第 23 章　高级远程控制配置

在第 13 章中，我们尽最大努力为你介绍 PowerShell 的远程控制技术。我们故意留下一些硬骨头，从而使得我们可以专注于远程控制背后的核心技术。但是在本章，我们希望重返该主题，并阐述一些更加高级和不常用的功能与场景。我们必须提前承认并不是本章所有的内容都能够派上用场——但是我们认为每个人都应该了解这些选项，以防以后对这些选项有需求。

同时，我们提醒你，本书主要内容是关于 PowerShell v3 以及之后版本，新版本关于远程控制的功能与之前版本的远程控制并无不同，关于找出当前运行的版本的办法，请重新查看第 1 章。本书涵盖的大部分内容无法在早期版本中运行。

23.1　使用其他端点

正如在第 13 章中所学，一台计算机可以包含多个端点。在 PowerShell 中，端点也被称为会话配置（session configurations）。举例来说，在 64 位机器上启用远程控制会同时为 32 位 PowerShell 和 64 位 PowerShell 各启用一个端点，其中 64 位 PowerShell 的端点是默认端点。

如果你拥有管理员权限，你可以在任何计算机上运行下述命令，获得可用的会话配置列表。

```
PS C:\> Get-PSSessionConfiguration

Name          : microsoft.powerShell
PSVersion     : 3.0
StartupScript :
RunAsUser     :
Permission    : NT AUTHORITY\NETWORK AccessDenied, BUILTIN\Administrators
```

```
                AccessAllowed
Name          : microsoft.powerShell.workflow
PSVersion     : 3.0
StartupScript :
RunAsUser     :
Permission    : NT AUTHORITY\NETWORK AccessDenied, BUILTIN\Administrators
                AccessAllowed
Name          : microsoft.powerShell32
PSVersion     : 3.0
StartupScript :
RunAsUser     :
Permission    : NT AUTHORITY\NETWORK AccessDenied, BUILTIN\Administrators
                AccessAllowed
```

每一个端点有一个名称；诸如 New-PSSession、Enter-PSSession、Invoke-Command 等远程控制命令默认使用其中一个名称为“Microsoft.PowerShell”的端点。在 64 位系统中，端点是 64 位的 Shell；在 32 位系统中，“Microsoft.PowerShell”是 32 位的 Shell。

你可以注意到，我们的 64 位系统有一个运行 32 位 Shell 的备用端点：“Microsoft.PowerShell32”用于兼容性目的。如果希望连接到备用端点，只需要在远程控制命令的 -ConfigurationName 参数中指定端点名称。

```
PS C:\> Enter-PSSession -ComputerName DONJONES1D96 -ConfigurationName
➥'Microsoft.PowerShell32'

[DONJONES1D96]: PS C:\Users\donjones\Documents>
```

什么时候会使用备用端点？需要显式通过 32 位的端点连接到 64 位的机器的可能原因之一是当需要运行的命令依赖于 32 位的 PowerShell 插件时。另外一种可能是存在自定义端点。当需要执行一些特定任务时，你或许需要连接到这些端点上。

23.2 创建自定义端点

创建一个自定义端点可以分为以下两步。

（1）通过 New-PSSessionConfigurationFile 命令创建一个新的会话配置文件，该文件的扩展名为 .PSSC。该文件用于定义端点的特征。特征主要指的是该端点允许运行的命令和功能。

（2）通过 Register-PSSessionConfiguration 命令载入.PSSC 文件，并在 WinRm 服务中创建新的端点。在注册过程中，可以设置多个可选参数，比如说谁可以连接到端点。也可以在必要时通过 Set-PSSessionConfiguration 命令改变设置。

我们将会带领你经历一个使用自定义端点进行授权管理的示例，这或许是PowerShell 最酷的功能之一。我们可以创建一个只有域中 HelpDesk 组的成员可以访问的端点。在端点内，我们启用与网络适配器相关的命令——并且只允许这些命令。我们并不打算给 HelpDesk 组运行命令的权限，仅仅是让他们可以查看命令。我们还配置端点从而可以在我们提供的备用凭据下运行命令，因此可以使得 HelpDesk 组可以在自身无须拥有执行命令的权限时执行命令。

