Auto-Start Virtual Machine Part 1

In this tutorial I will be creating a PowerShell script to automatically start an existing VirtualBox virtual machine. Refer to the prerequisites listed below to complete this tutorial.

Prerequisites

- VirtualBox VM

For instructions on how to install VirtualBox and extension pack, see my **VirtualBox Install** tutorial **here**.

If you do not already have a virtual machine, my other tutorial, **CentOS 7 Server Install**, can be accessed **here**.

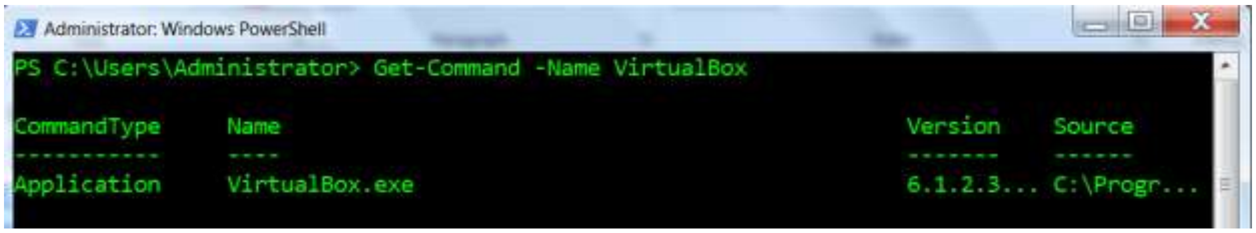
Steps to complete tutorial:

- Create Script
- Review Script
- Execute Script

Create Script

I have created an empty script file named **`auto_start_vm.ps1`**.
First, we will determine the path to the VirtualBox executable using the **`Get-Command`** cmdlet.

```
Get-Command -Name VirtualBox
```



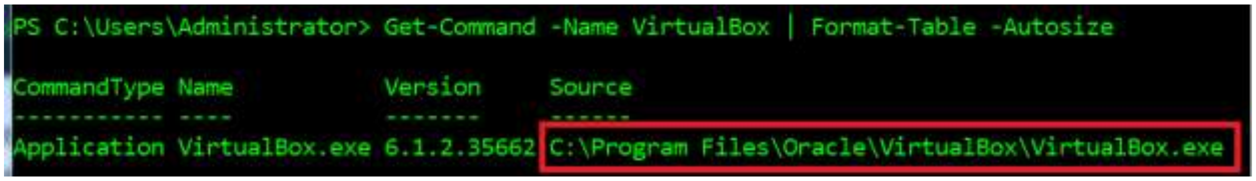
Administrator: Windows PowerShell

```
PS C:\Users\Administrator> Get-Command -Name VirtualBox

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Application     VirtualBox.exe                                     6.1.2.3... C:\Progr...
```

You will notice that the field value that we need (Source) is truncated. We will use the following to improve the readability of the tabular results.

```
Get-Command -Name VirtualBox | Format-Table -Autosize
```



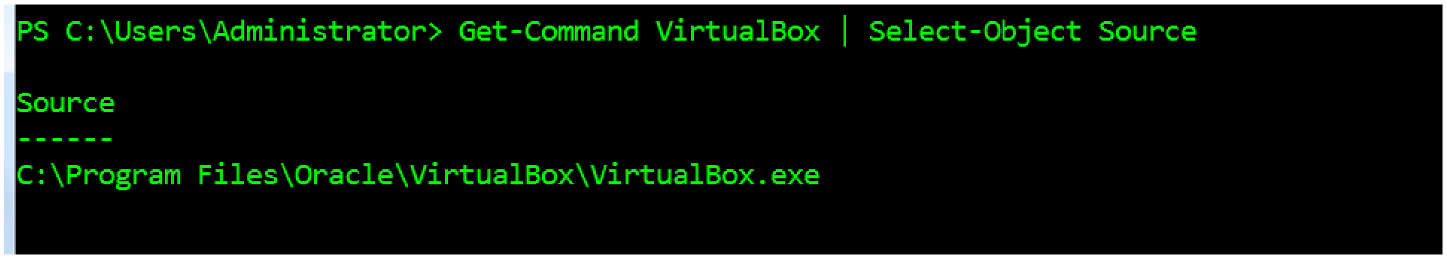
```
PS C:\Users\Administrator> Get-Command -Name VirtualBox | Format-Table -Autosize

CommandType Name           Version      Source
----------- ----           -------      ------
Application VirtualBox.exe 6.1.2.35662 C:\Program Files\Oracle\VirtualBox\VirtualBox.exe
```

Using the **`Autosize`** parameter, when running **`Format-Table`**, column widths are calculated based on size of actual data.

