Chapter 14.11 Lab: Installing and Updating Software Packages

Marko Shaffer

Information Technology, Franklin University

ITEC 200: Linux Fundamentals

Professor Kagan Ulucay

7/2/2023

**Red Hat System Administration I 8.2**

**Lab 10 CH 14.11 Installing and Updating Software Packages**

# ****Performance Checklist****

**In this lab, you will set up key-based authentication for users, and disable direct login as root and password authentication for all users for the OpenSSH service on one of your servers.**

# ****Outcomes****

**You should be able to:**

* **Authenticate using SSH keys.**
* **Prevent users from directly logging in as root over ssh.**
* **Prevent users from logging in to the system using SSH password-based authentication.**

# ****Log in to workstation as student using student as the password.****

|  |  |  |  |
| --- | --- | --- | --- |
|  | Franklin VM: | Standard User Account: | The Student's Root Account: |
| Username | kiosk | student | root |
| Password | redhat | student | redhat |

<https://franklin.instructure.com/courses/12488/modules/items/683350>

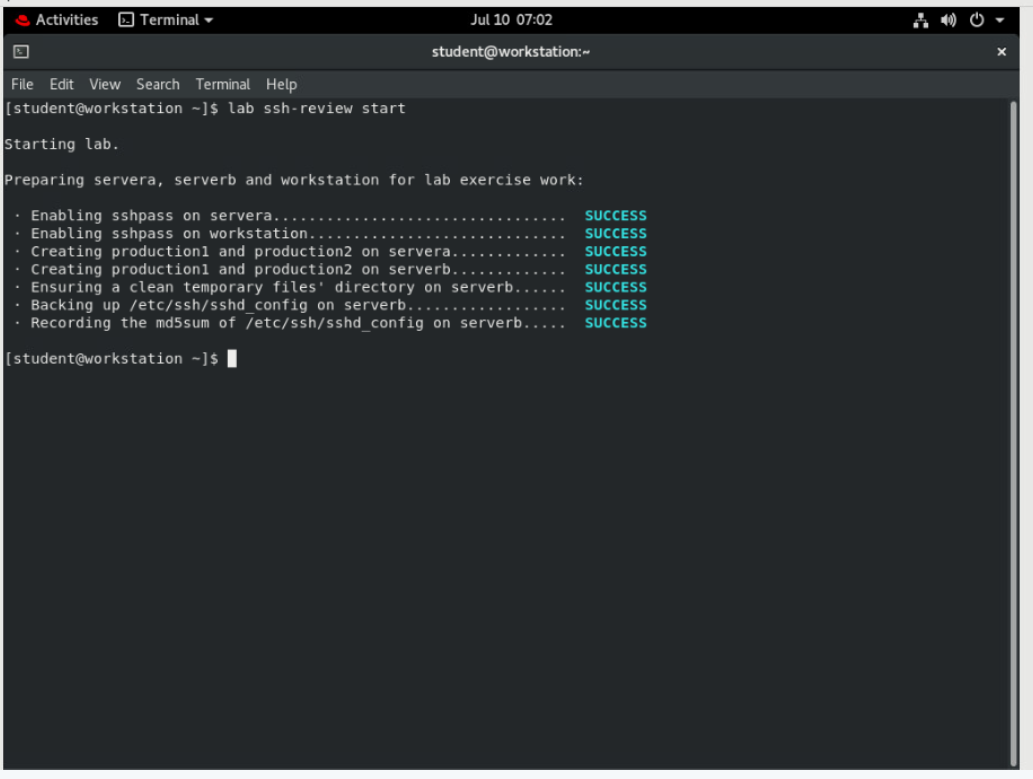
[kiosk@foundation0 ~]$ rht-vmctl start all

[kiosk@foundation0 ~]$ rht-vmview view workstation

# ****Start Lab****

On workstation, run lab ssh-review start to start the exercise. This script creates the necessary user accounts and files.

