Work in Progress - Automation of a Computer Networking Laboratory

Neville Palmer
Maritime and Technology Faculty
Southampton Solent University
Southampton
neville.palmer@solent.ac.uk

Abstract— **A dedicated networking laboratory is used by students on a computer networking degree program so that they can undertake practical work without affecting the main University network. Images of the laboratory computers are maintained on a server and deployed when major updates or reconfigurations of the computers are required. Improvements can be made in the configuration and management processes. Automation can help realize some of these improvements. A server based application for automating the networking laboratory, named Remote Configuration Console, has been developed using PowerShell scripts and Microsoft Visual C#. This paper discusses the design and development of the application and its effectiveness in improving the configuration and management processes. This paper concludes that the application has been successful in improving the efficiency of processes and that the knowledge gained during development can be used to inform the curriculum of the computer networking program itself. Future work will involve investigating how this can be achieved and also validating the results of this work during the academic year.**

***Keywords*— Computer Networking Laboratory; PowerShell; Visual C; Automation; Configuration Management**

I. Introduction

The computer networking degree program benefits from a dedicated networking laboratory isolated from the rest of the University network. It consists of a number of computers running virtual machines that can be patched to networking devices such as Cisco routers and switches. The layout is not unlike other networking laboratories. Whilst other research has focused on the logical structure and physical layout of the laboratory [1,2] this work concentrates on the configuration management of the laboratory computers and how this task in itself can be used for educational purposes. Two objectives relating to the network were considered as part of this study. The first to improve the management and automation of the network itself to improve productivity, and the second as part of ongoing research in this area whose outcomes can be used to inform the computer networking degree curriculum.

A central server has been installed to provide IP address allocation and other services to the laboratory network. In order to manage the build of the laboratory computers it was found necessary to maintain an image on the server for deployment within the laboratory. Whilst systems such as Microsoft Remote Installation Service (RIS) [3], Imagecast [4] and Clonezilla [5] have been investigated for deployment of images Symantec Ghost Solution Suite was eventually chosen as being more suitable and flexible for this purpose [6]. The benefits of allowing the laboratory computers to operate independently within the laboratory were seen to outweigh the benefits of some features of Active Directory such as software deployment, though it may be beneficial to investigate Windows Deployment Services in the future [7,8].

Virtual machines based on VMware are deployed on the laboratory computers, allowing access to other operating systems such as Linux, though the host operating system is Windows based. Whilst a method of hosting virtual machines on the server has been tested the current practice is to host them on the laboratory computers directly. This configuration increases the time required to deploy a complete image to the laboratory computers, but this can be mitigated by adopting the approach of placing each virtual machine on its own partition and only re-imaging the relevant partition when changes or updates are required. Students work on a linked clone of the virtual machine, so that this clone can be rebuilt from the master (re-cloning) if it needs to be returned to a default state without the need to re-image the partition [9].

II. Use of the Laboratory in Education

The networking laboratory is fully equipped with Cisco routers, switches and other networking devices. These can be accessed from the virtual machines hosted on the desktop computers within the laboratory. The laboratory is used for various educational tasks. It is mainly used for the teaching of practically based networking systems, the primary user being the computer networking degree program. The aims of this course are to produce graduates with an even mix of technical knowledge, practical skills and independent research capabilities to meet the demands of business and industry, and also the requirements of post graduate study. The objectives are to produce graduates typically pursuing careers as computer network engineers and managers, or continuing on postgraduate study up to doctorate level. The skills required to meet these objectives are theory and practice of computer networking concepts, network design, planning, implementation and management, which includes elements of physical and logical networking. The latter involves network operating systems, including network administration and scripting. Students would also benefit from knowledge of computer programming, particularly relating to development of network applications and network administration interfaces. In the final year of their degree students are expected to work on a major individual project that should build on knowledge gained over the previous years of study and develop ideas further through independent research. The most obvious concept for a relevant project might be the design and implementation of a computer network, however care must be taken in ensuring that the aims and objectives are sophisticated enough to meet the demands of a final year project. A typical project based around network design may involve implementation of a network as a software model, using applications such as Riverbed Modeler [10], in which the results of traffic simulation can be analyzed and used to optimize network design. Most graduates will be users of existing systems and software, though they may often use customized scripts and may occasionally find it useful to develop software to meet a particular need. If they continue on to postgraduate study they may be engaged in developing new systems. Therefore knowledge of scripting and computer programming languages relevant to computer networking would be desirable, and students may wish to focus on this area in their final year project, particularly if they intend to continue onto post graduate study.

