Invoke-Command

Run commands on local and remote computers.

Syntax

```
    Invoke-Command [-FilePath] string [[-ComputerName] string[]]
       [-ApplicationName string] [-AsJob] [-Authentication AuthenticationMe
          [-ConfigurationName string] [-Credential PSCredential] [-HideComp
             [-JobName string] [-Port int] [-SessionOption PSSessionOption]
                [-UseSSL] [-ArgumentList Object[]] [-InputObject psobject]

    Invoke-Command [-FilePath] string [[-Session] PSSession[]] [-AsJob]
       [-HideComputerName] [-JobName string] [-ThrottleLimit int]
          [-ArgumentList Object[]] [-InputObject psobject] [CommonParameter

    Invoke-Command [-FilePath] string [[-ConnectionURI] Uri[]] [-AllowRedir
       [-AsJob] [-Authentication AuthenticationMechanism]
          [-ConfigurationName string] [-Credential PSCredential] [-HideComp
             [-JobName string] [-SessionOption PSSessionOption] [-ThrottleL
                [-ArgumentList Object[]] [-InputObject psobject] [CommonPar

    Invoke-Command [-ScriptBlock] scriptblock
       [-ArgumentList Object[]] [-InputObject psobject] [CommonParameters]

    Invoke-Command [-ScriptBlock] scriptblock [[-ComputerName] string[]]
       [-ApplicationName string] [-AsJob] [-Authentication AuthenticationMe
          [-CertificateThumbprint string] [-ConfigurationName string]
             [-Credential PSCredential] [-HideComputerName] [-JobName strin
                [-Port int] [-SessionOption PSSessionOption] [-ThrottleLimi
                   [-UseSSL] [-ArgumentList Object[]] [-InputObject psobjec

    Invoke-Command [-ScriptBlock] scriptblock [[-Session] PSSession[]]
       [-AsJob] [-HideComputerName] [-JobName string] [-ThrottleLimit int]
          [-ArgumentList Object[]] [-InputObject psobject] [CommonParameter

    Invoke-Command [-ScriptBlock] scriptblock [[-ConnectionURI] Uri[]] [-Al
       [-AsJob] [-Authentication AuthenticationMechanism] [-CertificateThum
          [-ConfigurationName string] [-Credential PSCredential] [-HideComp
             [-JobName string] [-SessionOption PSSessionOption] [-ThrottleL
                [-ArgumentList Object[]] [-InputObject psobject] [CommonPar
```

Key

```
   -AllowRedirection
       Allow redirection of this connection to an alternate URI.

       When -ConnectionURI is used, the remote destination can return an in
       redirect to a different URI. By default, PowerShell does not redirec
       the AllowRedirection parameter changes this to allow the connection

       The number of times that the connection is redirected can be limited
       MaximumConnectionRedirectionCount property of the $PSSessionOption p
       or the MaximumConnectionRedirectionCount property of the value of -S
       The default value is 5. For more information, see New-PSSessionOptio

   -ApplicationName string
       The application name segment of the connection URI.
       Specify the application name when you are not using -ConnectionURI
```

The default value is the value of the $PSSessionApplicationName pref
the local computer. If this preference variable is not defined, the
This value is appropriate for most uses. For more info, see about_Pr

The WinRM service uses the application name to select a listener to
request. The value of this parameter should match the value of the U
a listener on the remote computer.

-ArgumentList Object[]
Set local variables in the command.
The variables in the command are replaced by these values before the
the remote computer. Enter the values in a comma-separated list.
Values are associated with variables in the order that they are list

The values in ArgumentList can be actual values, such as "1024", or
references to local variables, such as "$max".

To use local variables in a command, use the following command forma
{param(*$name1*[, *$name2*]...) *command-with-local-variables*} -Argume

The "param" keyword lists the local variables that are used in the c
The -ArgumentList parameter supplies the values of the variables, in

-AsJob
Run the command as a background job on a remote computer.
Use this parameter to run commands that take an extensive time to co

This will return an object that represents the job, and then display
You can continue to work in the session while the job completes.
To manage the job, use the Job cmdlets. To get the job results, use

Using -AsJob is similar to using Invoke-Command to run a Start-Job c
with -AsJob, the job is created on the local computer, even though t
and the results of the remote job are automatically returned to the
For more information, see help about_Jobs and help about_Remote_Jobs

-Authentication *AuthenticationMechanism*
The mechanism used to authenticate the user's credentials.
Valid values are:
Default, Basic, Credssp, Digest, Kerberos, Negotiate, and Negot
The default value is Default.
CredSSP authentication is available only in Vista, Server 2008, and

CAUTION: CredSSP authentication increases the security risk of the r
If the remote computer is compromised, the credentials that are pass
be used to control the network session.