**[student@workstation ~]$ lab ssh-review start**

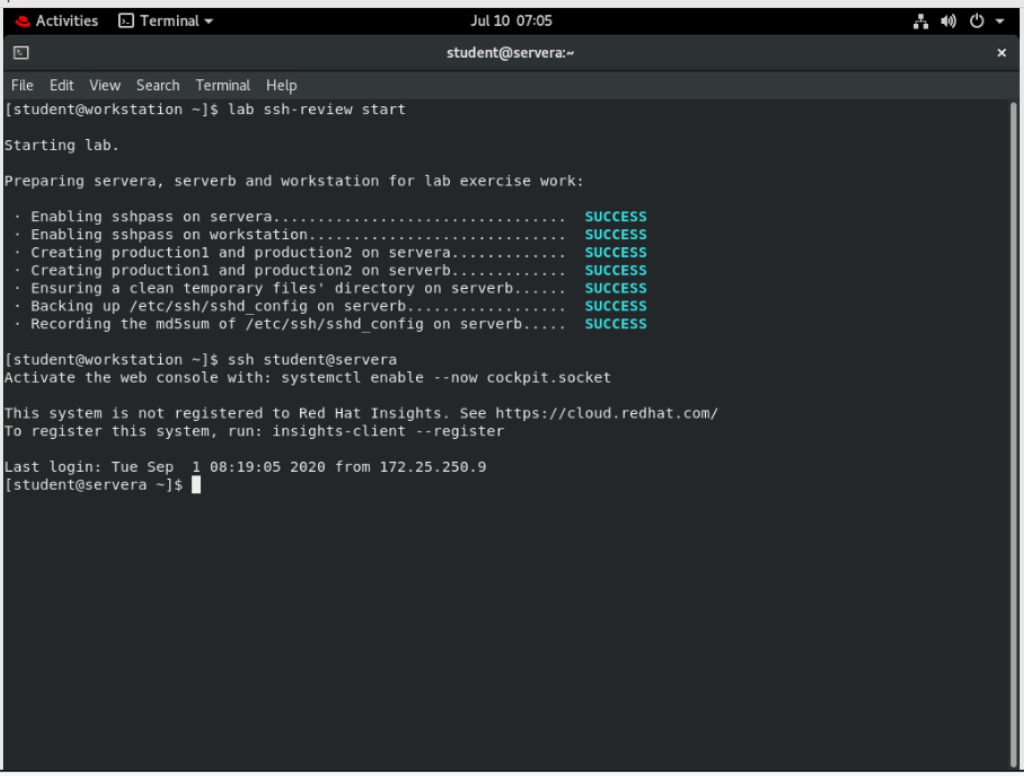


## From workstation, open an SSH session to servera as student.

**[student@workstation ~]$ ssh student@servera**

**...output omitted...**

**[student@servera ~]$**

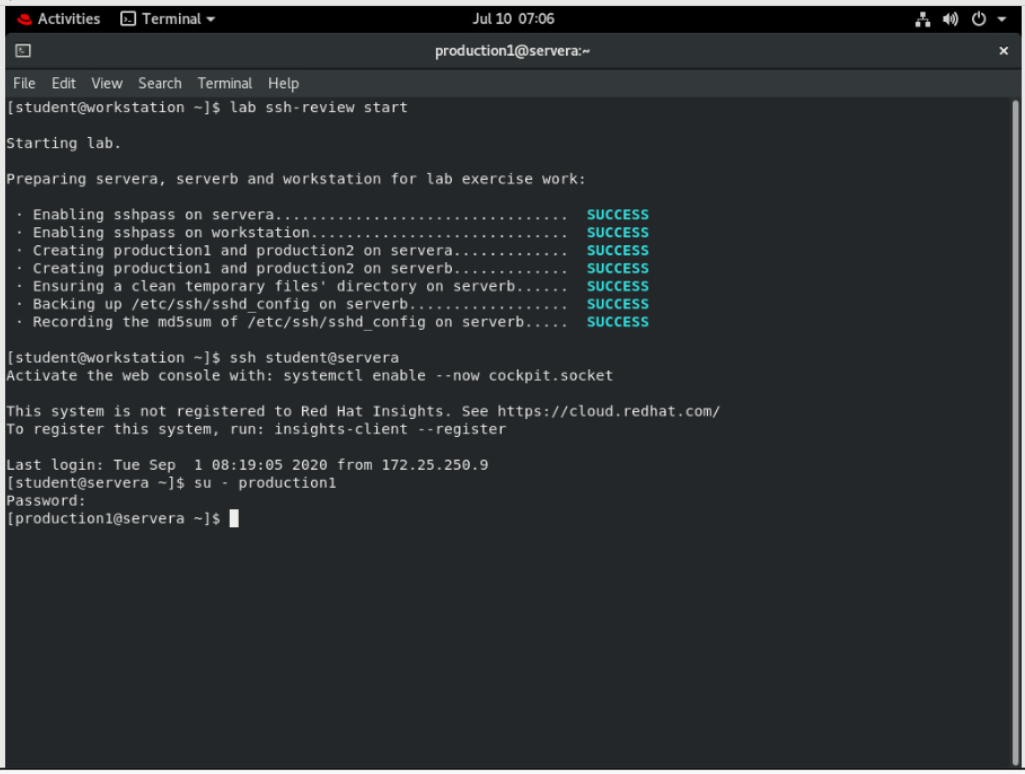


## Use the su command to switch to production1 on servera.

**[student@servera ~]$ su - production1**

**Password: redhat**

**[production1@servera ~]$**



## Use the ssh-keygen command to generate passphrase-less SSH keys for production1 on servera.

**[production1@servera ~]$ ssh-keygen**

**Generating public/private rsa key pair.**

**Enter file in which to save the key (/home/production1/.ssh/id\_rsa): Enter**

**Created directory '/home/production1/.ssh'.**

**Enter passphrase (empty for no passphrase): Enter**

**Enter same passphrase again: Enter**

**Your identification has been saved in /home/production1/.ssh/id\_rsa.