23.2.1 创建会话配置

下面是我们运行的命令（我们将该命令格式化以便于阅读，但实际上，我们输入后只有一行）。

```
PS C:\> New-PSSessionConfigurationFile
  ➥-Path C:\HelpDeskEndpoint.pssc
➥  -ModulesToImport NetAdapter
➥  -SessionType RestrictedRemoteServer
➥  -CompanyName "Our Company"
➥  -Author "Don Jones"
➥  -Description "Net adapter commands for use by help desk"
➥  -PowerShellVersion '3.0'
```

这里有一些关键参数，我们已经用粗体重点标注。我们将会解释为什么我们赋了这些值。我们将阅读帮助找出这些参数其他选项的任务留给你。

- -Path 参数是必需的，并且你提供的文件名称必须以.pssc 结尾。
- -ModulesToImport 列出组件（在本例中，只有一个名称为 NetAdapter 的组件），我们只希望对于本端点只有该组件可用。
- -SessionType RestrictedRemoteServer 除了一些必需的命令，移除所有 PowerShell 的核心命令。该列表会很小，仅包括 Select-Object、Measure-Object、Get-Command、Get-Help、Exit-PSSession 等。
- -PowerShellVersion 默认为 3.0。在本例中，我们将该参数包含在内，只是为了完整性。

还有一些以-Visible 开头的参数，比如说-VisibleCmdlets。正常情况下，当你使用-ModulesToImport 导入一个组件时，所有该组件中的命令都会对于使用最终端点的人可见。通过只列出你希望人们看到的 Cmdlet、别名、函数、提供程序，非常有效地隐藏了其他内容。这是限制人们通过该端口所能做的操作的好办法。请小心使用 visibility 参数，这是因为该参数有一点让人迷惑。举例来说，如果你导入由 Cmdlet 和函数组成的组件，使用 VisibleCmdlets 仅仅限制能够显示的 Cmdlets——却不影响是否显示函数，这意味着这些函数在默认情况下都可见。

注意，没有任何方法可以对命令可用的参数进行限制：PowerShell 支持参数级别的限制，但需要在 Visual Studio 中进行大量编码。这超出了本书的内容。还有你可以使用的其他高级技巧，比如说创建用于隐藏参数的代理函数。但这超出本书的篇幅，因为本书的目标读者是初学者。

23.2.2 会话注册

创建完会话配置文件之后，可以通过下述命令使配置文件生效。我们再一次将代码格式化以便于阅读，但实际上只有很长的一行。

```
PS C:\> Register-PSSessionConfiguration
➥ -Path .\HelpDeskEndpoint.pssc
➥ -RunAsCredential COMPANY\HelpDeskProxyAdmin
➥ -ShowSecurityDescriptorUI
➥ -Name HelpDesk
```

这就创建了名称为 HelpDesk 的新端点。如图 23.1 所示，提示我们输入 COMPANY\HelpDeskProxyAdmin 账户的密码；该端点运行的所有命令都通过该账户的身份运行，我们需要确保该账户拥有运行网络适配器相关命令的权限。

