Enabling multi-server management in Windows Server 2012

Enable remote management

Before you can take advantage of multi-server management, you will have to enable remote management on the individual servers. To do so, open the Server Manager and select the **Local Server** container. Next, click on the **Disabled** link next to Remote Management. Doing so will cause Windows to display a dialog box that gives you the option of enabling remote management from other computers. Simply select the Enable Remote Management check box (figure 1) and click OK.

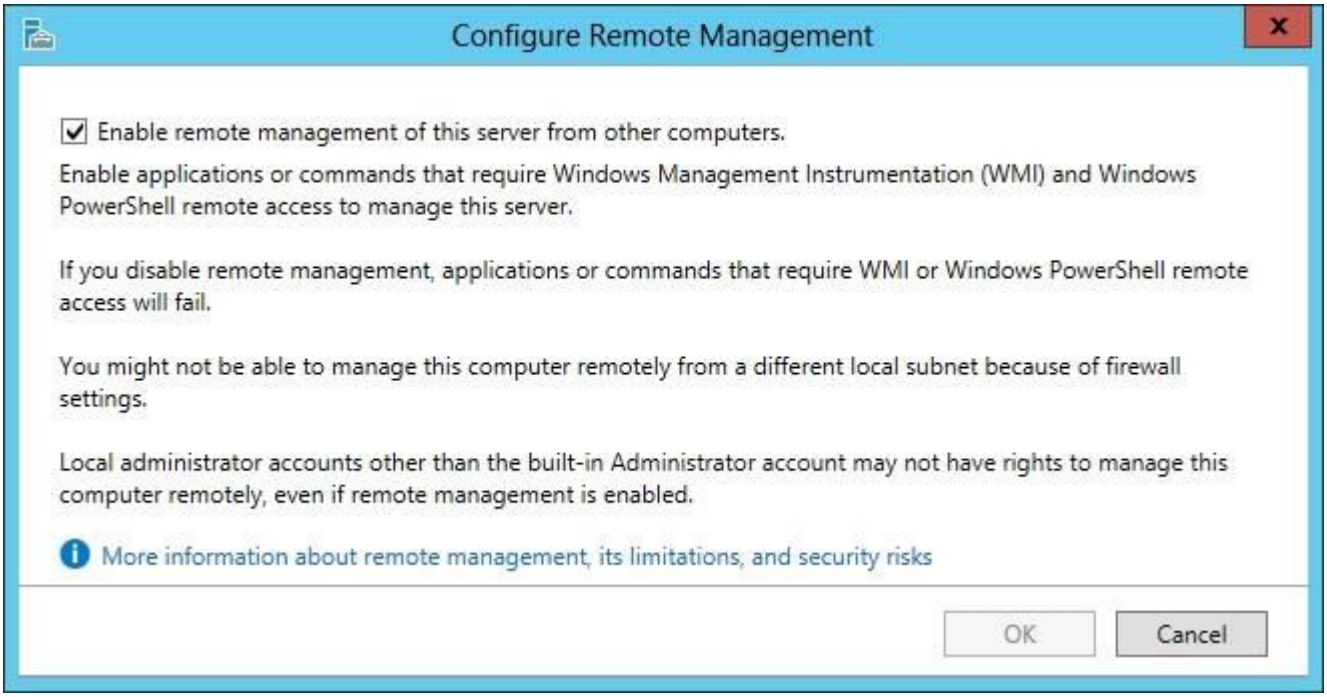

Figure 1. To enable remote management on individual servers, open Server Manager, select Local Server container and click on Disabled next to Remote Management.

Using Server Manager

The most effective way to manage multiple servers through Server Manager is to create a server group. A server group is a collection of physical or virtual servers that perform the same tasks and should be managed or monitored together. To create a server group, open Server Manager and click on the **Dashboard** option, followed by the **Create a Server Group** option shown (figure 2).

Pro+

Features

Enjoy the benefits of Pro+ membership, learn more and join.