**

**Your public key has been saved in /home/production1/.ssh/id\_rsa.pub.**

**The key fingerprint is:**

**SHA256:CsWCAmWOg5qaJujLzIAcengNj3u21kbrPP4Ysl3PXCA production1@servera.lab.example.com**

**The key's randomart image is:**

**+---[RSA 2048]----+**

**|..o |**

**|o+ . . |**

**|= o . o |**

**|.+ o |**

**|o.. . E . |**

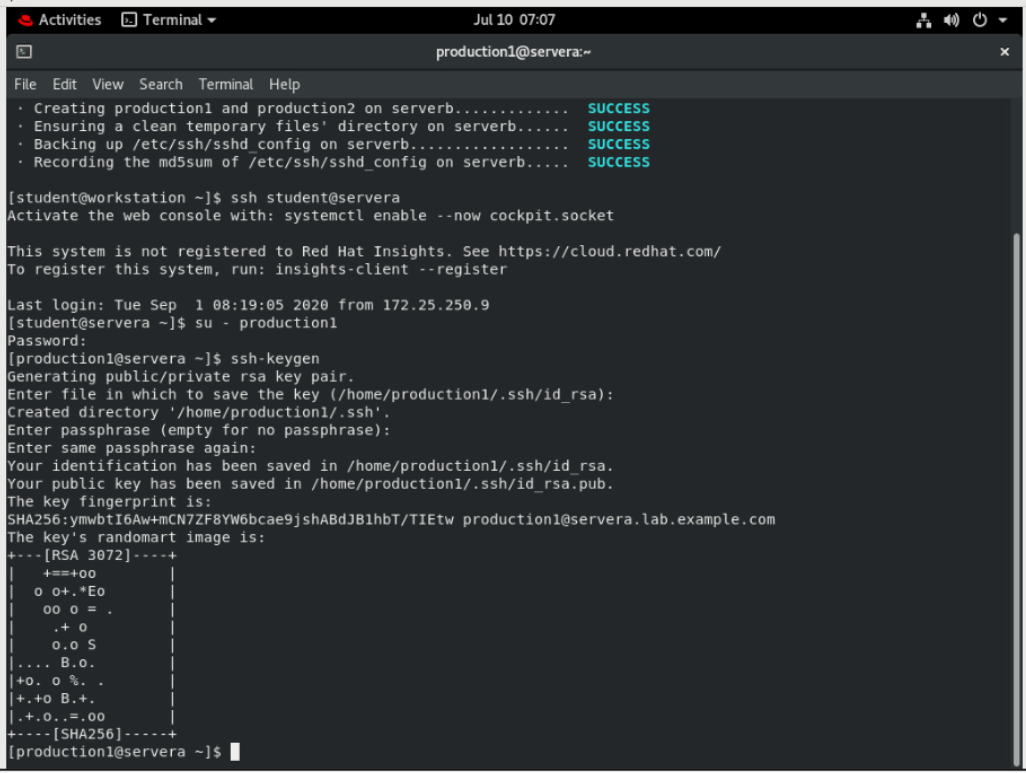
**|\*o.= ... . . |**

**|Xo+ +oo.. . |**

**|Oo .+==+ + . |**

**| \*o+o=\*o. + |**

**+----[SHA256]-----+**



## Use the ssh-copy-id command to send the public key of the SSH key pair to production1 on serverb.

**[production1@servera ~]$ ssh-copy-id production1@serverb**

**/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/production1/.ssh/id\_rsa.pub"**

**The authenticity of host 'serverb (172.25.250.11)' can't be established.**

**ECDSA key fingerprint is SHA256:ERTdjooOIrIwVSZQnqD5or+JbXfidg0udb3DXBuHWzA.**

**Are you sure you want to continue connecting (yes/no)? yes**

**/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed**

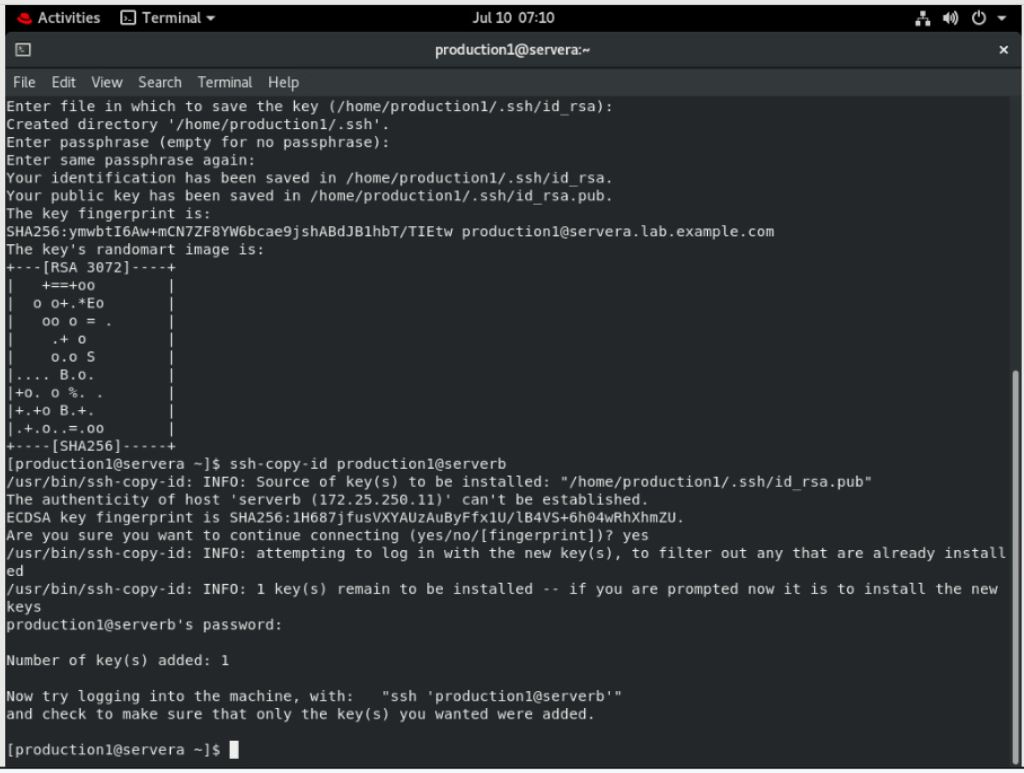