As well as employing real network hardware in the teaching of computer networking, extensive use is made of Cisco Packet Tracer for simulation of networks [11,12], however Graphical Network Simulator 3 (GNS3) has also been trialed. In order to enable them to interact with GNS3 loopback adapters have been configured on the lab computers [13]. Different sessions make use of various configurations and virtual machines, for example Windows or Linux. Where several classes use the computer networking laboratory configuration problems can be a major concern due to conflicting resource sharing demands. For example one class might configure the network equipment and virtual machines in one way which might conflict with the requirements of another class. Whilst use of simulators can help to reduce the problems associated with hardware reconfiguration issues, the configuration of laboratory computers needs to be managed, so that they can be returned to a default condition quickly between conflicting sessions. So it was found that the tasks of re-cloning virtual machines and re-configuring adapters need to be performed on a regular basis. Software must also be installed and updated when the need arises.

Students study network administration and operating systems, which involves investigating and configuring operating systems and server roles. They may for example consider the principles behind Dynamic Host Configuration Protocol (DHCP) and how a DHCP server is deployed. They are also introduced to Linux scripting and it was also seen as beneficial to introduce some computer programming concepts. Visual C# is used to introduce fundamental concepts within Windows [14,15], since some students are already familiar with this, and QT with C++ for Linux [16][16]. It is then possible to encourage more creativity and develop a better understanding of the subject by, for example, demonstrating how to develop a basic Linux administration interface using QT to configure a DHCP Server [17]. An important consequence of this is that students can take this further by developing a more sophisticated administration application as the subject of a final year project dissertation.

III. The Need for Automation

Due to the varied laboratory usage the management tasks can occupy a great deal of staff time. So the networking laboratory would benefit from automation in the form of a configuration management solution. The use of DeepFreeze had been considered as it is used on other parts of the Faculty network, however it was found not to be flexible enough for the needs of the laboratory [18]. Other configuration management applications such as Puppet, Chef or Ansible are available [19-21]. They use PowerShell scripts to manage the configuration of computers on a network, whilst workload automation or job scheduling systems like JAMS and ActiveBatch are capable of scheduling PowerShell scripts [22,23]. The lab computers may benefit from utilizing VMware ESX to host virtual machines. Whilst this may allow increased performance vsphere allows management of virtual machine hosts and the Power Command Line Interface (PowerCLI), which is based on PowerShell scripting technology, allows management of tasks [24]. It would be useful for students to study at least one of these systems as they may encounter similar ones during their career.

However it would be desirable for students to learn how to use PowerShell scripts in Windows and how to develop scripts of their own. It would also be desirable for students to examine how configuration management applications work by learning how to develop one for themselves, rather than just making use of an existing system. Powershell scripts incorporated into a customized application could make a useful case study for this purpose. This was the reasoning that led to the development of a customized application for configuration management within the laboratory.

IV. Requirements of Automation

When re-imaging the laboratory computers the GhostCast server requires that they should boot up to a Pre-boot Execution Environment (PXE) server [6]. As part of the process it would be useful to reboot the computers in the lab remotely from the server.