- **E-Handbook**

  Migration patterns: Ensuring successful Windows Server transitions

- **E-Chapter**

  The complex powers of network virtualization

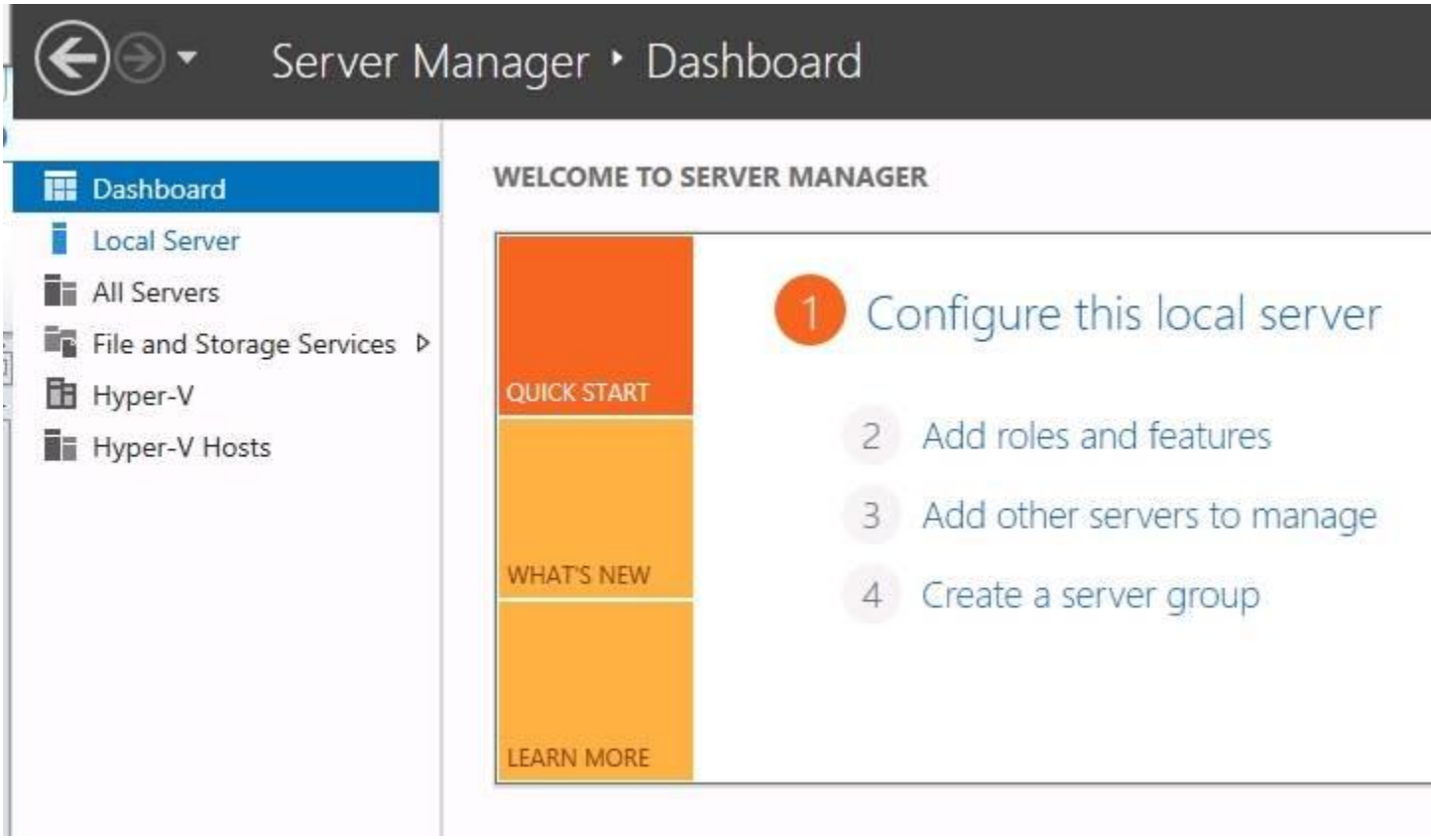

Figure 2. To create a server group, open Server Manager, click on the Dashboard option to come to the Create a Server Group option.

