Lab 239

Administrative processes

Student: Mane Zakarian
Bootcamp: Forge AWS re/Start UYMON5
Date: 2023

Objectives

In this lab, you will:

- Create a new log file for process listings
- Use the top command
- Establish a repetitive task that runs your previous auditing commands once a day

Accessing the AWS Management Console

1. At the top of these instructions, choose Start Lab to launch your lab. A **Start Lab** panel opens, and it displays the lab status.

   Tip: If you need more time to complete the lab, choose the Start Lab button again to restart the timer for the environment.

2. Wait until you see the message *Lab status: ready*, then close the **Start Lab** panel by choosing the **X**.
3. At the top of these instructions, choose AWS. This opens the AWS Management Console in a new browser tab. The system will automatically log you in.

   Tip: If a new browser tab does not open, a banner or icon is usually at the top of your browser with a message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon and then choose **Allow pop ups**.

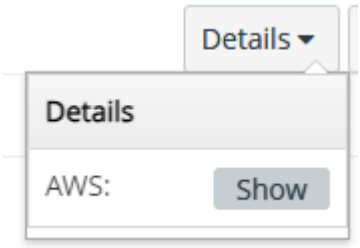
4. Arrange the AWS Management Console tab so that it displays alongside these instructions. Ideally, you will be able to see both browser tabs at the same time so that you can follow the lab steps more easily.

Task 1: Use SSH to connect to an Amazon Linux EC2 instance

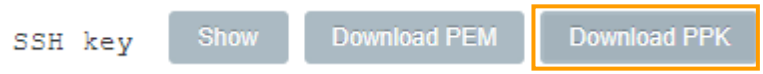
In this task, you will connect to a Amazon Linux EC2 instance. You will use an SSH utility to perform all of these operations.

Windows Users: Using SSH to Connect

1. Select the Details drop-down menu above these instructions you are currently reading, and then select Show. A Credentials window will be presented.



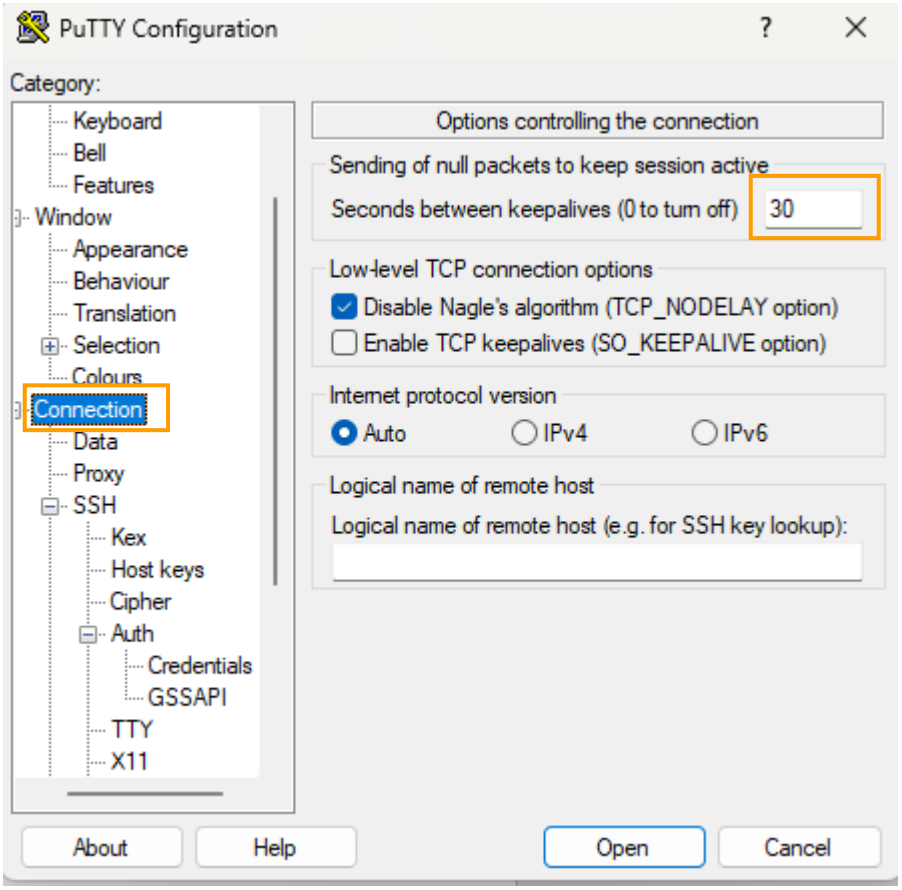
2. Select the **Download PPK** button and save the **labsuser.ppk** file.