The following will return an object with one property, **Source**.

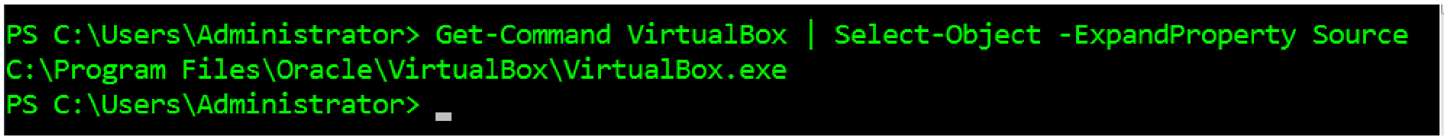
```
Get-Command VirtualBox | Select-Object Source
```



```
PS C:\Users\Administrator> Get-Command VirtualBox | Select-Object Source

Source
------
C:\Program Files\Oracle\VirtualBox\VirtualBox.exe
```

Before we assign this value to a variable, we will need to extract the **Source** property value from the object being passed down the pipeline. The following will provide us with the **Source** property value and remove any empty lines from the command results:
`Get-Command VirtualBox | Select-Object -ExpandProperty Source`

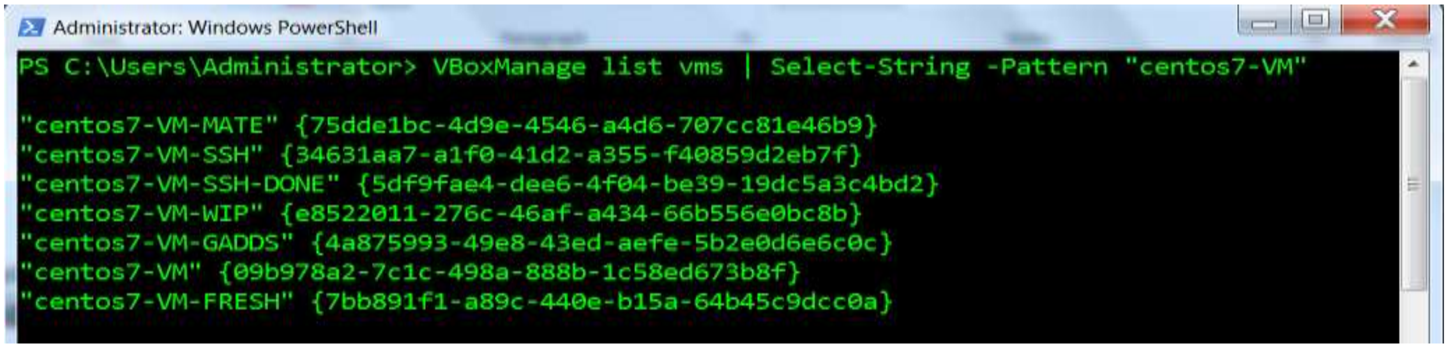


```
PS C:\Users\Administrator> Get-Command VirtualBox | Select-Object -ExpandProperty Source
C:\Program Files\Oracle\VirtualBox\VirtualBox.exe
PS C:\Users\Administrator> _
```

Now we can assign the command result to a variable for later use.

