PowerShell says “execution of scripts is disabled on this system.”

Ask Question

1415

356

I am trying to run a cmd file that calls a powershell script from `cmd.exe` , and I am getting the below error:

> `Management_Install.ps1` cannot be loaded because the execution of scripts is disabled on this system.

I have run

```
Set-ExecutionPolicy -ExecutionPolicy Unrestricted
```

and when I run `Get-ExecutionPolicy` from powershell , I get `Unrestricted` back.

```
PS C:\Users\Administrator\> Get-ExecutionPolicy
Unrestricted



C:\Projects\Microsoft.Practices.ESB\Source\Samples\Management Portal\Install\Scripts\>
powershell .\Management_Install.ps1 1

WARNING: Running x86 PowerShell...
```

> File `C:\Projects\Microsoft.Practices.ESB\Source\Samples\Management Portal\Install\Scripts\Management_Install.ps1` cannot be loaded because the execution of scripts is disabled on this system. Please see " `get-help about_signing` " for more details.
>
> At line:1 char:25
>
> - `.\Management_Install.ps1` <<<< 1
>   - CategoryInfo : NotSpecified: (:) [], PSSecurityException
>   - FullyQualifiedErrorId : RuntimeException

```
C:\Projects\Microsoft.Practices.ESB\Source\Samples\Management Portal\Install\Scripts\>
PAUSE

Press any key to continue . . .
```

The system is Windows Server 2008R2.

What am I doing wrong?

powershell windows-server-2008-r2

edited yesterday
TheIncorrigible1
10.4k 3 14 36

asked Oct 27 '10 at 21:39
Conor
7,084 3 11 4

3 No look mate, I am logged in as the Administrator and also did a run as "Administrator" and got the same result as above... – Conor Oct 27 '10

at 22:42

2 Got to the bottom of it, the error was misleading, the problem was in my powershell script, thanks! – Conor Oct 28 '10 at 0:27

@ChristopheD Wrong. Do not run as admin unless you know why. That's a dangerous habit and a bad way to solve a problem you don't understand – Kolob Canyon Nov 7 '17 at 17:35

1 @KolobCanyon Yeah, but it works surprisingly often. – Erhannis Jan 4 '18 at 22:24

Its worth pointing out that Execution Policy carries several scopes, and running PowerShell in different ways can get you different policies. To view the list of policies, run `Get-ExecutionPolicy -List` . – Bacon Bits Mar 9 '18 at 16:26

22 Answers

1846

✓

If you're using Windows Server 2008 R2 then there is an *x64* and *x86* version of PowerShell both of which have to have their execution policies set. Did you set the execution policy on both hosts?

As an *Administrator*, you can set the execution policy by typing this into your PowerShell window:

```
Set-ExecutionPolicy RemoteSigned
```

For more information, see *Using the Set-ExecutionPolicy Cmdlet*.

edited Feb 8 at 17:23
SWdV
533 7 21

answered Oct 28 '10 at 1:16
Chad Miller
23.5k 3 21 29

117 `Set-ExecutionPolicy Restricted` seems to be the way to undo it if you want to put the permissions back to as they were: technet.microsoft.com/en-us/library/ee176961.aspx. The temporary bypass method by `@Jack Edmonds` looks safer to me: `powershell -ExecutionPolicy ByPass -File script.ps1` – SharpC Nov 4 '14 at 10:39

15 For a more secure policy, scope it to the actual user: Set-ExecutionPolicy RemoteSigned -Scope CurrentUser – Nuno Aniceto Jul 7 '15 at 13:18

Set-ExecutionPolicy RemoteSigned cannot be the first line in your script. If it is, highlight it and run selected only INITIALLY before running the rest of your script. – ozzy432836 Jun 21 '17 at 13:36

I came across a similar question on SF site, "Powershell execution policy within SQL Server" asked Oct 10 '14. The answers there included `Get-ExecutionPolicy -List` which helped me to see the different scopes. The cmd `Get-ExecutionPolicy` does not show all the scopes. `Import-Module SQLPS` is now working with policies changed as follows: `{Undefined-Process,MachinePolicy,UserPolicy,}; {RemoteSigned-CurrentUser, LocalMachine}` . – SherlockSpreadsheets Feb 1 at 17:45

595

You can bypass this policy by adding `-ExecutionPolicy ByPass` when running PowerShell

```
powershell -ExecutionPolicy ByPass -File script.ps1
```

answered Feb 6 '12 at 21:28

Jack Edmonds
21.3k 13 54 75

3 This is also really handy if you're on a non-administrator account. I made a shortcut to `%SystemRoot%\system32\WindowsPowerShell\v1.0\powershell.exe -ExecutionPolicy ByPass` on my taskbar. – zek19 Jan 14 '15 at 12:15