If students have been working with a virtual machine on a lab computer it may need to be rebuilt afterwards to return it to a default state. This can be done by re-cloning the virtual machine from the master rather than having to re-image the whole partition from the server. It would be useful to be able to automate this process, for example by re-cloning on Friday evenings after classes have finished. Whilst this process could be scheduled from within the Task Scheduler on each lab computer, it would be beneficial to run this task from the server so that the timing of the task could be changed. In order to ensure that no virtual machines are running prior to re-cloning the lab computers could be rebooted from the server.

When the loopback adapters are being used in laboratory sessions that require them, for example when using the GNS3 simulator, ideally they need to be enabled and then disabled afterwards so that they do not conflict or cause confusion with the real network adapter. It would also be useful to configure the IP address, gateway and DNS server settings on each loopback adapter according to the requirements of the laboratory session [13]. It would be desirable to be able to configure the adapters on the lab computers remotely from the server rather than to have to rely on re-imaging to update their configuration.

Occasionally new folders have to be installed on the laboratory computers, for example this may be to run a time constrained assignment using simulation software such as Packet Tracer or GNS3. Rather than re-imaging or copying the folder manually onto each computer it would be beneficial to be able to copy the folder directly from the server to all computers at once.

Whilst energy saving measures could be configured on each lab computer it would be beneficial if this could be controlled from the server, for example to shut down all lab computers from the server at night time from a task within the Task Scheduler.

The initial functional requirements for the PowerShell scripts can be summarized as follows. PowerShell scripts must allow the following actions to be performed on all laboratory computers centrally from the server when required:

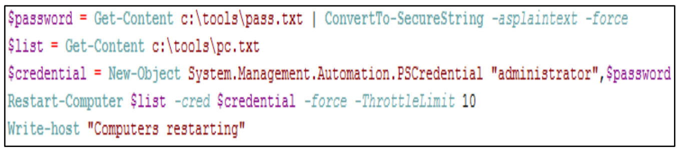
1. Reboot the computers.
2. Enable Virtual Machines on the computers to be re-cloned.
3. To enable or disable loopback adapters, and configure IP, Gateway and DNS server on the loopback adapters.
4. To copy a folder from the server to the computers.
5. To shutdown the computers.
6. Scripts must either be run manually or at certain times from the Task Scheduler on the server.

V. Implementing PowerShell Scripts

Simple tasks like folder and user management were performed with some success using DOS batch scripts in the past, but there is a limit to their capabilities. Therefore PowerShell scripts were developed to customize the automation of the networking laboratory, which included some of the important configuration, re-imaging and re-cloning tasks so that they could be performed remotely from the server at times when the laboratory was not in use.

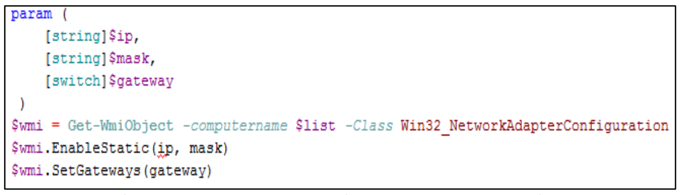
The first script, shown in Fig. 1, enabled the laboratory computers to be restarted remotely in order to assist the re-imaging process. This used the Restart-Computer command to act on a list of computers and a similar script was used with Stop-Computer to shut down all lab computers at a certain time using Task Scheduler on the server. Another script made us of Invoke-Command to remotely run a batch file on each lab computer that re-cloned the virtual machines from the server at scheduled times. Invoke-Command was also used to map a network drive and copy a folder from the server to the lab computers when required [25].