At this point, Windows will display the Create Server Group dialog box. Enter a name for the server group you are creating and then specify the servers that should be included in the group. For example, I created a server group called Hyper-V hosts (figure 3). Click OK when you finish selecting the servers that should be included in the group.

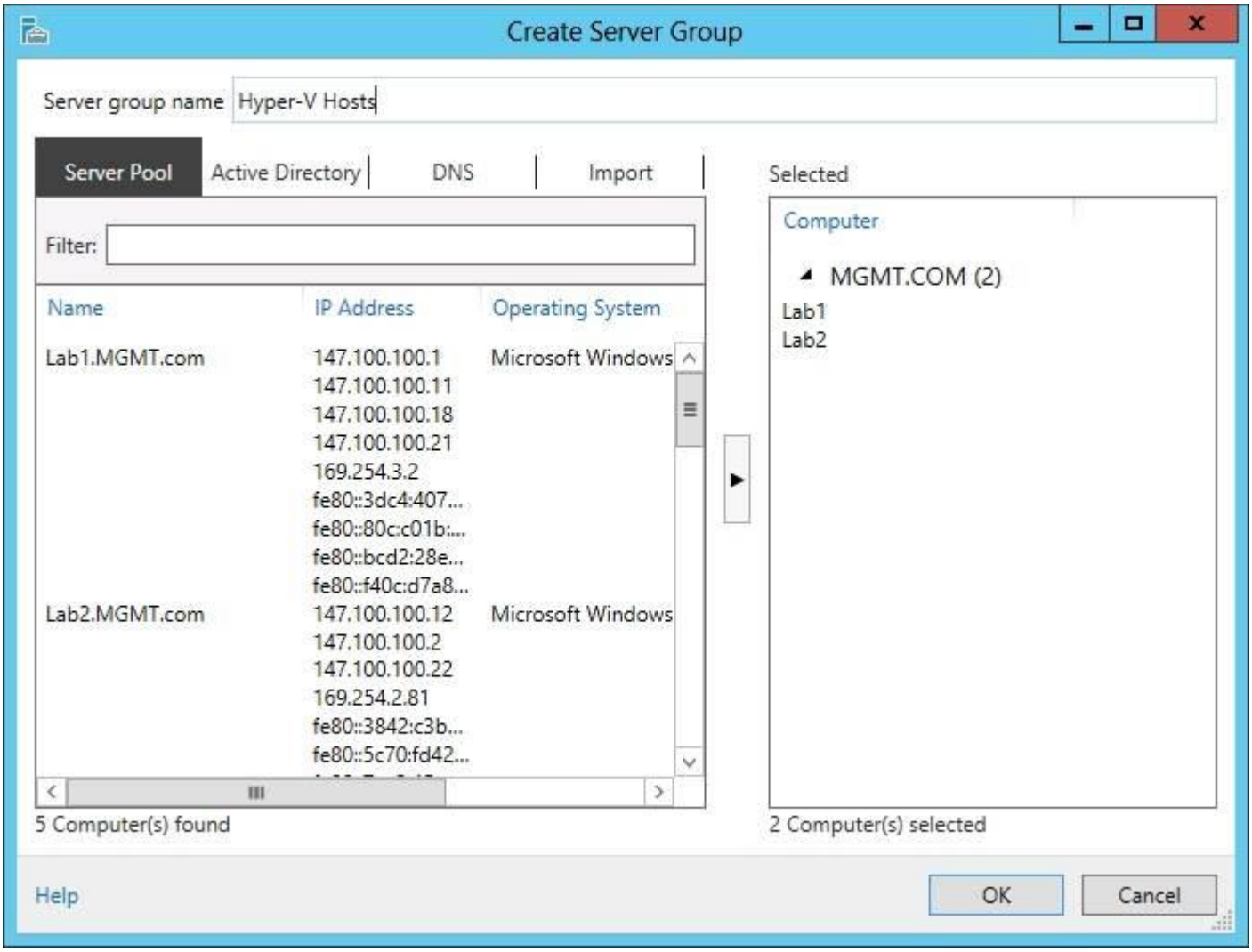

Figure 3. Enter a name for the server group and specify the servers to be included in the group.