Note that Microsoft Technet stylizes it as "Bypass", not "ByPass". See: technet.microsoft.com/nl-nl/library/hh849812.aspx – Jelle Geerts Mar 1 '16 at 1:57

This doesn't work for me, I get the same permission denied as if I called it normally. Calling a ps1 from a .bat by doing `type script.ps1 | powershell -` does work though. – Some_Guy Oct 19 '17 at 17:56

6 The purpose to Execution Policy is to prevent people from double-clicking a `.ps1` and accidentally running something they didn't mean to. This would happen with `.bat` files – Kolob Canyon Nov 7 '17 at 17:38

115

I had a similar issue and noted that the default `cmd` on Windows Server 2012, was running the x64 one.

For **Windows 7**, **Windows 8**, **Windows 10**, **Windows Server 2008 R2** or **Windows Server 2012**, run the following commands as **Administrator**:

x86 (32 bit)
Open `C:\Windows\SysWOW64\cmd.exe`
Run the command `powershell Set-ExecutionPolicy RemoteSigned`

x64 (64 bit)
Open `C:\Windows\system32\cmd.exe`
Run the command `powershell Set-ExecutionPolicy RemoteSigned`

You can check mode using

- In CMD: `echo %PROCESSOR_ARCHITECTURE%`
- In Powershell: `[Environment]::Is64BitProcess`

References:
MSDN - Windows PowerShell execution policies
Windows - 32bit vs 64bit directory explanation

edited 2 days ago

answered Aug 30 '13 at 13:10

Ralph Willgoss
7,523 4 52 60

103

Most of the existing answers explain the *How*, but very few explain the *Why*. And before you go around executing code from strangers on the Internet, especially code that disables security measures, you should understand exactly what you're doing. So here's a little more detail on this problem.

From the TechNet About Execution Policies Page:

> Windows PowerShell execution policies let you determine the conditions under which Windows PowerShell loads configuration files and runs scripts.

The benefits of which, as enumerated by PowerShell Basics - Execution Policy and Code Signing, are:

> - **Control of Execution** - Control the level of trust for executing scripts.
> - **Command Highjack** - Prevent injection of commands in my path.

> - **Identity** - Is the script created and signed by a developer I trust and/or a signed with a certificate from a Certificate Authority I trust.
> - **Integrity** - Scripts cannot be modified by malware or malicious user.

To check your current execution policy, you can run `Get-ExecutionPolicy` . But you're probably here because you want to change it.

To do so you'll run the `Set-ExecutionPolicy` cmdlet.

You'll have two major decisions to make when updating the execution policy.

Execution Policy Type:

- `Restricted` † - No Script either local, remote or downloaded can be executed on the system.
- `AllSigned` - All script that are ran require to be digitally signed.
- `RemoteSigned` - All remote scripts (UNC) or downloaded need to be signed.
- `Unrestricted` - No signature for any type of script is required.

Scope of new Change

- `LocalMachine` † - The execution policy affects all users of the computer.
- `CurrentUser` - The execution policy affects only the current user.
- `Process` - The execution policy affects only the current Windows PowerShell process.

† = Default

For example: if you wanted to change the policy to RemoteSigned for just the CurrentUser, you'd run the following command:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

Note: In order to change the Execution policy, you must be running **PowerShell As Adminstrator**. If you are in regular mode and try to change the execution policy, you'll get the following error:

> Access to the registry key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell' is denied. To change the execution policy for the default (LocalMachine) scope, start Windows PowerShell with the "Run as administrator" option.

If you want to tighten up the internal restrictions on your own scripts that have not been downloaded from the Internet (or at least don't contain the UNC metadata), you can force the policy to only run signed sripts. To sign your own scripts, you can follow the instructions on Scott Hanselman's article on Signing PowerShell Scripts.

Note: Most people are likely to get this error whenever they open Powershell because the first thing PS tries to do when it launches is execute your user profile script that sets up your environment however you like it.