`$vbox = Get-Command VirtualBox | Select-Object -ExpandProperty Source`

Next, we will locate the virtual machine we wish to auto-start using the following:
(NOTE: use the VM name you wish to auto-start)
`VBoxManage list vms | Select-String -Pattern "centos7-VM"`

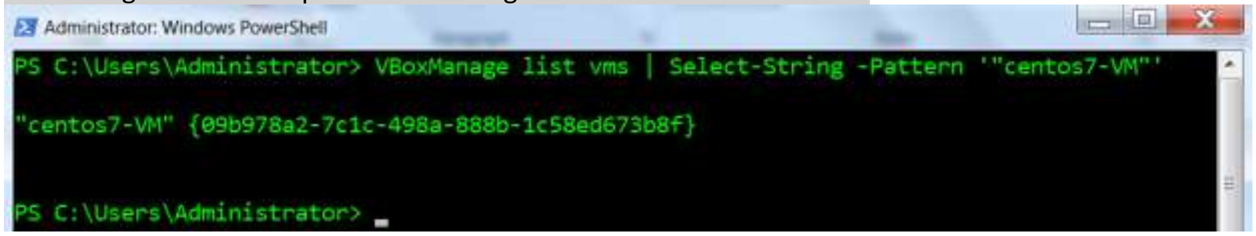


```
PS C:\Users\Administrator> VBoxManage list vms | Select-String -Pattern "centos7-VM"

"centos7-VM-MATE" {75dde1bc-4d9e-4546-a4d6-707cc81e46b9}
"centos7-VM-SSH" {34631aa7-a1f0-41d2-a355-f40859d2eb7f}
"centos7-VM-SSH-DONE" {5df9fae4-dee6-4f04-be39-19dc5a3c4bd2}
"centos7-VM-WIP" {e8522011-276c-46af-a434-66b556e0bc8b}
"centos7-VM-GADDS" {4a875993-49e8-43ed-aefe-5b2e0d6e6c0c}
"centos7-VM" {09b978a2-7c1c-498a-888b-1c58ed673b8f}
"centos7-VM-FRESH" {7bb891f1-a89c-440e-b15a-64b45c9dcc0a}
```

You will notice that I have many VMs whose names begin with "**centos7-VM**". PowerShell's **`Select-String`** cmdlet returned all of them, as demonstrated above.

I will use single quotes outside of the double quotes (**'"centos7-VM"'**) to ensure that the double-quotes are included in the search.
`VBoxManage list vms | Select-String -Pattern '"centos7-VM"'`

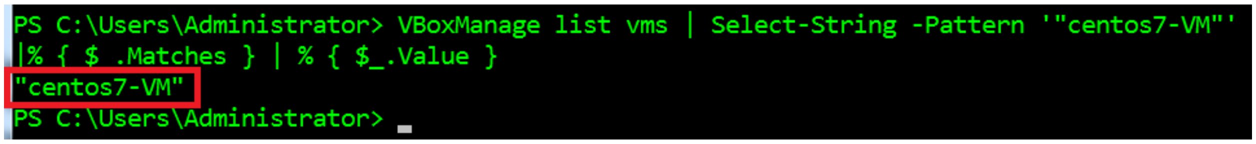


```
PS C:\Users\Administrator> VBoxManage list vms | Select-String -Pattern '"centos7-VM"'

"centos7-VM" {09b978a2-7c1c-498a-888b-1c58ed673b8f}


PS C:\Users\Administrator> _
```

Before we assign this value to a variable, we will need to remove any empty lines from the command results. To do this we will add the following to the command:
`VBoxManage list vms | Select-String -Pattern '"centos7-VM"' |% { $_.Matches } | % { $_.Value }`



```
PS C:\Users\Administrator> VBoxManage list vms | Select-String -Pattern '"centos7-VM"'
|% { $_.Matches } | % { $_.Value }
"centos7-VM"
PS C:\Users\Administrator> _
```

For each object returned by **`Select-String`**, we are finding the value of each match. Although this tutorial is not a deep-dive into PowerShell cmdlets, the default output of **`Select-String`** is a **`MatchInfo`** object, which includes detailed information (properties) about the matches. The **`MatchInfo`** object has a property called **`Matches`**, which contains a list of regular expression matches. For each of the matches in the list returned, we are using the **`Value`** property to get the actual value.