A script, shown in Fig. 2, was constructed to change the address of the loopback adapters on the laboratory computers from the server. This made use of PowerShell commands associated with Windows Management Instrumentation, such as Get-WmiObject. The WMI method EnableStatic is used to set the IP address and subnet mask of the adapter, and SetGateways for the gateway [26].



```
$password = Get-Content c:\tools\pass.txt | ConvertTo-SecureString -asplaintext -force
$list = Get-Content c:\tools\pc.txt
$credential = New-Object System.Management.Automation.PSCredential "administrator",$password
Restart-Computer $list -cred $credential -force -ThrottleLimit 10
Write-host "Computers restarting"
```

Fig. 1. PowerShell script to remotely restart computers.



```
param (
    [string]$ip,
    [string]$mask,
    [switch]$gateway
 )
$wmi = Get-WmiObject -computername $list -Class Win32_NetworkAdapterConfiguration
$wmi.EnableStatic(ip, mask)
$wmi.SetGateways(gateway)
```

Fig. 2. Part of PowerShell script to configure IP addressing information.

The outcome of this work has produced a repository of useful PowerShell scripts and there is scope for further work.

VI. Productivity and Efficiency

Before scripts were implemented changes to virtual machines would have to be made on the test bed computer, then the image uploaded to the server, before being deployed to the computers in the lab. Typically the time to deploy a new image to the lab computers is about 15 minutes for one partition. On the other hand if only a fresh clone of the virtual machine is required the script that does this can complete this on all virtual machines within the lab in under a minute. Provided that an update to the virtual machine's software is not required the script will suffice. This script can also be scheduled to run automatically on certain days of the week. If a laboratory session requiring the loopback adapters is to be configured without scripts then someone must either go to each laboratory computer and change the loopback adapter IP addresses manually, or configure the adapters on the test bed computer, save the image and then deploy it to the lab computers. The latter method takes at least 30 minutes plus the time taken to modify the adapter IP addresses on the test computer. Whereas if a script is used it is only necessary to modify the script and deploy it – a process that typically takes 5 minutes. The same applies to transferring a folder to a lab computer.

The lab computers are not used overnight and if there is no automated means of shutting them down they could remain powered on unnecessarily for every night and weekend of the academic year. This could amount to over 3,200 hours of wasted energy, which can be saved by running a shutdown script just after 9pm every day.

So there are benefits for using PowerShell scripts in terms of increased productivity and energy efficiency. Though there is a learning curve for users of the scripts where they are not incorporated into Task Manager schedules because management and usage of the scripts requires specialist knowledge.

VII. Aggregating Scripts Into an Application

Whilst the scripts are stored in a common directory on the server some knowledge of what each script does, how to modify it and how to use it is required to use them properly. It would be beneficial to be able to access the scripts from a single menu driven system that requires little understanding of the script process itself. It would also be beneficial to allow a means to store various configurations for later retrieval. This system could also be used to demonstrate the development and functionality of configuration management software to students.

It was decided to investigate how the scripts could be aggregated into a more user friendly interface. Visual C# was chosen as a Windows Forms [15] interface could easily be constructed that called the existing PowerShell scripts. With a Windows interface it was also possible to allow the network administrator to select laboratory computers to apply scripts to from a list rather than forcing all computers to receive the scripts.

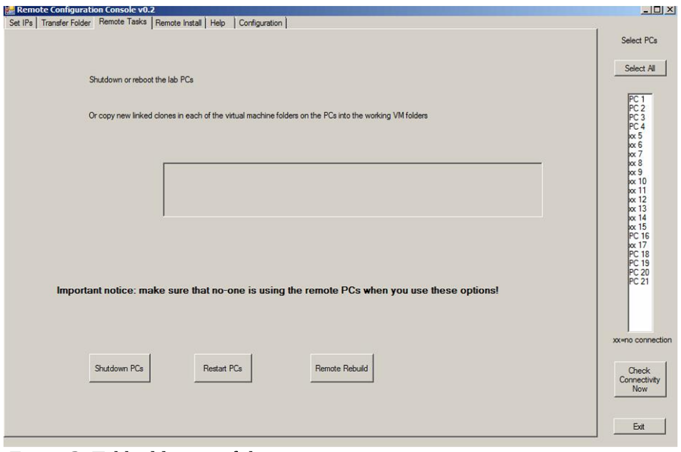
The key requirements of a visual interface are similar to those for PowerShell scripts, but incorporation within an interface allows more centralized control. The interface must allow the user to perform the following tasks:

1. select which laboratory computers to apply scripts to.
2. select which loopback adapters on computers to enable or disable, and to be able to enter IP address, gateway and DNS server for each adapter. To be able to deliver this configuration to selected computers, and to be able to save and retrieve configurations.
3. select a folder on the server and transfer it to the selected computers.
4. shutdown or restart selected PCs from the server.
5. reclone the Virtual Machines on selected computers.
6. install Windows Installer packages on selected computers.
7. save application configuration information such as usernames and passwords for remote access and the base IP address of lab computers.
8. There must be a help system to assist the user of the application in understanding how to use it.

A further requirement called for the shutdown, restart and re-cloning functions to be capable of being scheduled, but this was not initially incorporated in the prototype.

There were three goals associated with the development of the application. The first goal was to improve efficiency of configuration management within the networking laboratory, the second to demonstrate to students how the software is used in this environment and the third to demonstrate how to develop this type of application.

The new program, shown in Fig. 3, used a tabbed layout in Windows Forms. A list of lab computers appears in a listbox on the right hand side of the main form. Computers that do not have connectivity to the server are marked with an "x". Configurations can only be delivered to those computers that have connectivity and that have been selected from the list.
The functions to shutdown or restart the lab computers, along with the ability to re-clone or rebuild the virtual


Remote Configuration Console v0.2

Shutdown or reboot the lab PCs

Or copy new linked clones in each of the virtual machine folders on the PCs into the working VM folders

Important notice: make sure that no-one is using the remote PCs when you use these options!

xx=no connection

Figure 3. Tabbed layout of the new program.

machines were incorporated into a Remote Task tab on the form. They call the PowerShell scripts developed previously.

The PowerShell scripts were constructed dynamically within the program based on the state and content of the form controls, for example from text within text boxes and from the content of list boxes. A text box may for example contain an IP address and a list box may contain a list of selected computers that a script will act upon. For debugging purposes a text box was placed on the form to show the constructed commands, which would also be useful for demonstrating how the program works.

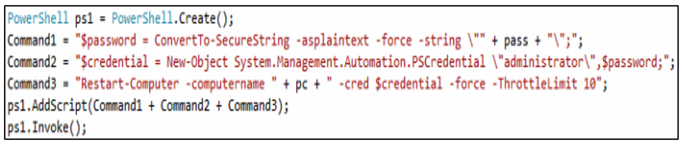
To run PowerShell scripts from within a Visual C# application it is necessary to add a reference to the System.Management.Automation assembly reference and then add the corresponding namespace within the application, along with the System.Collections.Objectmodel namespace [26]. PowerShell scripts are then executed by means of the Invoke method within the PowerShell class.

An example of the syntax for building and calling a PowerShell script from within C# can be seen within the Restart PC click method in Fig. 4. Strings are concatenated to form the PowerShell script commands. A PowerShell object is created which is invoked once the commands have been added to the script using the AddScript method. Target computers are taken from the listbox on the right.

All other PowerShell scripts used within the program are built and called in a similar way. Another tab within the application allowed the IP address configurations of the loopback adapters on the lab computers to be set from the server, including enabling and disabling them. The user can also save a particular configuration and retrieve it for later use.

A tab to allow for the transfer of folders from the server to the lab computers was also implemented. A further tab allows the installation of software, using a Windows Installer Package, on the lab computers by means of a remote call to "msiexec" [27].