**/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys**

**production1@serverb's password: redhat**

**Number of key(s) added: 1**

**Now try logging into the machine, with: "ssh 'production1@serverb'"**

**and check to make sure that only the key(s) you wanted were added.**

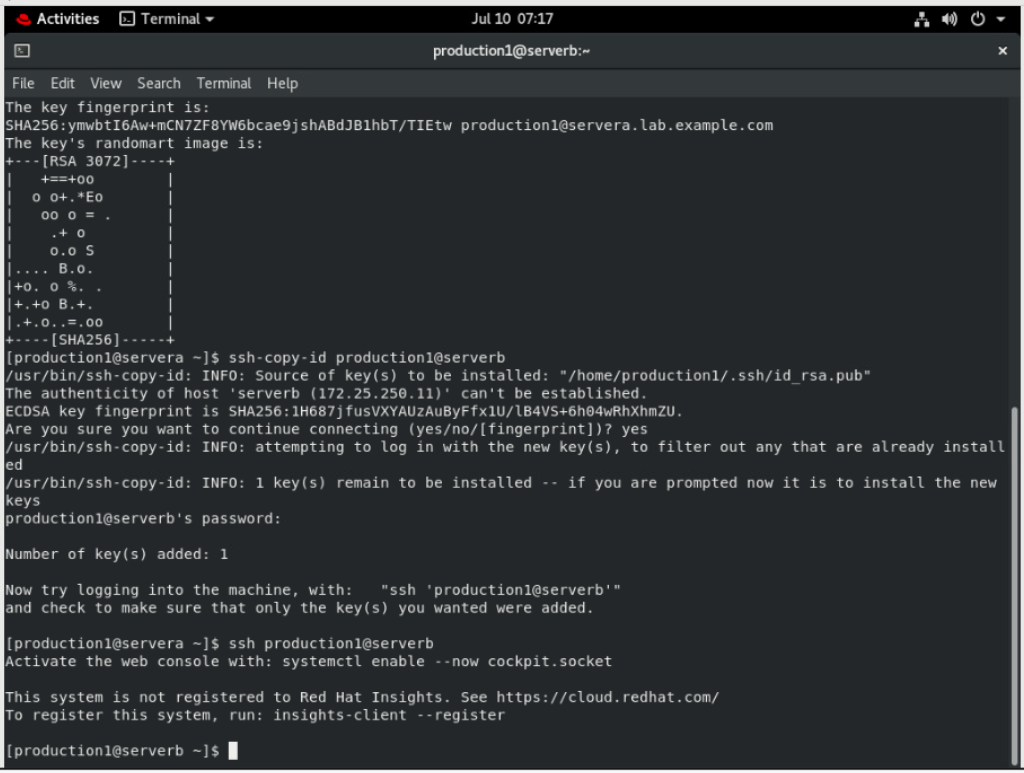


## Confirm that production1 can successfully log in to serverb using the SSH keys.

**[production1@servera ~]$ ssh production1@serverb**

**...output omitted...**

**[production1@serverb ~]$**



## Configure sshd on serverb to prevent users logging in as root. Use redhat as the password of the superuser.

* 1. Use the **su -** command to switch to root on serverb.

[production1@serverb ~]$ su -

Password: redhat

[root@serverb ~]#

## Set PermitRootLogin to no in /etc/ssh/sshd\_config and reload sshd. You may use vim /etc/ssh/sshd\_config to edit the configuration file of sshd.