The file is typically located in:

```
%UserProfile%\My Documents\WindowsPowerShell\Microsoft.PowerShellISE
```

You can find the exact location by running the powershell variable

```
$profile
```

If there's nothing that you care about in the profile, and don't want to fuss with your security settings, you can just delete it and powershell won't find anything that it cannot execute.

edited Apr 23 '15 at 13:25

answered Nov 16 '14 at 8:05

KyleMit
59.2k 36 248 408

7 I feel like it's important to note that while execution policy is a security measure, it's not one that's intended to prevent users from executing PowerShell code. Execution policy is something that's intended to *protect your scripts* and determine who wrote, modifed or approved them and that's all. It's trivial to get around execution policy with something as simple as `Get-Content .\MyFile.ps1 | powershell.exe -NoProfile - .` – Bacon Bits Mar 29 '16 at 12:52

1 Given the existence of `-ExecutionPolicy ByPass` though, what is the purpose of this policy anyway? Is it just to prevent users from accidentally opening a powershell console and running a malicious script? Couldn't the attacker just use an executable or a batch script if they wanted to get around this? Even after reading @BaconBits comment I'm not quite sure what scenario this policy is meant to prevent... – Ajedi32 Jul 19 '16 at 16:29

2 @Ajedi32 Say I have a task that runs a script on a network share. When I invoke my script, I want my process to verify that the script is signed. I want my code to double check that the script I'm going to run is the code I trust to run. I don't care if you can run my code. Stopping that is access rights' job. I just want to prevent you from making me run code that I didn't write. Access rights means the operating system prevents you from modifying my code when you're logged on. Code signing and execution policy means my script hasn't been modified when *I* go to run it. – Bacon Bits Jul 19 '16 at 21:46

35

In Windows 7:

Go to Start Menu and search for "Windows PowerShell ISE".

Right click the x86 version and choose "Run as administrator".

In the top part, paste `Set-ExecutionPolicy RemoteSigned` ; run the script. Choose "Yes".

Repeat these steps for the 64-bit version of Powershell ISE too (the non x86 version).

I'm just clarifying the steps that @Chad Miller hinted at. Thanks Chad!

answered Dec 4 '12 at 5:25
Ryan
9,010 8 66 163

34

Also running this command before the script also solves the issue:

```
set-executionpolicy unrestricted
```

edited Mar 9 '16 at 18:16
Peter Mortensen
13.8k 19 87 113

answered Mar 27 '12 at 6:11
manik sikka
405 4 2

24 -1 - Follow the principle of least permission. At least set the policy to `RemoteSigned` before removing all restrictions on your security policy. If that doesn't work, then re-assess what your pain points are and why it isn't working. You can set `unrestricted` as a last resort, but it shouldn't be your starting point. – KyleMit Nov 16 '14 at 8:08

3 Thanks for pointing out this option too. With all due respect to security needs for production purposes, in the times when quick prototyping ability demand is so high, all the policies and security really get in the way of getting stuff done. – tishma Apr 2 '16 at 12:27

1 Regarding the comment re prototyping, I'm afraid that this is why crappy code gets into production. Of course this is just a trivial example, but if you can't solve something this trivial during development, it's a worry for release. Also, for bespoke code, and if you can, know the target environment - we set the majority of our internal systems as `Remotesigned` . – Trix Feb 12 '17 at 13:43

29

If you are in an environment where you are not an administrator, you can set the Execution Policy just for you, and it will not require administrator.

```
Set-ExecutionPolicy -Scope "CurrentUser" -ExecutionPolicy "RemoteSig
```

or

```
Set-ExecutionPolicy -Scope "CurrentUser" -ExecutionPolicy "Unrestric
```

You can read all about it in the help entry.

```
Help Get-ExecutionPolicy -Full
Help Set-ExecutionPolicy -Full
```

edited Dec 1 '14 at 19:49
Peter Mortensen
13.8k 19 87 113

answered Nov 19 '13 at 19:13
Micah 'Powershell Ninja'
339 3 4

Worked great for me in Windows 8, even when `Set-ExecutionPolicy Unrestricted` as an admin didn't seem to "unrestrict" enough to actually help. – patridge Aug 21 '14 at 15:32

1 I believe what you may be experiencing is a GPO or something else overwriting your setting of the "LocalMachine" level of ExecutionPolicy. You cannot overwrite what a Domain Policy has in place with the Set-ExecutionPolicy command. However, but setting the "CurrentUser" level of access, you and only you will have the specified Execution Policy. This is because the computer looks at the CurrentUser for execution policy before it looks at the LocalMachine setting. – Micah 'Powershell Ninja' Sep 25 '14 at 14:57

1 Set-ExecutionPolicy -Scope "CurrentUser" -ExecutionPolicy "Unrestricted" is the only solution that worked for me. Thank you – Paulj Aug 23 '16 at 19:43

23

RemoteSigned: all scripts you created yourself will be run, and all scripts downloaded from the Internet will need to be signed by a trusted publisher.