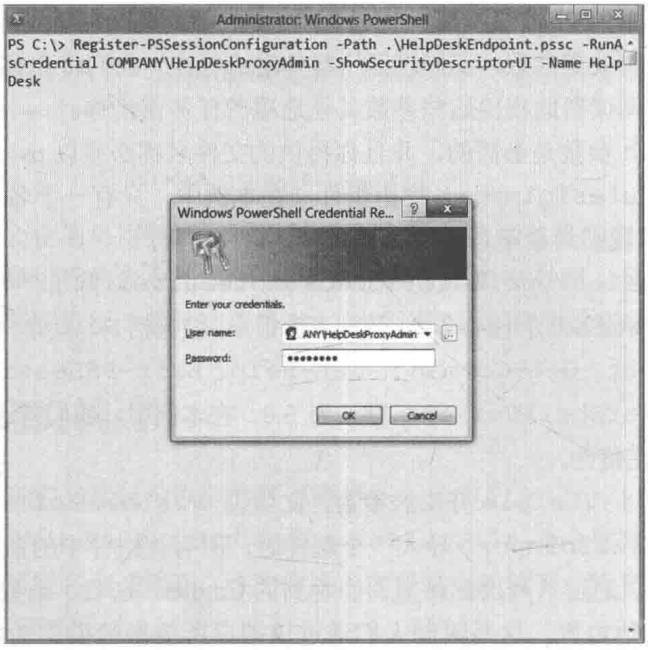

图 23.1 提示输入以凭据运行的密码

我们完成几个“是否继续运行”的提示，建议你仔细阅读提示。该命令会停止并启动 WinRM 服务，这会导致中断其他管理员管理本地机器，所以请小心。

如图 23.2 所示，该命令还为我们提供了图形化对话窗口指定哪个用户可以连接到端点。之所以会显示对话框，是由于我们使用了-ShowSecurityDescriptorUI 参数，而不是使用复杂的安全描述符定义语言（SDDL）设置权限。坦白讲，这也是我们不熟悉的语言。这同时是相对于 Shell 使用 GUI 方式更好的例子——我们将 HelpDesk 用户组添加在内，并确保该组拥有执行和读权限。执行是所需的最小权限，执行权限将我们计划给该账号的权限赋予端点；读权限是另一个我们需要的权限。

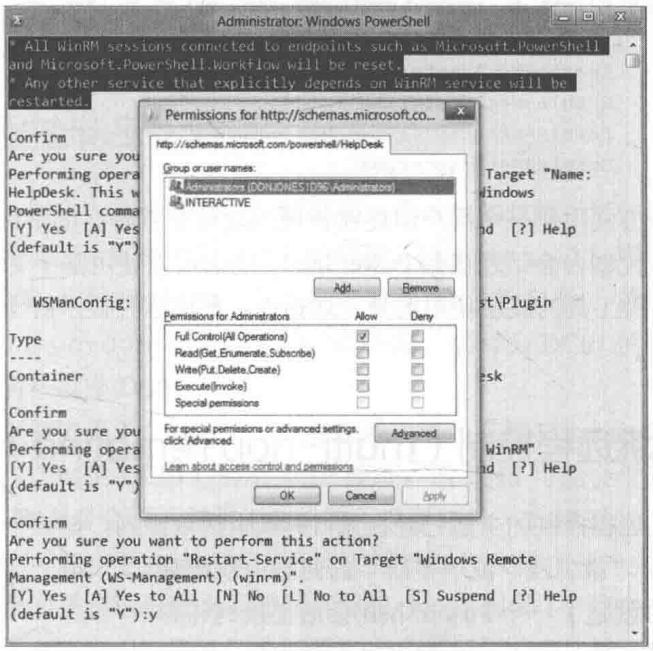

图 23.2 设置端点权限

基于我们完成的内容，可以看到下述输出结果（截断后的），使用新端点的用户只能使用非常有限的命令。

```
PS C:\> Enter-PSSession -ComputerName DONJONES1D96 -ConfigurationName
➥HelpDesk
[DONJONES1D96]: PS>Get-Command

Capability      Name                                                ModuleN
                                                                    ame
----------      ----                                                -------
CIM             Disable-NetAdapter                                  NetA...
CIM             Disable-NetAdapterBinding                           NetA...
CIM             Disable-NetAdapterChecksumOffload                   NetA...
CIM             Disable-NetAdapterEncapsulatedPacketTaskOffload     NetA...
CIM             Disable-NetAdapterIPsecOffload                      NetA...
CIM             Disable-NetAdapterLso                               NetA...
```

```
CIM             Disable-NetAdapterPowerManagement                NetA...
CIM             Disable-NetAdapterQos                            NetA...
CIM             Disable-NetAdapterRdma                           NetA...
CIM             Disable-NetAdapterRsc                            NetA...
CIM             Disable-NetAdapterRss                            NetA...
CIM             Disable-NetAdapterSriov                          NetA...
CIM             Disable-NetAdapterVmq                            NetA...
CIM             Enable-NetAdapter                                NetA...
CIM             Enable-NetAdapterBinding                         NetA...
CIM             Enable-NetAdapterChecksumOffload                 NetA...
CIM             Enable-NetAdapterEncapsulatedPacketTaskOffload   NetA...
CIM             Enable-NetAdapterIPsecOffload                    NetA...
CIM             Enable-NetAdapterLso                             NetA...
CIM             Enable-NetAdapterPowerManagement                 NetA...
CIM             Enable-NetAdapterQos                             NetA...
```

通过这种方式限制某个用户组能够使用的功能非常好。正如我们做的测试那样，他们甚至不必从控制台会话连接到 PowerShell，他们可以使用基于 PowerShell 远程控制的 GUI 工具。这类工具的底层使用的是上述命令，利用这种技术给予用户使用某些功能的权限再好不过。