Now we can assign the command result to a variable for later use.
`$vm = VBoxManage list vms | Select-String -Pattern '"centos7-VM"' | % { $_.Matches } | % { $_.Value }`

We now have both the path to the VirtualBox executable, as well as, the VM name assigned to variables.

Although we do not need to first start VirtualBox before we can start a virtual machine, I like having the management interface at my disposal. You can include the following in your script, or not, the choice is yours.

```
# start VirtualBox for access to management interface
Start-Process $vbox
```

To ensure that VirtualBox has successfully started, we will perform the following test:

```
# ensure VirtualBox has successfully started
if (Get-Process VirtualBox -ErrorAction SilentlyContinue) {
      Write-Host "VirtualBox has successfully started."
} else {
      Write-Host "Something went wrong during VirtualBox startup."
}
```

You will notice that I have included **`-ErrorAction SilentlyContinue`** in our test. This is what's known as a "Common Parameter" of the **`Get-Process`** cmdlet. If VirtualBox did not start, instead of generating error messages to the PowerShell console, the command will continue execution and jump to the **`else`** clause of the **`if/else`** statement.

Now we are ready to start the virtual machine, whose name we assigned to the variable **`$vm`**.

```
# NOTE: works without VirtualBox being started first
VBoxManage startvm $vm
```

To verify that the previous command executed successfully, we perform another test.

```
# verify success of last command executed
if ( "$?" -eq "True") {
      Write-Host "$vm is booting up."
} else {
      Write-Host "Problem starting $vm"
}
```

This time we are using the PowerShell automatic variable **`$?`**. This variable contains the status of the last command executed: **True** if it succeeded and **False** if it failed.

We are ready to review the contents of the entire script.

Review Script

```
Write-Host "Automate VirtualBox VM startup."

# retrieve path to VirtualBox executable
$vbox = Get-Command VirtualBox | Select Source -ExpandProperty Source

# locate VM based on it's name
# remove any empty lines from result of command
$vm = VBoxManage list vms | Select-String -Pattern '"centos7-VM"' | % { $_.Matches } | % { $_.Value }

# start VirtualBox for access to management interface
Start-Process $vbox

# ensure VirtualBox has started
if (Get-Process VirtualBox -ErrorAction SilentlyContinue) {
      Write-Host "VirtualBox has successfully started."
} else {
      Write-Host "Something went wrong during VirtualBox startup."
}

# NOTE: works without VirtualBox being started first
VBoxManage startvm $vm
```

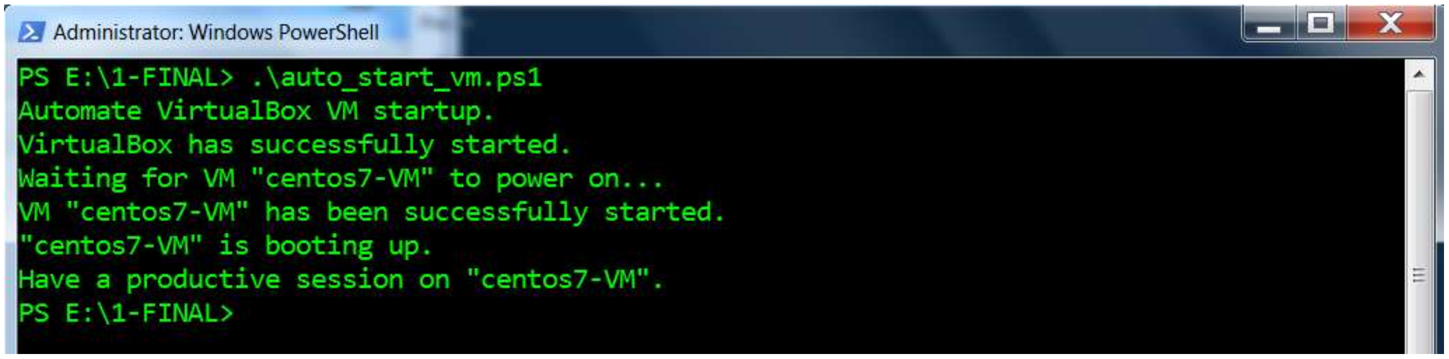
```
# verify success of last command executed
if ( "$?" -eq "True") {
        Write-Host "$vm is booting up."
} else {
        Write-Host "Problem starting $vm"
}

Write-Host "Have a productive session on $vm."

sleep 10
```

