Task Scheduler can be used to start an action whenever a specific event ID appears in the event log. In this case, you can cause the PowerShell script to be run whenever event ID 20225 appears in the Application event log.

The command that you schedule to run is similar to the following command:

Powershell.exe –NonInteractive –command \\vpnserver.contoso.com\scripts\create-conn.ps1 \\vpnserver.contoso.com\scripts\VPNsettings.xml

Substitute the path to your copy of the scripts on your server.

To create the scheduled task, run the following command. Although it wraps to multiple lines here, be sure to enter the command on a single line:

schtasks /create /F /TN "VPN Connection Update" /TR "Powershell.exe -NonInteractive -command \\vpnserver.contoso.com\scripts\create-conn.ps1 \\vpnserver.contoso.com\scripts\settings.xml" /SC ONEVENT /EC Application /MO " \*[System[(Level=4 or Level=0) and (EventID=20225)]] and \*[EventData[Data='Contoso VPN']] "

Substitute the correct paths for your environment. The “Level=4 or Level=0” parameter specifies Information events only. Also, note the last parameter, and change the name of the VPN connection to the friendly name of your VPN connection. This last clause ensures that the script only runs when you connect by using the specified VPN connection. Other connections do not start the script.

After you test the command, you can include it in a script that is deployed by using Group Policy similarly to the one that deploys the script in a logon script.