3. Make a note of the **PublicIP** address.

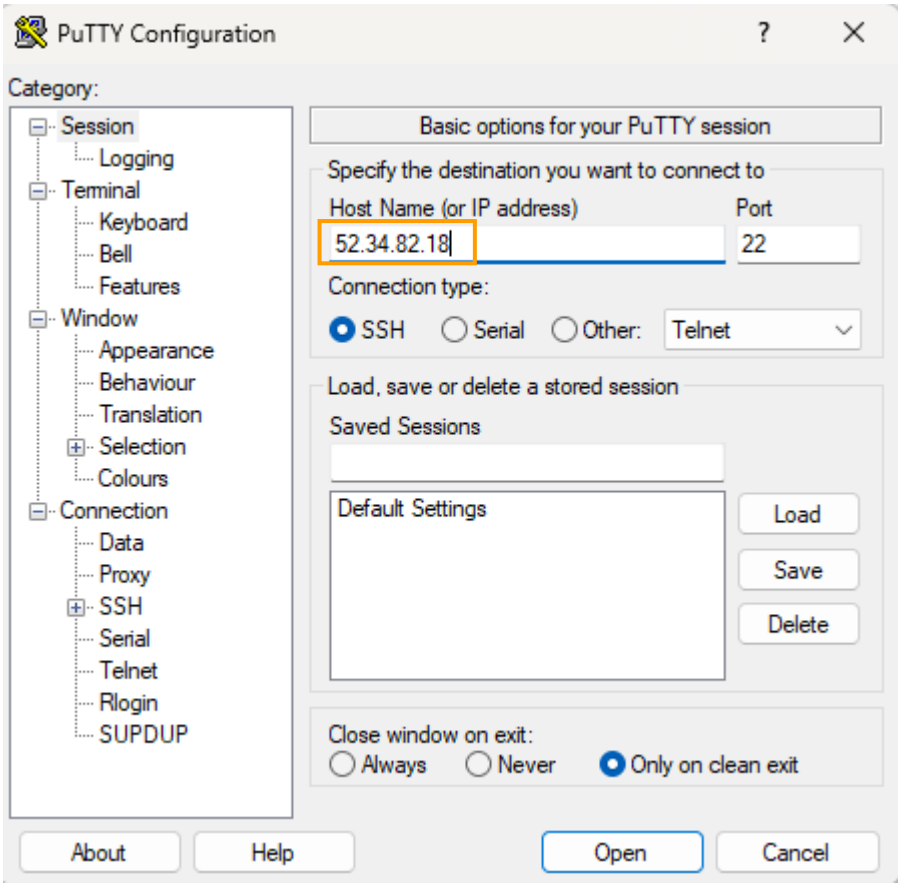
4. Then exit the Details panel by selecting the **X**.
5. Download **PuTTY** to SSH into the Amazon EC2 instance. If you do not have PuTTY installed on your computer.
6. Open **putty.exe**
7. Configure PuTTY timeout to keep the PuTTY session open for a longer period of time.:

- Select **Connection**
- Set **Seconds between keepalives** to `30`

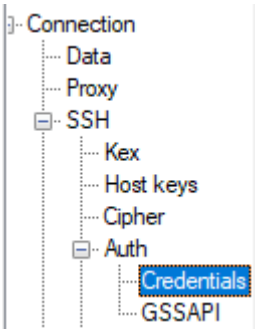


8. Configure your PuTTY session:
   - Select **Session**

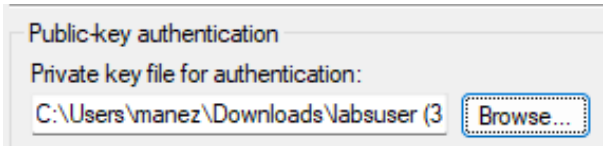
- **Host Name (or IP address):** Paste the **Public DNS or IPv4 address** of the instance you made a note of earlier. Alternatively, return to the EC2 Console and select **Instances**. Check the box next to the instance you want to connect to and in the *Description* tab copy the **IPv4 Public IP** value



- Back in PuTTY, in the **Connection** list, expand **SSH** and select **Auth** *(don't expand it)*

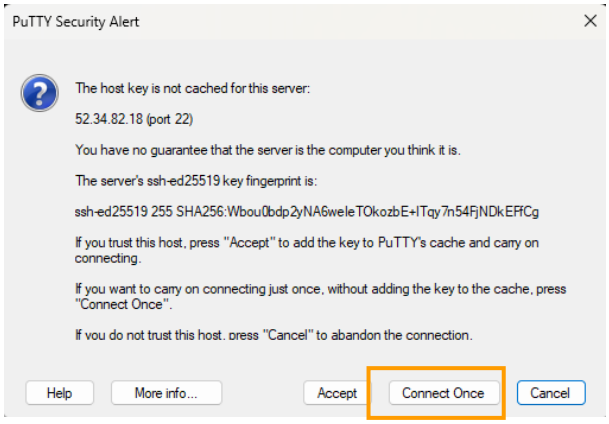


- Select **Browse** and select the lab#.ppk file that you downloaded

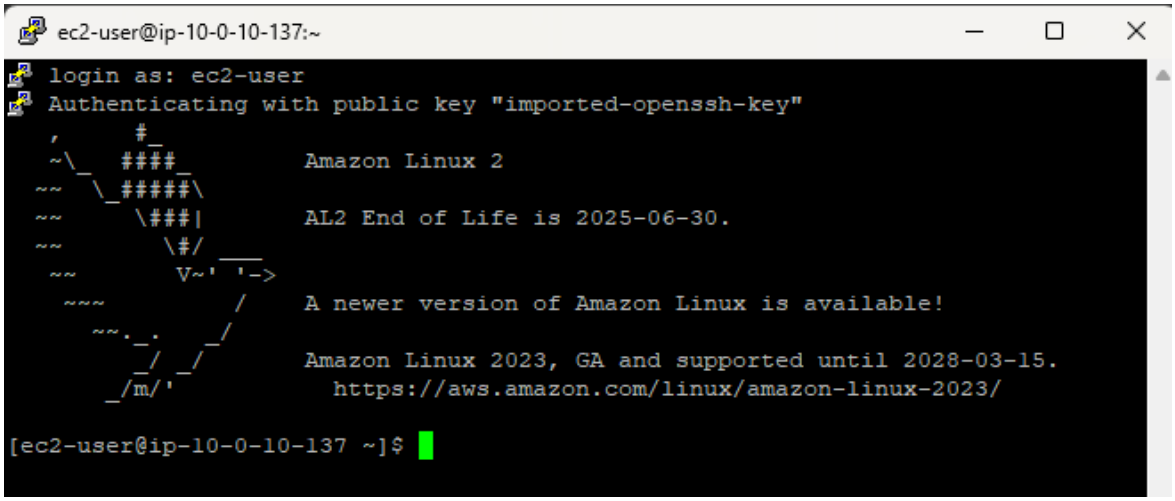


- Select **Open** to select it and then select **Open** again.