23.3 启用多跳远程控制（multi-hop remoting）

启用多跳远程控制的主题已经在第 13 章中简单提到，但该主题值得进一步深入。图 23.3 描述了“第二跳”或“多跳”的问题：从计算机 A 开始，并创建了一个 PowerShell 会话连接到计算机 B。这是第一跳，通常该步骤可以正常工作。但当请求由计算机 B 再次创建第二跳，或者连接到计算机 C 时，操作失败。

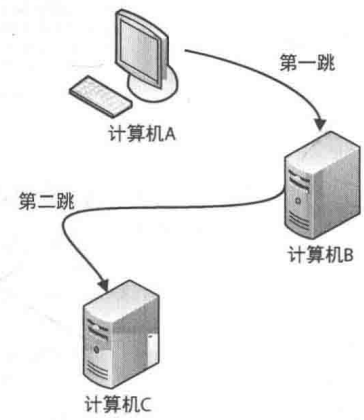

图 23.3 在 Windows PowerShell 中的多跳远程控制

问题是由于 PowerShell 将凭据由计算机 A 委托到计算机 B 时出现的。所谓委托，是使得计算机 B 以你的身份运行任务的过程，因此确保你可以在计算机 B 上做任何有权限做的事，但无法做权限之外的事。默认情况下，委托只能传输一跳；计算机并没有权限将你的凭据委托给第三台计算机，也就是计算机 C。

在 Windows Vista 以及之后版本，你可以启用多跳委托。该过程需要两步。

（1）在你的计算机（比如计算机 A）上，运行 `Enable- WSManCredSSP -Role Client -Delegate Computer x`。可以将“x”替换为希望将身份委托到的计算机名称。你可以指定具体的计算机名称，当然也可以使用通配符。我们不推荐使用*，这会

导致一些安全问题，但是可以对整个域进行授权，比如*.company.com。

（2）在第一跳连接到的计算机（比如计算机 B）上，运行 Enable-WSManCredSSP-Role Server。

通过上述命令所做的变更，将会应用到计算机的本地安全策略；你也可以通过组策略对象手动进行变更，在较大的域环境中可能需要这么做。通过组策略管理这些超过了本章篇幅，但你可以通过 Enable-WSManCredSSP 的帮助信息获得更多信息。Don 还写过一本书——*Secrets of PowerShell Remoting Guide*，在该书中对策略相关的元素进行了更详细的阐述。

23.4 深入远程控制身份验证

我们发现，很多人都会认为身份验证是一个单向的过程：当你访问远程控制计算机时，你必须在登录该计算机之前提供你的凭据。但 PowerShell 远程控制采用了双向身份验证，这意味着远程控制计算机必须向你证明它的身份。换句话说，当你执行 Enter-PSSession-computerName DC01 时，名称为 DC01 的计算机必须在连接建立完成之前证明它就是 DC01。

为什么？正常情况下，你的计算机将会通过域名系统（Domain Name System，DNS）将计算机名称（比如说 DC01）解析为 IP 地址。但 DNS 可能会受到电子欺骗的攻击，因此不难想象，攻击者会攻入并将 DC01 的入口指向另一个 IP 地址——一个受攻击者控制的 IP 地址。你可能在不知情的情况下连接到 DC01，实际上是一台冒名顶替的计算机，然后将你的凭据委托给这台冒名顶替的计算机——该倒霉了！双向身份验证会防止这类事发生：如果你连接到的计算机无法证明它就是那台你希望连接到的计算机，远程控制连接将会失败，这是好事——因此你不会希望在没有周密计划和考虑的情况下将这种保护关掉。

23.4.1 双向身份验证默认设置

微软期望更多是在域环境下使用 PowerShell。因此可以通过活动目录列出的实际计算机名称连接到计算机，域会为你处理双向身份验证。在访问其他可信任的计算机时也可以由域处理双向身份验证。该技巧需要你为 PowerShell 提供的计算机名称满足以下两点要求。

- 名称可以被解析为 IP 地址。
- 名称必须与活动目录中的计算机名称匹配。

提供你所在域的计算机名称，而对于可信域则需要提供完全限定名（也就是计算机和域名称，比如 DC01.COMPANY.LOC），这样远程控制通常就会生效。但如果你提供的是 IP 地址，或者需要提供与 DNS 中不同的名称（比如说 CNAME 别名），那么默认的双向身份验证将无法正常工作。因此你只有如下两种选择：SSL 或是“受信任的主机”。