OK, change the policy by simply typing:

```
Set-ExecutionPolicy RemoteSigned
```

edited Mar 12 '16 at 18:59
Peter Mortensen
13.8k 19 87 113

answered Jul 20 '11 at 12:37
Jaime
239 2 2

We can get the status of current `ExecutionPolicy` by the command below:

20

```
Get-ExecutionPolicy;
```

By default it is **Restricted**. To allow the execution of PowerShell scripts we need to set this ExecutionPolicy either as **Bypass** or **Unrestricted**.

We can set the policy for Current User as `Bypass` or `Unrestricted` by using any of the below PowerShell commands:

```
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Bypass -Forc

Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Unrestricted
```

Unrestricted policy loads all configuration files and runs all scripts. If you run an unsigned script that was downloaded from the Internet, you are prompted for permission before it runs.

Whereas in **Bypass** policy, nothing is blocked and there are no warnings or prompts during script execution. Bypass `ExecutionPolicy` is more relaxed than `Unrestricted`.

edited Jun 26 '18 at 19:56
Peter Mortensen
13.8k 19 87 113

answered Sep 7 '16 at 7:00
Pratik Patil
2,055 14 24

2 `Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Bypass -Force;` AKA quick and dirty way to tell VS2015 to stop complaining and run my bloody script. thanks. lifesaver. – Eon Sep 8 '16 at 13:48

1 The trailing semicolons on the ends of your commands are superfluous. – Bill_Stewart Nov 27 '18 at 21:57

18

I'm using **Windows 10** and was unable to run any command. The only command that gave me some clues was this:

[x64]

> 1. Open C:\Windows\SysWOW64\cmd.exe *[as administrator]*
> 2. Run the command> **powershell Set-ExecutionPolicy Unrestricted**

But this didn't work. It was limited. Probably new security policies for Windows10. I had this error:

> Set-ExecutionPolicy: Windows PowerShell updated your execution policy successfully, but the setting is overridden by a policy defined at a more specific scope. Due to the override, your shell will retain its current effective execution policy of...

So I found another way (**solution**):

1. Open Run Command/Console (Win + R)
2. Type: **gpedit.msc** (Group Policy Editor)
3. Browse to *Local Computer Policy* -> *Computer Configuration* -> *Administrative Templates* -> *Windows Components* -> *Windows Powershell*.
4. Enable "**Turn on Script Execution**"
5. Set the policy as needed. I set mine to "**Allow all scripts**".

Now open PowerShell and enjoy ;)

edited Mar 9 '16 at 18:26
Peter Mortensen
13.8k 19 87 113

answered Sep 1 '15 at 9:32

hugocabral
616 6 7

Is this a stand-alone installation, or are you connected to a workgroup or domain? – MEMark Oct 2 '15 at 15:09

Why open cmd to do it? Just open ISE (as admin) and type `Set-ExecutionPolicy RemoteSigned` – Kolob Canyon Oct 27 '16 at 16:49

OMG this finally fixed my win10 box, @kolob set-executionpolicy is not enough – xer0x Jan 14 at 6:48

You should not be using `Unrestricted`. It is better practice to use `RemoteSigned` – Kolob Canyon Jan 14 at 18:37

@xer0x it should be as long as you run powershell as an administrator – Kolob Canyon Jan 14 at 18:38

9

Setting the execution policy is environment-specific. If you are trying to execute a script from the running x86 ISE you have to use the x86 PowerShell to set the execution policy. Likewise, if you are running the 64-bit ISE you have to set the policy with the 64-bit PowerShell.

edited Mar 9 '16 at 18:17
Peter Mortensen
13.8k 19 87 113

answered Aug 25 '12 at 0:33
ScriptAholic
99 1 1

7

You can also bypass this by using the following command:

```
PS > powershell Get-Content .\test.ps1 | Invoke-Expression
```

You can also read this article by Scott Sutherland that explains 15 different ways to bypass the PowerShell `Set-ExecutionPolicy` if you don't have administrator privileges:

15 Ways to Bypass the PowerShell Execution Policy

edited Jun 26 '18 at 19:54
Peter Mortensen
13.8k 19 87 113

answered Dec 30 '16 at 14:45
Alexander Sinno
344 4 16

6

Win + R and type copy paste command and press OK:

```
powershell Set-ExecutionPolicy -Scope "CurrentUser" -ExecutionPolicy
```

And execute your script.