A Configuration tab was included to enable default configuration information to be saved, such as passwords stored as a hash value.



```
PowerShell ps1 = PowerShell.Create();
Command1 = "$password = ConvertTo-SecureString -asplaintext -force -string \"" + pass + "\";";
Command2 = "$credential = New-Object System.Management.Automation.PSCredential \"administrator\",$password;";
Command3 = "Restart-Computer -computername " + pc + " -cred $credential -force -ThrottleLimit 10";
ps1.AddScript(Command1 + Command2 + Command3);
ps1.Invoke();
```

Fig. 4. Calling a PowerShell script from within C#.

VIII. Educational use of the Scripts and Application

The Remote Configuration Console application could be used as the basis of case study on relevant units within the networking degree program provided students have adequate preparation. As part of the first year students are introduced to Linux scripting. The concepts of Windows PowerShell scripts could also be introduced here. The basic concepts of programming using Visual C# for Windows and QT and C++ for Linux are also introduced. In the second year this is taken further in the development of an administration interface for Linux, which can then be used to inform final year projects. Therefore students will already be familiar with Visual C# by the second year of their studies. At this point the concepts of incorporating PowerShell scripts into a Visual C# interface could be introduced. The debugging window within the Remote Configuration Console demonstrates the syntax of PowerShell commands that are constructed by the program prior to delivery to the laboratory computers. This can be used to demonstrate to students the syntax and function of PowerShell commands and how the program formulates these commands into a coherent script.

The outcomes of the research in developing a customized configuration management and automation interface, based on the use of PowerShell scripts, can be used to inform the curriculum of the degree program. Further work will need to investigate the best approach to achieving this. Some of the PowerShell scripts used within the Remote Configuration Console are quite powerful and some caution may be required when using them within the classroom. For example it may not be desirable to encourage the widespread usage of remote shutdown or restart scripts. Perhaps more benign scripts could be developed to demonstrate the principles applied to network management.

IX. Conclusions and Recommendations

This study has investigated methods for improving productivity within the computer networking laboratory using PowerShell scripts within a graphical user interface, whilst simultaneously building a repository of knowledge in this area.

The independent PowerShell scripts have been tested extensively in the laboratory and tests of the Visual C# program into which they are aggregated have initially indicated significant improvements in efficiency within the networking laboratory, though it will need to be tested over the course of an academic year to further validate the results.

Additional features and improvements have been incorporated into the latest version of the program. These features include the ability to schedule tasks such as shutting down or restarting the laboratory computers or re-cloning virtual machines at certain times. This allows these functions to be performed automatically without operator intervention within the program. For example the virtual machines can be re-cloned on all computers on Friday evening and they can be shut down over the weekend.

We now have a useful body of knowledge that can be used to teach students how to make effective use of PowerShell scripts and how they can be used within an application to manage a network. Learning materials will need to be developed to enable learners to benefit from this work. It is then hoped that some students will go on to develop these concepts in a final year project. Further work will need to investigate how this final objective can be achieved and to examine the success of the chosen approach.

Acknowledgements

Thanks to Dr. Nick Whitelegg, Jomo Batola, Prins Butt and Warren Earle at Southampton Solent University for technical assistance in the production of this paper.