-CertificateThumbprint *string*
The digital public key certificate (X509) of a user account that has
perform this action. Enter the certificate thumbprint of the certifi

Certificates are used in client certificate-based authentication.
They can only be mapped to local user accounts; they do not work wit

To get a certificate thumbprint, use Get-Item or GCI commands in the

-ComputerName *string*[]
The computers on which to run the command.
The default is the local computer.

When you use -ComputerName, PowerShell will create a temporary conn
used only to run the specified command and is then closed.
If you need a persistent connection, use -Session

Type the NETBIOS name, IP address, or fully-qualified domain name of
in a comma-separated list. To specify the local computer, type the c

To use an IP address in the value of -ComputerName , the command mus
Also, the computer must be configured for HTTPS transport or the IP
computer must be included in the WinRM TrustedHosts list on the loca
See also: "How to Add a Computer to the Trusted Host List" in help

Note: On Vista, and later, to include the local computer in the valu
open PowerShell with the "Run as administrator" option.

-ConfigurationName *string*
The session configuration to be used for the new PSSession.

Enter a configuration name or the fully qualified resource URI for a
If only the configuration name is specified, the following schema UR
   http://schemas.microsoft.com/powershell

The session configuration for a session is located on the remote com
If the specified session configuration does not exist on the remote

The default value is the value of the $PSSessionConfigurationName pr
the local computer. If this preference variable is not set, the defa

-ConnectionURI *Uri*[]
A Uniform Resource Identifier (URI) that defines the connection endp
The URI must be fully qualified.

The format of this string is:
     Transport://ComputerName:*Port*/*ApplicationName*

The default value is:
     http://localhost:5985/WSMAN

Valid values for the Transport segment of the URI are HTTP and HTTPS
If you do not specify a -ConnectionURI, the -UseSSL, -ComputerName,
parameters can be used to specify the individual URI values.

If the destination computer redirects the connection to a different
prevent the redirection unless the -AllowRedirection parameter is in

-Credential *PSCredential*
A user account that has permission to perform this action. The defau

Type a user name, such as "User64" or "Domain64\User64", or enter a
contains a PSCredential object, such as one generated by Get-Credent
When you type a user name, you will be prompted for a password.

-FilePath string
Run the specified local script on one or more remote computers.
Enter the path and file name of the script, or pipe a script path to
The script must reside on the local computer or in a directory that
access. Use -ArgumentList to specify the values of parameters in the

When you use this parameter, PowerShell converts the contents of the file to a script block, transmits the script block to the remote com on the remote computer.

-HideComputerName
Omit the computer name of each object from the output display. By default, the name of the computer that generated the object appea

This parameter affects only the output display. It does not change t

-InputObject *psobject*
Specifies input to the command.
Enter a variable that contains the objects or type a command or expr gets the objects.

When using -InputObject, use the $input automatic variable in the va ScriptBlock parameter to represent the input objects.

-JobName *string*
A friendly name for the background job.
By default, jobs are named "Job*n*", where *n* is an ordinal number.
This parameter is valid only with -AsJob.

If -JobName is used in a command, -AsJob is implied, even if -AsJob For more information about PowerShell background jobs, see help abou

-Port *int*
The network port on the remote computer used for this command.
The default is port 80 (the HTTP port).

Before using an alternate port, configure the WinRM listener on the to listen at that port.
To configure the listener:
remove-item -path wsman:\Localhost\listener\listener* -recurse
new-item -path wsman:\Localhost\listener -Transport http -Addres

Do not use -Port parameter unless you must. The Port set in the comm computers or sessions on which the command runs. An alternate port s the command from running on all computers.

-ScriptBlock *scriptblock*
The commands to run.
Enclose the commands in curly braces { } to create a script block.
This parameter is required.

By default, any variables in the command are evaluated on the remote To include local variables in the command, use -ArgumentList or in PowerShell 3.0+ use the prefix **$using:** before the local variab e.g. { echo $using:mylocalVar }

-Session *PSSession*[]
Run the command in the specified PowerShell sessions (PSSessions).
Enter a variable that contains the PSSessions or a command that crea the PSSessions, such as New-PSSession or Get-PSSession.

Using -Session establishes a persistent connection to the remote com Use this to run a series of related commands that share data.
To run a single command or a series of unrelated commands, use -Comp To create a PSSession, use New-PSSession.

```
-SessionOption PSSessionOption
    Set advanced options for the session.
    Enter a SessionOption object that you create by using New-PSSessionO

    The default values for the options are determined by the value of th
    $PSSessionOption preference variable, if set. Otherwise, the session

-ThrottleLimit int
    The maximum number of concurrent connections that can be established
    If you omit this parameter or enter a value of 0, the default value,

    The throttle limit applies only to the current command, not to the s

-UseSSL
    Use the Secure Sockets Layer (SSL) protocol to establish a connectio
    By default, SSL is not used.

    WS-Management encrypts all PowerShell content transmitted over the n
    an additional protection that sends the data across an HTTPS, instea

    If you use this parameter, but SSL is not available on the port used
```

Standard Aliases for Invoke-Command: `icm`

Invoke-Command runs commands on a local or remote computer and returns all output from the commands, including errors. A single Invoke-Command command, can run commands on multiple computers.