9. Select **Yes**, to trust and connect to the host.



10. When prompted **login as**, enter: `ec2-user` This will connect you to the EC2 instance.



Task 2: Exercise - Create List of Processes

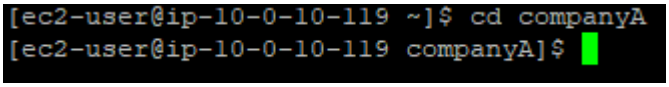
In this exercise, you will create a log file from the ps command. This log file should be added to the SharedFolders section:

Create a log file named processes.csv from ps -aux and omit any processes that contain root user or contain "["or"]" in the COMMAND section.

Note: There is a space following the command followed by a period to represent the current location.

24. To validate that you are in the **/home/ec2-user/companyA** folder, enter `pwd` and press Enter.

    If you are not in this folder, enter `cd companyA` and press Enter.



25. View all processes running on the machine and filter out the word root by typing `sudo ps -aux | grep -v root | sudo tee SharedFolders/processes.csv` and pressing ENTER.

```
[ec2-user@ip-10-0-10-119 companyA]$ sudo ps -aux | grep -v root | sudo tee Share
dFolders/processes.csv
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
rpc        1699  0.0  0.3  67256  3332 ?        Ss   22:37   0:00 /sbin/rpcbind -
w
dbus       1706  0.0  0.4  58248  4028 ?        Ss   22:37   0:00 /usr/bin/dbus-d
aemon --system --address=systemd: --nofork --nopidfile --systemd-activation
chrony     1711  0.0  0.3 120344  3236 ?        S    22:37   0:00 /usr/sbin/chron
yd -F 2
libstor+   1721  0.0  0.1  12628  1856 ?        Ss   22:37   0:00 /usr/bin/lsmd -
d
rngd       1726  0.0  0.4  94212  4484 ?        Ss   22:37   0:00 /sbin/rngd -f -
-fill-watermark=0 --exclude=jitter
postfix    2135  0.0  0.6  90388  6692 ?        S    22:37   0:00 pickup -l -t un
ix -u
postfix    2136  0.0  0.6  90464  6712 ?        S    22:37   0:00 qmgr -l -t unix
 -u
ec2-user   3180  0.0  0.4 148504  4608 ?        S    22:38   0:00 sshd: ec2-user@
pts/0
ec2-user   3181  0.0  0.4 124736  3992 pts/0    Ss   22:38   0:00 -bash
[ec2-user@ip-10-0-10-119 companyA]$
```

26. Validate your work by typing `cat SharedFolders/processes.csv` and pressing ENTER.

Figure: The command sudo ps -aux | grep -v root | sudo tee SharedFolders/processes.csv shows all the current proccesses running on your machine. This is also validated by using the command cat SharedFolders/processes.csv.

```
[ec2-user@ip-10-0-10-119 companyA]$ cat SharedFolders/processes.csv
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
rpc        1699  0.0  0.3  67256  3332 ?        Ss   22:37   0:00 /sbin/rpcbind -
w
dbus       1706  0.0  0.4  58248  4028 ?        Ss   22:37   0:00 /usr/bin/dbus-d
aemon --system --address=systemd: --nofork --nopidfile --systemd-activation
chrony     1711  0.0  0.3 120344  3236 ?        S    22:37   0:00 /usr/sbin/chron
yd -F 2
libstor+   1721  0.0  0.1  12628  1856 ?        Ss   22:37   0:00 /usr/bin/lsmd -
d
rngd       1726  0.0  0.4  94212  4484 ?        Ss   22:37   0:00 /sbin/rngd -f -
-fill-watermark=0 --exclude=jitter
postfix    2135  0.0  0.6  90388  6692 ?        S    22:37   0:00 pickup -l -t un
ix -u
postfix    2136  0.0  0.6  90464  6712 ?        S    22:37   0:00 qmgr -l -t unix
 -u
ec2-user   3180  0.0  0.4 148504  4608 ?        S    22:38   0:00 sshd: ec2-user@
pts/0
ec2-user   3181  0.0  0.4 124736  3992 pts/0    Ss   22:38   0:00 -bash
[ec2-user@ip-10-0-10-119 companyA]$
```

c

Task 3: Exercise - List the processes using the top command

In this exercise, you will use the top command:

- Run the **top** command to display processes and threads that are active in the system.
- Observe the outputs of the top command.