References

[1] Design of a computer networking laboratory for efficient manageability and effective teaching. Frontiers in Education Conference, 2009. FIE '09. 39th IEEE; 2009.
[2] Prieto-Blázquez J, Arnedo-Moreno J, Herrera-Joancomartí J. An Integrated Structure for a Virtual Networking Laboratory. Industrial Electronics, IEEE Transactions on 2008;55[6]:2334-2342.
[3] Russel C, Crawford S. Windows 2000 Server Administrator's Companion. USA: Microsoft Press; 2000.
[4] Deignan M. "ImageCast IC3". 1999; Available at: http://windowsitpro.com/windows/imagecast-ic3. Accessed September 10th, 2014.
[5] Clonezilla.org. "ClonezillaThe Free and Open Source Software for Disk Imaging and Cloning". nd; Available at: http://clonezilla.org/clonezilla-SE/. Accessed September 10th, 2014. .
[6] Symantec. "Symantec Ghost Implementation Guide". 2008; Available at: ftp://ftp.symantec.com/public/english_us_canada/products/symantec_ghost_solution_suite/2.5/manuals/Ghost_imp_guide.pdf. Accessed September 9th, 2014.
[7] Microsoft. "Windows Deployment Services for Windows Server 2008 R2". 2009; Available at: http://technet.microsoft.com/en-us/library/dd348502%28v=ws.10%29.aspx. Accessed September 9th, 2014.
[8] Minasi M. Mastering Windows Server 2008. Indianapolis, Ind.: Wiley ; 2008.
[9] VMware. "Using VMware Workstation". 2013; Available at: https://www.vmware.com/pdf/desktop/ws1001-using.pdf. Accessed September 9th, 2014.
[10] Riverbed. "Introduction to Riverbed Modeler Academic Edition". n/d; Available at: https://splash.riverbed.com/servlet/JiveServlet/download/4833-2-6165/Introduction%20to%20Modeler.pdf. Accessed January 3rd, 2015.
[11] Dye MA, McDonald R, Rufi AW. Network fundamentals : CCNA exploration companion guide. Indianapolis, Ind.: Cisco Press; 2008.
[12] CISCO NA. CCNA security lab manual version 1.1. Indianapolis, Ind.: Cisco Press; 2012.
[13] GNS3. "Connecting GNS3 to Real Networks". 2014; Available at: http://www.gns3.net/documentation/gns3/connecting-gns3-to-real-networks/. Accessed September 10th, 2014.
[14] Sharp J. Microsoft Visual C# 2010 step by step. Redmond, Wash.: Microsoft; 2010.
[15] Dorman S. Sams Teach Yourself Visual C# 2010 in 24 Hours. Indianapolis: Sams; 2010.
[16] Blanchette J, Summerfield M. C++ GUI programming with Qt 4. Upper Saddle River, N.J. London: Prentice Hall; 2008.
[17] Davies J, Whittaker R, Von Hagen W. SUSE Linux 10 bible. Indianapolis, Ind.: Wiley; 2006.
[18] Faronics. "Product Data Sheet Faronics Deep Freeze Server". 2014; Available at: http://www.faronics.com/assets/DSE7-5_V1_SpecSheet.pdf. Accessed September 9th, 2014.

[19] Puppet Labs. "Docs: Learning Puppet". 2014; Available at: https://docs.puppetlabs.com/learning/. Accessed September 11th, 2014.
[20] Getchef.com. "An Overview of Chef". nd; Available at: http://docs.getchef.com/chef_overview.html. Accessed September 11th, 2014.
[21] Ansible.com. "Ansible in Depth". 2014; Available at: http://cdn2.hubspot.net/hub/330046/file-480366556-pdf/pdf_content/Ansible_in_Depth.pdf?t=1410536005388. Accessed September 11th, 2014.
[22] Scott M. "The Shortcut Guide to IT Workload Automation and Job Scheduling". 2009; Available at: http://www.advsyscon.com/ebook/chapters/full.pdf. Accessed September 15th, 2014.
[23] Advanced Systems Concepts I. "PowerShell Integration". 2014; Available at: http://www.advsyscon.com/home/products/activebatch/powershell.aspx. Accessed September 15th, 2014.
[24] VMware. "VMware ESX and VMware ESXi". 2009; Available at: http://www.vmware.com/files/pdf/VMware-ESX-and-VMware-ESXi-DS-EN.pdf. Accessed September 10th, 2014.
[25] ss64.com. "An A-Z Index of Windows PowerShell 2.0 commands". 2014; Available at: http://ss64.com/ps/. Accessed September 10th, 2014.
[26] Siddaway R. PowerShell and WMI. New York: Manning; 2012.
[27] Microsoft. "Msiexec". 2014; Available at: https://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/msiexec.mspx?mfr=true. Accessed January 2nd, 2015.