**...output omitted...**

**PermitRootLogin no**

**...output omitted...**

**[root@serverb ~]# systemctl reload sshd.service**

## 

## Open another terminal on workstation and open an SSH session to servera as production1. From servera, try logging in to serverb as root. This should fail because you disabled root user login over SSH in the preceding step.

## Note

## For your convenience, the password-less login is already configured between workstation and servera in the classroom environment.

**[student@workstation ~]$ ssh production1@servera**

**...output omitted...**

**[production1@servera ~]$ ssh root@serverb**

**root@serverb's password: redhat**

**Permission denied, please try again.**

**root@serverb's password: redhat**

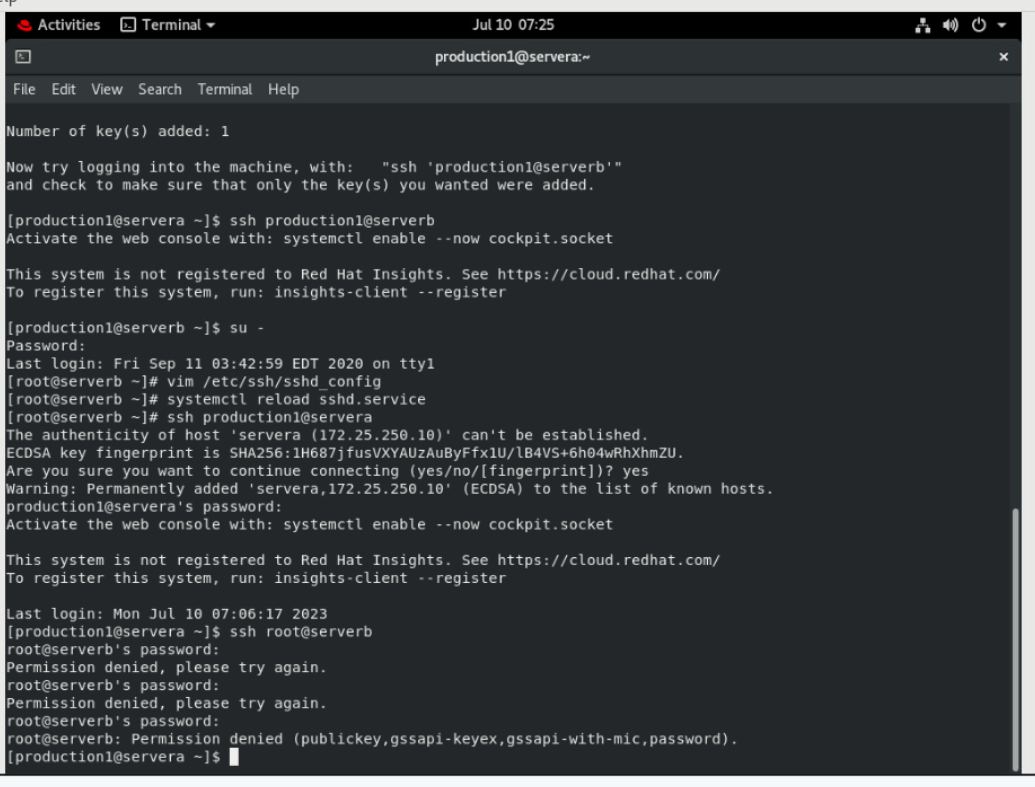
**Permission denied, please try again.**

**root@serverb's password: redhat**

**root@serverb: Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).**

**[production1@servera ~]$**

## The preceding ssh command returned after three failed attempts to log in to servera as root. By default, the ssh command prefers to use SSH keys for authentication but if it does not find the necessary keys of the user, it requests the user's password for authentication.