When you finish creating the group, it will be listed in the Server Manager console. When you click on the group, Server Manager will provide you with an aggregate view of the servers that make up the group (figure 4). For example, you can see the event logs, services, performance and Best Practices Analyzer results for all of the servers in the group. You can even configure performance alerts across the various servers that make up the group.

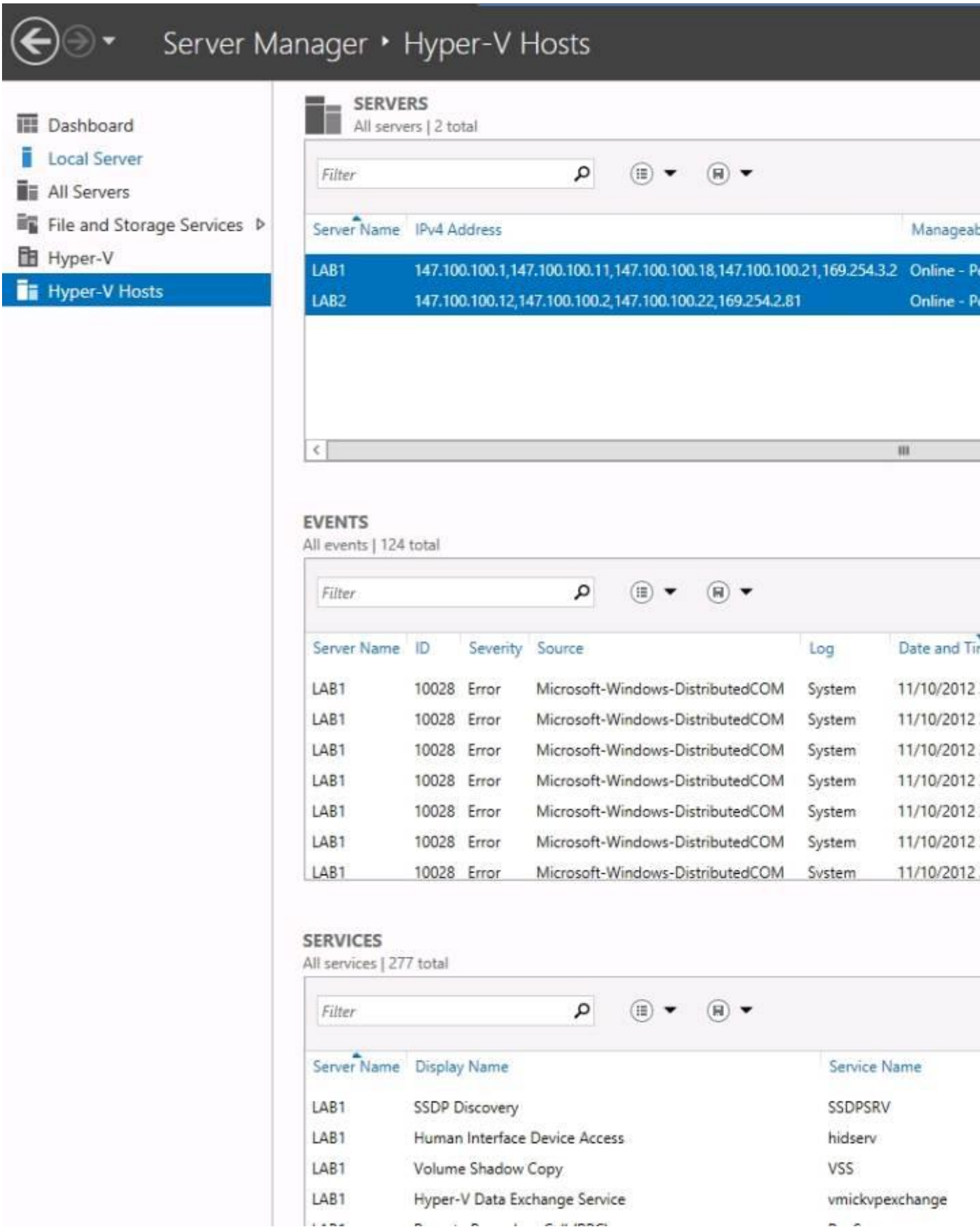
Server Manager ▸ Hyper-V Hosts
Dashboard
Local Server
All Servers
File and Storage Services
Hyper-V
Hyper-V Hosts
SERVERS
All servers | 2 total
Filter
Server Name
IPv4 Address
Manageab
LAB1
147.100.100.1,147.100.100.11,147.100.100.18,147.100.100.21,169.254.3.2
Online - P
LAB2
147.100.100.12,147.100.100.2,147.100.100.22,169.254.2.81
Online - P
EVENTS
All events | 124 total
Filter
Server Name
ID
Severity
Source
Log
Date and Ti
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
LAB1 10028 Error Microsoft-Windows-DistributedCOM System 11/10/2012
SERVICES
All services | 277 total
Filter
Server Name
Display Name
Service Name
LAB1 SSDP Discovery SSDPSRV
LAB1 Human Interface Device Access hidserv
LAB1 Volume Shadow Copy VSS
LAB1 Hyper-V Data Exchange Service vmickvpexchange