Then revert changes like:

```
powershell Set-ExecutionPolicy -Scope "CurrentUser" -ExecutionPolicy
```

edited Jun 19 '18 at 21:09
Peter Mortensen
13.8k 19 87 113

answered Oct 3 '17 at 6:29
Qamar
2,466 14 34

5

1. Open PowerShell as Administrator and run **Set-ExecutionPolicy -Scope CurrentUser**
2. Provide **RemoteSigned** and press Enter
3. Run **Set-ExecutionPolicy -Scope CurrentUser**
4. Provide **Unrestricted** and press Enter

answered Apr 19 '18 at 5:11
Ramanujam Allam
394 4 10

2

In the PowerShell ISE editor I found running the following line first allowed scripts.

```
Set-ExecutionPolicy RemoteSigned -Scope Process
```

edited Mar 9 '16 at 18:18
Peter Mortensen
13.8k 19 87 113

answered May 29 '15 at 14:50
David Douglas
8,652 2 45 48

1

In PowerShell 2.0, the execution policy was set to disabled by default.

From then on, the PowerShell team has made a lot of improvements, and they are confident that users will not break things much while running scripts. So from PowerShell 4.0 onward, it is enabled by default.

In your case, type `Set-ExecutionPolicy RemoteSigned` from the PowerShell console and say yes.

edited Mar 12 '16 at 19:05
Peter Mortensen
13.8k 19 87 113

answered Oct 28 '15 at 19:12
Adil Arif
11 3

1

Go to the registry path `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell` and set `ExecutionPolicy` to `RemoteSigned`.

edited Jun 26 '18 at 19:55
Peter Mortensen
13.8k 19 87 113

answered Oct 8 '16 at 19:20
sunish surendran.k
41 2 12

1. Open PowerShell as Administrator and run **Set-ExecutionPolicy -Scope CurrentUser** 2. Provide **RemoteSigned** and press Enter 3. Run **Set-ExecutionPolicy -Scope CurrentUser** 4. Provide **Unrestricted** and press Enter – Ramanujam Allam Apr 19 '18 at 5:12

0

I found this line worked best for one of my Windows Server 2008 R2 servers. A couple of others had no issues without this line in my PowerShell scripts:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Force -Scope Proc
```

edited Mar 9 '16 at 18:22
Peter Mortensen
13.8k 19 87 113

answered Jul 2 '15 at 13:03
WIlliamT
11

0

Open the PowerShell window as an **Administrator**. It will work.

edited Jun 26 '18 at 19:52
Peter Mortensen
13.8k 19 87 113

answered May 12 '17 at 11:21
Suganthan Raj
643 2 11 24

It did not works. However, running powershell Set-ExecutionPolicy RemoteSigned did works. – Thronk Nov 16 '17 at 15:51

0

If you're here because of running it with Ruby or Chef and using `` system execution, execute as follows:

```
`powershell.exe -ExecutionPolicy Unrestricted -command
[Environment]::GetFolderPath(\'mydocuments\')`
```

That command is for getting "MyDocuments" Folder.

`-ExecutionPolicy Unrestricted` does the trick.

I hope it's helpful for someone else.

edited Dec 1 '14 at 19:50
Peter Mortensen
13.8k 19 87 113

answered Oct 20 '14 at 0:09
JGutierrezC
2,833 1 19 39

Should the enclosing `` really be there? – Peter Mortensen Dec 1 '14 at 19:52

0

Several answers point to execution policy. However some things require "runas administrator" also. This is safest in that there is no permanent change to execution policy, and can get past administrator restriction. Use with schedtask to start a batch with:

```
    runas.exe /savecred /user:administrator powershell -ExecutionPol
script.ps1
```

from both Jack Edmonds above, and Peter Mortensen / Dhana of post [How to run an application as "run as administrator" from the command prompt?]()

edited May 23 '17 at 11:55
Community ♦
1 1

answered Jan 16 '16 at 1:40
Kirt Carson
1 1

0

I had the same problem today. 64-bit execution policy was unrestricted, while 32-bit was restricted.

Here's how to change just the 32-bit policy remotely:

```
Invoke-Command -ComputerName $servername -ConfigurationName Microsof
scriptblock {Set-ExecutionPolicy unrestricted}
```

edited Jun 26 '18 at 19:44
Peter Mortensen
13.8k 19 87 113

answered Nov 10 '17 at 13:49
rko281
44 5

protected by Josh Crozier Mar 2 '17 at 18:42

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?