## Configure sshd on serverb to allow users to authenticate using SSH keys only, rather than their passwords.

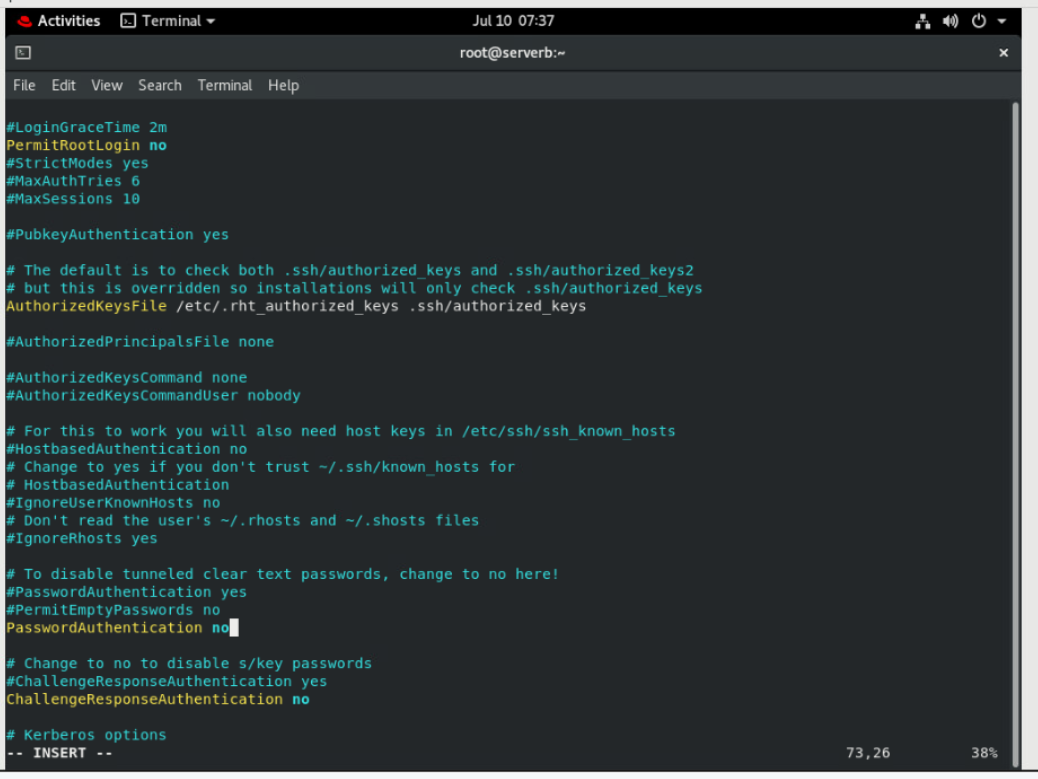
1. Return to the first terminal that has the root user's shell active on serverb. Set PasswordAuthentication to no in /etc/ssh/sshd\_config and reload sshd. You may use vim /etc/ssh/sshd\_config to edit the configuration file of sshd.

**...output omitted...**

**PasswordAuthentication no**

**...output omitted...**

**[root@serverb ~]# systemctl reload sshd**



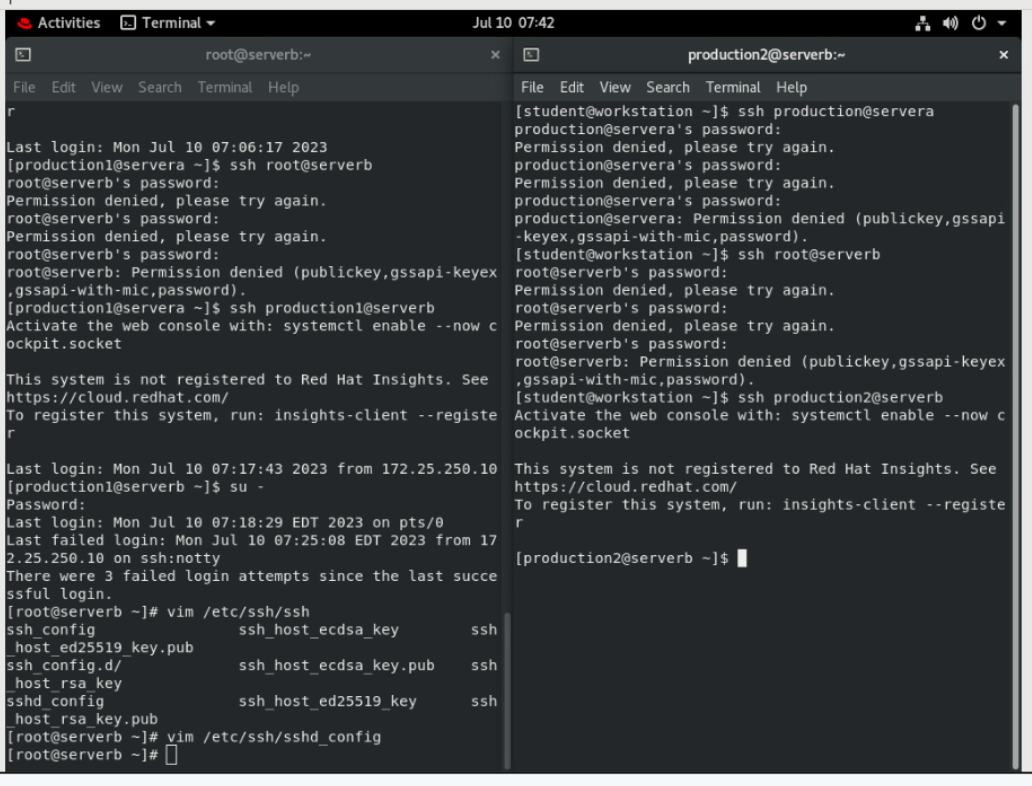
1. Go to the second terminal that has production1 user's shell active on servera and try logging in to serverb as production2. This should fail because SSH keys are not configured for production2, and the sshd service on serverb does not allow the use of passwords for authentication.

**[production1@servera ~]$ ssh production2@serverb**

**production2@serverb: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).**

## Note

## For more granularity, you may use the explicit -o PubkeyAuthentication=no and -o PasswordAuthentication=yes options with the ssh command. This allows you to override the ssh command's defaults and confidently establish whether the preceding command actually fails based on the settings you adjusted in /etc/ssh/sshd\_config in the preceding step.



