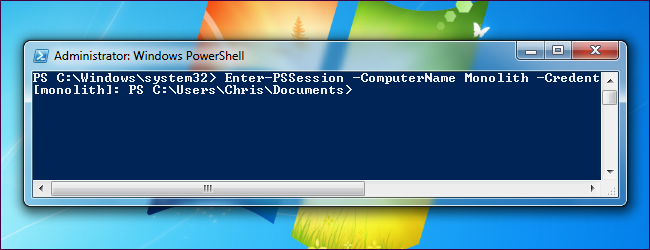
**[How to Run PowerShell Commands on Remote Computers](http://www.howtogeek.com/117192/how-to-run-powershell-commands-on-remote-computers/" \o "How to Run PowerShell Commands on Remote Computers)**

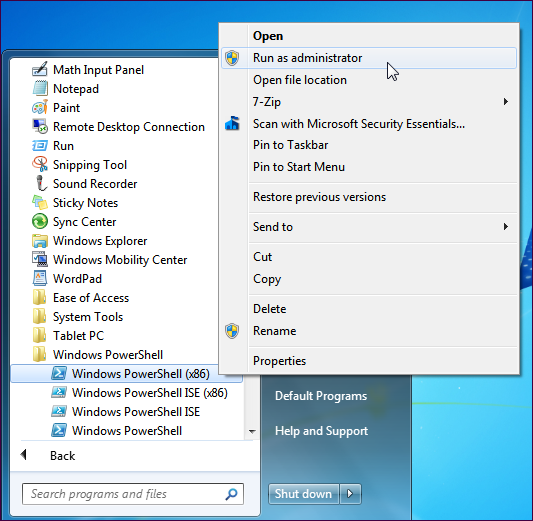


PowerShell Remoting allows you to run individual PowerShell commands or access full PowerShell sessions on remote Windows systems. It’s similar to SSH for accessing remote terminals on other operating systems.

PowerShell is locked-down by default, so you’ll have to enable PowerShell Remoting before using it. This setup process is a bit more complex if you’re using a workgroup – for example, on a home network — instead of a domain.

**Enabling PowerShell Remoting**

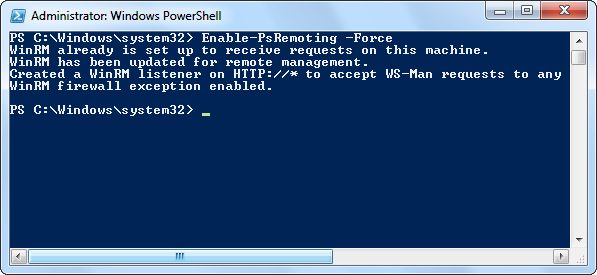
On the computer you want to access remotely, open a PowerShell window as Administrator – right click the PowerShell shortcut and select Run as Administrator.



To enable PowerShell Remoting, run the following command (known as a cmdlet in PowerShell):

Enable-PSRemoting -Force

This command starts the WinRM service, sets it to start automatically with your system, and creates a firewall rule that allows incoming connections. The -Force part of the command tells PowerShell to perform these actions without prompting you for each step.



**Workgroup Setup**

If your computers aren’t on a domain – say, if you’re doing this on a home network – you’ll need to perform a few more steps. First, run the **Enable-PSRemoting -Force** command on the computer you want to connect from, as well. (Remember to launch PowerShell as Administrator before running this command.)

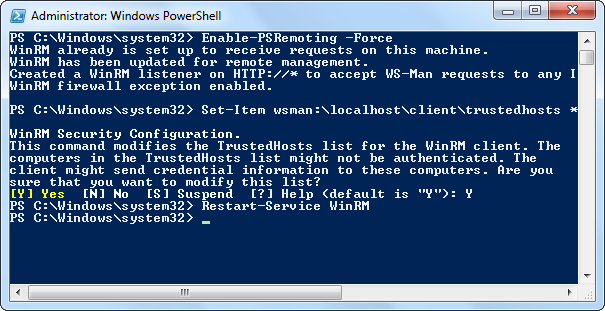
On both computers, configure the TrustedHosts setting so the computers will trust each other. For example, if you’re doing this on a trusted home network, you can use this command to allow any computer to connect:

Set-Item wsman:\localhost\client\trustedhosts \*

To restrict computers that can connect, you could also replace the \* with a comma-separated list of IP addresses or computer names.

On both computers, restart the WinRM service so your new settings will take effect:

Restart-Service WinRM

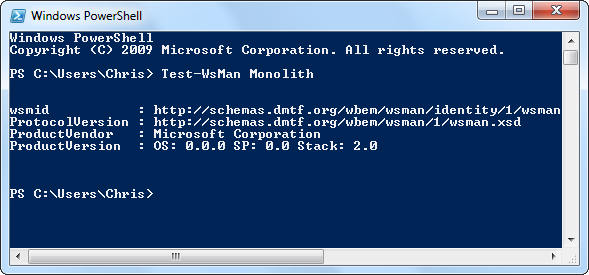


**Testing the Connection**

On the computer you want to access the remote system from, use the Test-WsMan cmdlet to test your configuration. This command tests whether the WinRM service is running on the remote computer – if it completes successfully, you’ll know that WinRM is enabled and the computers can communicate with each other. Use the following cmdlet, replacing COMPUTER with the name of your remote computer:

Test-WsMan COMPUTER

If the command completes successfully, you’ll see information about the remote computer’s WinRM service in the window. If the command fails, you’ll see an error message instead.



**Executing a Remote Command**

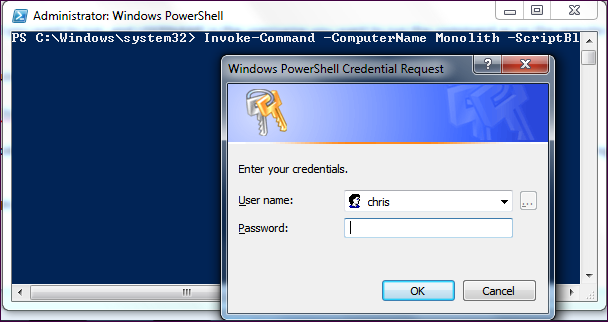
To run a command on the remote system, use the Invoke-Command cmdlet. The syntax of the command is as follows:

Invoke-Command -ComputerName COMPUTER -ScriptBlock { COMMAND } -credential USERNAME

COMPUTER represents the computer’s name, COMMAND is the command you want to run, and USERNAME is the username you want to run the command as on the remote computer. You’ll be prompted to enter a password for the username.

For example, to view the contents of the C:\ directory on a remote computer named Monolith as the user Chris, we could use the following command:

Invoke-Command -ComputerName Monolith -ScriptBlock { Get-ChildItem C:\ } -credential chris



**Starting a Remote Session**

Use the Enter-PSSession cmdlet to start a remote PowerShell session, where you can run multiple commands, instead of running a single command:

Enter-PSSession -ComputerName COMPUTER -Credential USER