23.4.2 通过 SSL 实现双向身份验证

要想通过 SSL 实现双向身份验证，你必须获得目标计算机的 SSL 数字证书。证书颁发给的计算机名称必须与你输入访问的计算机名称相同。也就是说，如果你运行 `Enter-PSSession -computerName DC01.COMPANY.LOC -UseSSL -credential COMPANY\Administrator`，那么安装在DC01 上的证书必须颁发给“dc01.company.loc”，否则整个过程就会失败。注意，`-credential` 参数在该场景中是强制参数。

在获取到证书之后，还需要将其安装到当前用户下的个人证书存储目录——通过微软管理控制台（Microsoft Management Console，MMC）界面是导入证书的最佳方式。仅仅是双击证书，通常情况下也能够将证书导入到账户的个人目录之下，但不通过 MMC 导入证书对 SSL 连接不会生效。

在完成证书安装之后，你需要在计算机上创建一个 HTTP 侦听器，并告诉侦听器使用刚刚安装的证书。而详细的指导教程会很长。由于这并不是大部分人会去配置的工作，我们在此不会将该部分内容包含在内。查看 Don 的 *Secrets of PowerShell Remoting Guide*（免费），你可以在此书中找到包含截图的详细教程。

23.4.3 通过受信任的主机实现双向身份验证

该技术比使用 SSL 证书略微简单，需要的配置步骤也会少很多。但该方式更加危险，这是由于该技术主要是对于选定的主机关闭双向身份验证。在开始之前，你需要能够自信地声明“不会有任何人会冒充这几台主机中的任何一台，或者入侵 DNS 记录”。对于在内部局域网的计算机来说，你或许可以确保这一点。

然后你仅需一种方式去识别哪些计算机不需要双向验证。在一个域中，这或许是类似“`*.COMPANY.COM`”这样在 Company.com 域中的所有主机。

这是你需要配置整个域设置的一个实例，所以我们给你一个操作组策略的指南。该指南对于单机中的本地安全策略同样有效。

在任意 GPO 或本地计算机策略编辑器中，执行这些步骤。

（1）展开计算机配置。

（2）展开管理模板。

（3）展开 Windows 组件。

（4）展开 Windows 远程控制管理。

（5）展开 WinRM 客户端。

（6）双击受信任的主机。

（7）启用策略并添加信任的主机列表，多个条目可以通过逗号分隔，比如“`*.company.com,*.sales.company.com.`”。

注意： 老版本的 Windows 版本可能没有在本地计算机策略中显示上述设置所需的模板，旧的域控制器的组策略对象中或许没有这些设置。对于这种情况，你可以在 PowerShell 中修改受信任的主机。在 Shell 中运行 `help about_remote_troubleshooting` 获取帮助。

现在你就可以在没有双向身份验证拦截的情况下连接到这些计算机。所有用于连接到这些计算机的远程控制命令中必须提供`-Credential` 参数——如果不这么做，可能会导致连接失败。

23.5 动手实验

注意： 对于本次动手实验来说，你需要 Windows8 或 Windows Server 2012 或更新版本的操作系统，从而运行 PowerShell v3 或更新版本。

在本地计算机创建一个名称为 TestPoint 的端点。将端点配置为仅自动载入 SmbShare 组件，但该组件只有 `Get-SmbShare` 命令可见。同时要确保类似 `Exit-PSSession` 的关键 Cmdlet 可见，但不允许使用其他核心 PowerShell Cmdlet。

通过 `Enter-PSSession`（指定 localhost 作为计算机名称，TestPoint 作为配置名称）连接到该端口，对该端口进行测试。当连接成功后，运行 `Get-Command`，从而确保只有少数配置为可见的命令可以被发现。

注意，本次动手实验可能只在 Windows 8、Windows Server 2012 以及更新版本的 Windows 上可做——SmbShare 组件并没有与更老版本的 Windows 一起发行。

23.6 动手实验答案

```
#create the session configuration file in the current location
#this is one long line
New-PSSessionConfigurationFile -Path .\SMBShareEndpoint.pssc
     -ModulesToImport SMBShare -SessionType RestrictedRemoteServer
     -CompanyName "My Company" -Author "Jane Admin"
     -Description "restricted SMBShare endpoint" -PowerShellVersion '4.0'

#register the configuration
Register-PSSessionConfiguration -Path .\SMBShareEndpoint.pssc -Name TestPoint

#enter the restricted endpoint
Enter-PSSession -ComputerName localhost -ConfigurationName TestPoint
get-command
exit-pssession
```