1. Return to the first terminal that has the root user's shell active on serverb. Verify that PubkeyAuthentication is enabled in /etc/ssh/sshd\_config. You may use vim /etc/ssh/sshd\_config to view the configuration file of sshd.

**...output omitted...**

**#PubkeyAuthentication yes**

**...output omitted...**

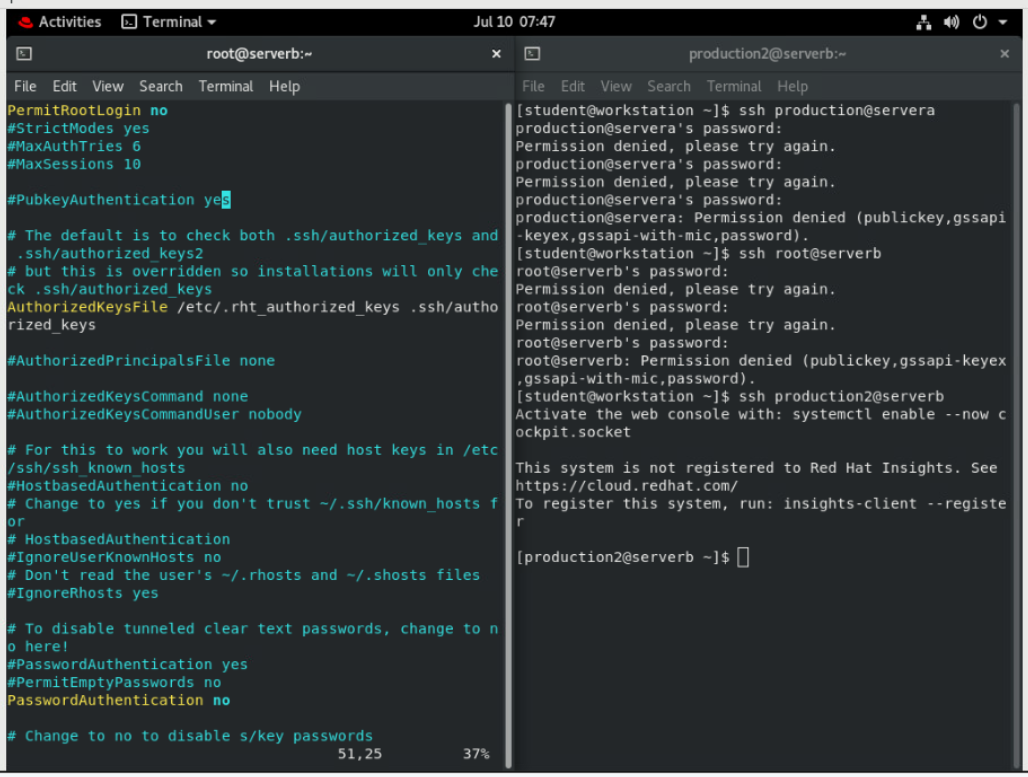
## Notice that the PubkeyAuthentication line is commented. Any commented line in this file uses the default value. Commented lines indicate the default values of a parameter. The public key authentication of SSH is active by default, as the commented line indicates.

1. Return to the second terminal that has production1 user's shell active on servera and try logging in to serverb as production1. This should succeed because SSH keys are configured for production1 to log in to serverb from servera.

**[production1@servera ~]$ ssh production1@serverb**

**...output omitted...**

**[production1@serverb ~]$**

]