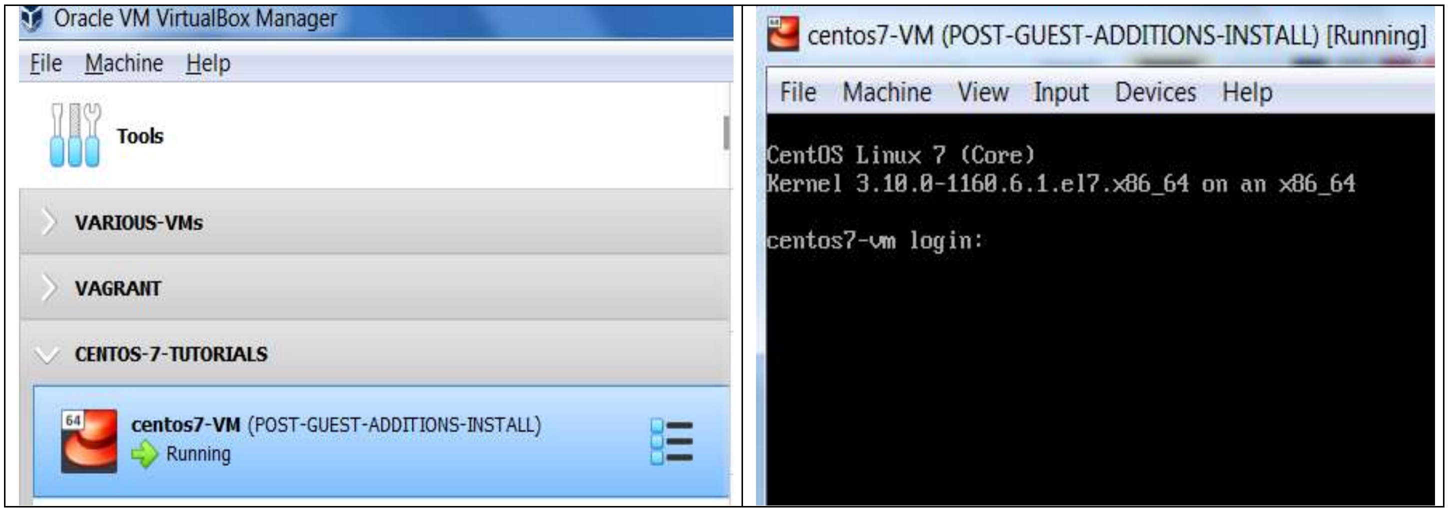
Execute Script

To execute the script, open a PowerShell console, if need be, change directory to the location of your script and execute the following: `.\auto_start_vm.ps1`



```
PS E:\1-FINAL> .\auto_start_vm.ps1
Automate VirtualBox VM startup.
VirtualBox has successfully started.
Waiting for VM "centos7-VM" to power on...
VM "centos7-VM" has been successfully started.
"centos7-VM" is booting up.
Have a productive session on "centos7-VM".
PS E:\1-FINAL>
```

VirtualBox successfully started and so did the virtual machine.


Oracle VM VirtualBox Manager
File Machine Help
Tools
VARIOUS-VMs
VAGRANT
CENTOS-7-TUTORIALS
centos7-VM (POST-GUEST-ADDITIONS-INSTALL)
Running
centos7-VM (POST-GUEST-ADDITIONS-INSTALL) [Running]
File Machine View Input Devices Help

```
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.6.1.el7.x86_64 on an x86_64

centos7-vm login:
```

Hopefully, you've enjoyed completing this tutorial and found it helpful.

Now that you can auto-start your VM, you might want to see my other tutorial, **Auto-Stop Virtual Machine Basic**, accessible here.

If you would like to see my other tutorials, they can be accessed here.

Back to top