To run a single command on a remote computer, use `-ComputerName`.

To run a series of related commands that share data, create a PSSession (a persistent connection) on the remote computer, and then use `Invoke-Command -Session` to run the command in the PSSession.

When retrieving information from a remote machine PowerShell will by default return an object with a large number of properties, performance can be greatly improved by using Select-Object to return only the properties needed.

Invoke-Command may also be used on a local computer to evaluate or run a string in a script block as a command. PowerShell converts the script block to a command and runs the command immediately in the current scope, instead of just echoing the string at the command line.

Before using Invoke-Command to run commands on a remote computer, read help`about_Remote`.

Alternatives

`Invoke-Command` uses WSMAN to run commands on remote machines.
An alternative to this is using the WMI method `Win32_Process Create()` which can be run locally or remotely over RPC

Open a notepad process on a remote computer:

```
Invoke-WMIMethod -Class Win32_Process -Name Create -Computername workst
```

Examples

List any running firefox processes running on 3 remote workstations:

```
PS C:\> invoke-command -ComputerName
Workstation64,Workstation65,Workstation66 -ScriptBlock {Get-Process -Name
'firefox'}
```

Run the Test.ps1 script on the Server64 computer. The script is located on the local computer. The script runs on the remote computer and the results are returned to the local computer:

```
PS C:\> invoke-command -filepath c:\scripts\test.ps1 -computerName
Server64
```

Run a script block (containing just a Get-Culture command) on the Server64 computer. Pass user credentials with permission to run the command:

```
PS C:\> invoke-command -computername server64 -credential domain64\user64
-scriptblock {get-culture}
```

Run the same "Get-Culture" command in a session (a persistent connection):

```
PS C:\> $sess = new-pssession -computername server64 -credential
domain64\user64
PS C:\> invoke-command -session $sess -scriptblock {get-culture}
```

Save a command in a local variable, then use Invoke-Command to run the command against several remote computers:

```
PS C:\> $command = { get-eventlog -log "windows powershell" | where
{$_.message -like "*certificate*"} }
PS C:\> invoke-command -computername Server64, Server65, Server102 -
scriptblock $command
```

Get the version of the PowerShell host running on a remote computer:

```
PS C:\> invoke-command -computername server64 -scriptblock {(get-
host).version}
```

Get the version of the PowerShell host running on a list of remote computers (computers.txt):

```
PS C:\> $version = invoke-command -computername (get-content
computers.txt) -scriptblock {(get-host).version}
PS C:\> $version
```

Run the delprof2 utility on workstation64, use the call operator & to run the non-PowerShell utility:

```
PS C:\> invoke-command -computername workstation64 -scriptblock {& "c:\Temp
```

Run a background job on two remote computers. Because the Invoke-Command command uses the AsJob parameter, the commands run on the remote computers, but the job actually resides on the local computer and the results are transmitted to the local computer:

```
PS C:\> $s = new-pssession -computername Server01, Server02
PS C:\> invoke-command -session $s -scriptblock {get-eventlog system} -
AsJob
PS C:\> $j = Get-Job
PS C:\> $results = $j | Receive-Job
```

Run the Sample.ps1 script on all of the computers listed in the Servers.txt file. Using the -FilePath parameter to specify the script file has the effect that the content of the script is automatically copied into a script block and then passed to and run on each of the remote computers:

```
PS C:\> invoke-command -comp (get-content servers.txt) -filepath
c:\scripts\sample.ps1 -argumentlist Process, Service
```

"There is another side to chivalry. If it dispenses leniency, it may with equal justification invoke control" ~ Freda Adler

Related PowerShell Cmdlets:

Get-Command - Retrieve basic information about a command.
Get-StartApps - Get the names and AppIDs of installed apps.
Invoke-Expression - Run a PowerShell expression.
Invoke-History - Invoke a previously executed Cmdlet.
Invoke-Item - Invoke an executable or open a file (START)
Enter-PSSession - Start an interactive session with a remote computer.
Test-WSMan - Test whether the WinRM service is running.
Start-Process - Start one or more processes, optionally as a specific user.
--% - Stop parsing input as PowerShell commands.
. (source) - Run a command script in the current shell (persist variables and functions)
Equivalent bash command: exec - Execute a command.