Figure 4. If you click on the group, Server Manager will give admins a view of servers in the group.

What about PowerShell?

Microsoft has long stated that Windows PowerShell is the preferred mechanism for managing Windows Server 2012. It should come as no surprise that most of the multi-server management capabilities are only exposed through PowerShell.

There are a number of different techniques that can be used to simultaneously manage multiple computers. The easiest of these techniques involves using the *Invoke-Command* cmdlet. There are three parts to the *Invoke-Command* cmdlet. First, there is the *Invoke-Command* cmdlet itself. Next, you must provide the *ComputerName* switch, followed by a list of the computers on which you wish to run the command. The last part of the command is the actual command that you want to run against the remote machines.

To see how this technique is useful, imagine that you had a number of Hyper-V servers and you wanted to see the names of the virtual machines residing on each Hyper-V host. Normally, the command you would use to create a list of virtual machines and the host server each is running on is:

```
Get-VM | FT VMName, ComputerName
```

```
PS C:\Users\Administrator.MGMT> Invoke-Command -ComputerName Lab1, Lab2, Lab3 {Get-VM | FT VMName, Comp

VMName                                                  ComputerName
------                                                  ------------
Core                                                    LAB1
Lab15-DC                                                LAB1
Lab15-Exchange                                          LAB1
Lab15-SPT                                               LAB1
Lab15-W8                                                LAB1
Lab-DC                                                  LAB1
Lab-E2K10                                               LAB1
Lab-SharePoint                                          LAB1
Lab-VM1                                                 LAB1
Lab-W7B                                                 LAB1
NewVM1                                                  LAB1
NewVM2                                                  LAB1
NewVM3                                                  LAB1
NewVM4                                                  LAB1
NewVM5                                                  LAB1
PowerShellVM                                            LAB1
Solarwinds                                              LAB1
VM2                                                     LAB1


VMName                                                  ComputerName
------                                                  ------------
Test-VM                                                 LAB2
VM3                                                     LAB2


PS C:\Users\Administrator.MGMT>
```

Figure 5. The PowerShell command being run against the remote servers is in brackets at the end.

The problem with this command is that it only looks at the virtual machines on the local server. If we wanted to run the command against multiple servers we would need to use the *Invoke-Command* cmdlet. To show how this works, imaging that we wanted to analyze three servers named Lab1, Lab2 and Lab3. To do so, we would use the following command:

```
Invoke-Command –ComputerName Lab1, Lab2, Lab3 {Get-VM | FT VMName,
ComputerName}
```

We started out by issuing the *Invoke-Command* cmdlet. Next, we used the *ComputerName* switch and provided the names of the servers that we wanted to run the command against. The actual command that is being run against the remote servers is encased in brackets at the end of the command (figure 5).

One important caveat when thinking about using Windows Server 2012 for multi-server management: In most cases the remote hosts must be running Windows Server 2012 in order to be remotely managed