1. From the second terminal, exit the production1 user's shell on both serverb and servera.

**[production1@serverb ~]$ exit**

**logout**

**Connection to serverb closed.**

**[production1@servera ~]$ exit**

**logout**

**[student@workstation ~]$**

1. Close the second terminal on workstation.

**[student@workstation ~]$ exit**

1. From the first terminal, exit the root user's shell on serverb.

**[root@serverb ~]# exit**

**logout**

1. From the first terminal, exit the production1 user's shell on both serverb and servera.

**[production1@serverb ~]$ exit**

**logout**

**Connection to serverb closed.**

**[production1@servera ~]$ exit**

**logout**

**[student@servera ~]$**

1. Log out of servera and return to the student user's shell on workstation .

**[student@servera ~]$ exit**

**logout**

**Connection to servera closed.**

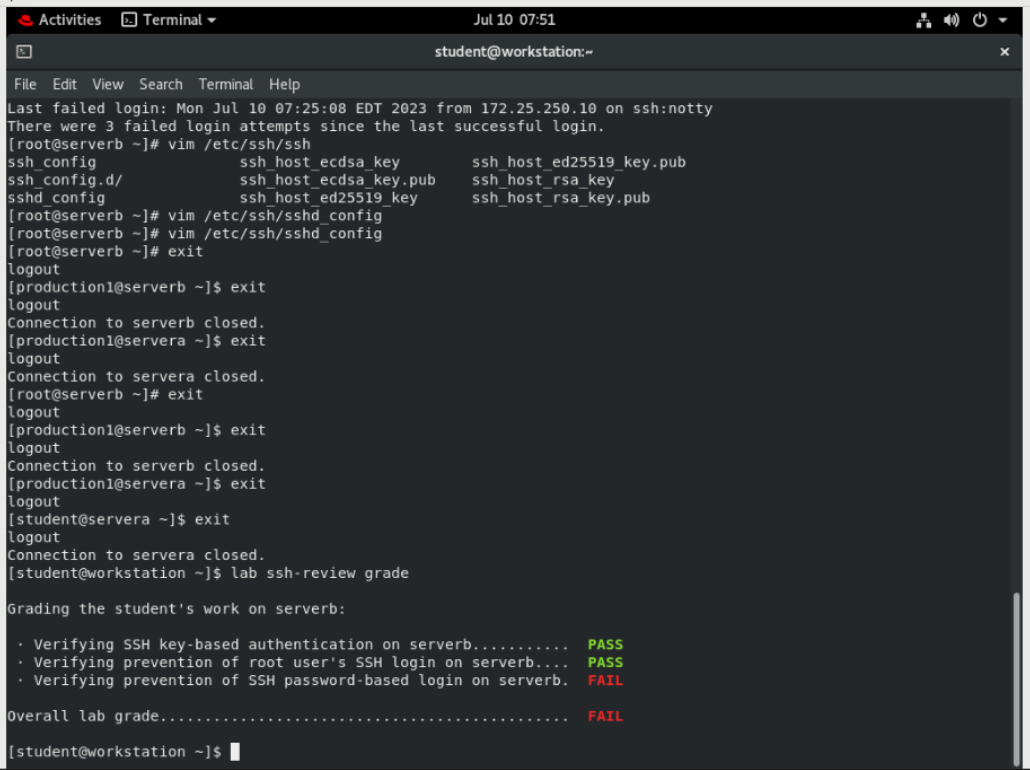
**[student@workstation ~]$**

## 

# Evaluation

On workstation, run the lab ssh-review grade command to confirm success of this exercise.

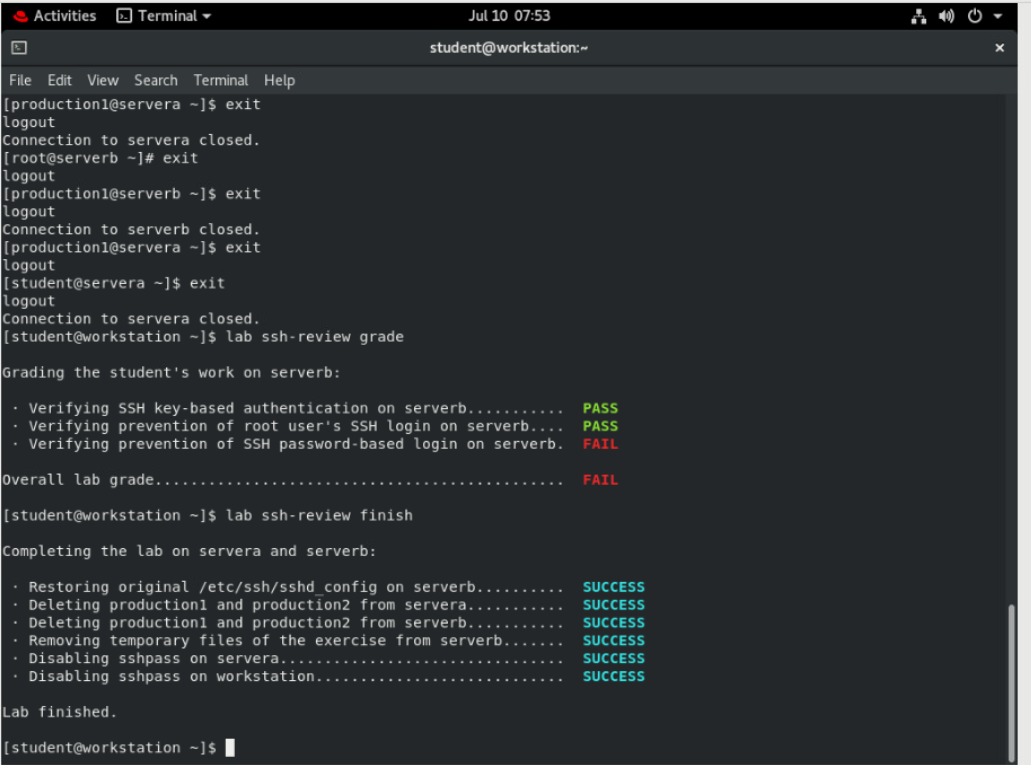
**[student@workstation ~]$ lab ssh-review grade**



# Finish

On workstation, run lab ssh-review finish to complete this lab.

**[student@workstation ~]$ lab ssh-review finish**



This concludes the lab.