Lab: Enabling PSRemoting

# Instructions

Work together with your partner to complete these lab activities. Prepare your own report as a Microsoft Word document, in which you explain the work you and your partner did to accomplish this lab. Where appropriate, please illustrate your report by including screen-capture evidence showing that you accomplished each lab task.

*(DO NOT submit one jointly-authored document. Each team member must write and submit their own individual report. DO NOT just submit a sequence of screenshots; you must submit a professionally written report that explains what you did, what you learned, and what specific evidence each screenshot illustration conveys.)*

# Overview

For this lab, you enable and configure computers to support PowerShell remoting, as presented in chapter 3 of the textbook.

Your instructor has provisioned two new virtual machines (a Windows Server and a Linux server) just for you and your team. They are not and will not be joined to a domain; instead, they will remain independent hosts on the domain’s subnet. You can remote-access them with SSH, but PowerShell remoting has not yet been enabled on these hosts. You are tasked with reconfiguring these two hosts, as follows:

* Log in to the local administrator account on each machine, provision new users on each machine (a user account for each team member), configure the new users with administrator privileges, then log out and log back in as the new users (to use the new accounts for the rest of this lab).
* Using new host names provided by your instructor, rename and reboot each machine.
* Enable PSRemoting on each machine, then verify that you can use cmdlets **Invoke-Command**, **New-PSSession**, etc. to remotely operate each host from the other, and also back and forth between each machine and the jump server.

To complete this lab, use the instructions in textbook chapter 3, as well as the PSRemoting skills you practiced in your homework exercises. Also, please leverage Internet resources at your disposal, including advanced search engines and generative-AI LLM chatbots. Collaborate with your partner and share successful searches and chatbot prompts. Include your most useful search queries and chatbot conversations in your report. *(Beware: generative AIs almost always share suggestions that do NOT produce correct results on your new VMs!)*

Your instructor will provide the IP addresses of the two machines assigned to you and your lab team, along with their new hostnames. Record them at the beginning of your report.

# Requirements

* SSH client
* Internet access

# Setup

Use SSH as a remote access tool to connect to the lab’s “jump server:”   
Computer: **cit361-lab.cit.byui.edu**Port: **22**   
Username: The mailbox portion of your BYU-I email address, usually three letters and five numbers. (Example: If your BYU-I email is **mer23079@byui.edu** you would use **mer23079** for your username.)  
Password: Your I-Number.

If you are using OpenSSH from a CLI, the parameters above can be specified like this:  
**ssh mer23079@cit361-lab.cit.byui.edu**

# Tasks:

## Connect to each of your team’s lab VMs

1. After using SSH to log in to the “jump server,” use SSH again to connect to your team’s Windows Server. It’s an independent server (it’s not on a domain), but it has a *local* account: **ps** is the username, **PowerShellRocks!** is the password.   
   **ssh ps@***<ip address>*   
   Figure out how to obtain elevated *local administrator* privilege on your independent Windows Server. (Hint: try the same password.)
2. Back on the jump server, use SSH to connect to your team’s independent Linux server. Its local account also has the same **ps** username and **PowerShellRocks!** password. Figure out how to obtain elevated *root* (local superuser) privilege on your Linux Server. (Hints: try the same password.) (Another hint: the actual privilege elevation procedure is *not* the same for these two tasks!)

## Create your own local user accounts on your team’s lab VMs

1. Figure out and use PowerShell commands to create your own *local* user account on your independent Windows Server. Figure out and use Linux shell commands to create your own local user accounts on your Linux server. Your team partner(s) should create their own local user accounts on your hosts. (Help them, but don’t do it for them!)
2. Figure out how to configure your new local user accounts with the capability to use elevated (administrator or root) privilege to execute commands. *Your administrator capability must be auditable*, in the sense that privileged access from your account is secured using AAA (authentication, authorization, accounting) controls. Hints:
   * On your Linux servers, the **sudo** command already implements local AAA capability.
   * Do *not* edit the **sudoers** file; you don’t need to. Figure out what to do!
   * Generative AI chatbots have been consistently *wrong* about how you should configure **sudo** privilege on the lab’s Linux servers. Figure out what to do!
   * For your Windows server, see textbook chatper 4 for a few command-logging ideas. Choose and try to implement one of them as an accounting control. (For some administration events, such as rebooting, see if you can observe any event patterns recorded in event logs.)

## Rename and reboot your team’s lab VMs

1. Figure out and use PowerShell commands or other configuration tools to change the hostname of your team’s independent Windows and Linux servers. (Your instructor should have specified what your new hostnames should be.)
2. Restart and verify that each new hostname has taken effect.
3. Verify that your independent servers’ system logs or event logs have records that show the hostname change and/or reboot events.

## Enable PowerShell remoting on your team’s lab VMs

1. Figure out and use PowerShell to create objects of type PSCredential. Create a valid credential object that represents your domain account on the jump server. Create valid credential objects that represent your new local account on both of your team’s independent lab VMs.
2. Figure out and use PowerShell commands and other configurations to enable PowerShell remoting between your independent machines. Hints:
   * Configure **TrustedHosts**
   * Try using your PSCredential objects to authenticate your PSRemoting attempts.
   * You need to configure a PowerShell subsystem for the Linux VM’s SSH daemon **sshd**.
3. Verify that you can create and use PSSession objects to remotely execute a script block between each of your team’s independent servers.
4. Verify that you can create and use PSSession objects to remote between the jump server and each of your team’s independent servers.

# Deliverable

Again, as mentioned in the instructions above:

Prepare your own team’s report as a Microsoft Word document, in which you explain the work you and your partner did to accomplish this lab. Illustrate your report by including screen-capture evidence showing that you accomplished each lab task. Copy the most useful search queries and chatbot conversation prompts that you and your partner found most helpful, and include them in your report. *(DO NOT submit one jointly-authored document. Each team member should write and submit their own individual report.)*. Submit your report in I-Learn Canvas.

# Scoring Standard (“rubric”)

|  |  |
| --- | --- |
| Report in Microsoft Word format, with team members’ names at the top | 1 point |
| Successfully log in to the Administrator account (Windows) and the root account (Linux) | 1 |
| Local user accounts created and configured with elevated privilege | 1 |
| VMs renamed and rebooted, with evidence of it in system event logs | 2 |
| PSRemoting enabled and working from your Windows VM to your Linux VM | 5 |
| PSRemoting enabled and working from your Linux VM to your Windows VM | 5 |
| PSRemoting enabled and working from the jump server to each of your team’s VMs | 5 |
| PSRemoting working from each of your team’s VMs to the jump server | 5 |
| TOTAL | 25 points |