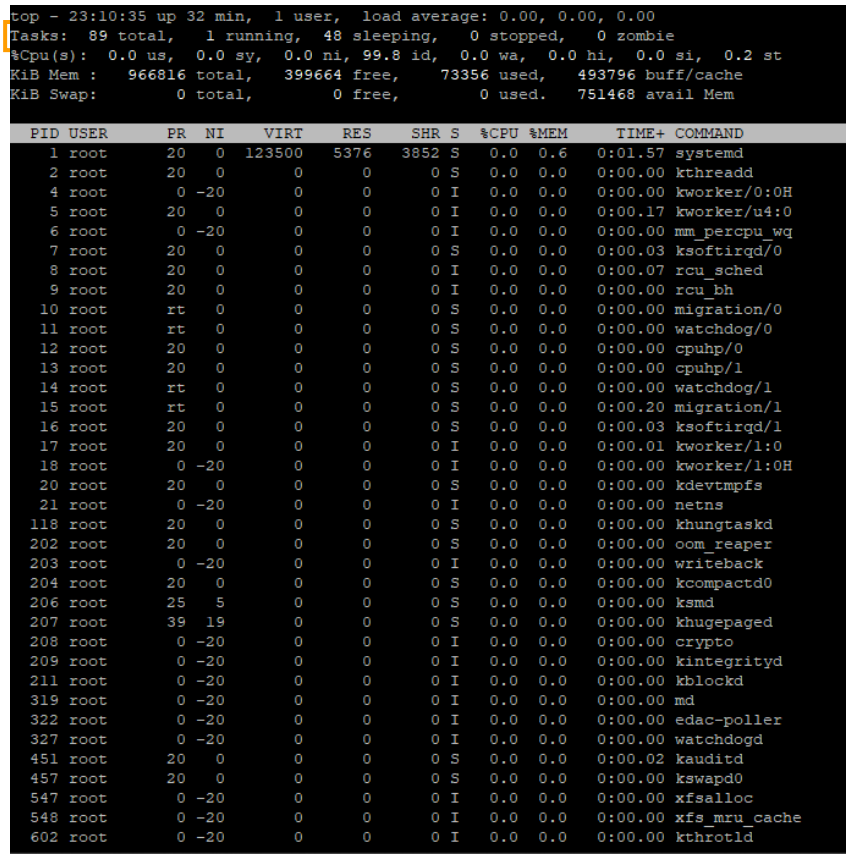
27. In the main terminal run the command top and press ENTER:

```
top
```

The top command is used to display the system performance and lists the processes and threads active in the system. The output of the top command should look similar to the picture below:

Figure: The output of the top command gives the system performance and gives you the following information: Total number of tasks, how many are running, how many are sleeping, how many are stopped, zombie state. It gives the percentage of CPU used, the KiB memory used, and KiB swap.

28. While observing the output of top, the second line below the command top, we can see the Tasks (outlined in orange). Tasks in top either have a running, sleep, stopped or zombie state. How many running tasks do you see?



```
top - 23:10:35 up 32 min,  1 user,  load average: 0.00, 0.00, 0.00
Tasks:  89 total,   1 running,  48 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.2 st
KiB Mem :   966816 total,   399664 free,    73356 used,   493796 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   751468 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    1 root      20   0  123500   5376   3852 S   0.0  0.6   0:01.57 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H
    5 root      20   0       0      0      0 I   0.0  0.0   0:00.17 kworker/u4:0
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 mm_percpu_wq
    7 root      20   0       0      0      0 S   0.0  0.0   0:00.03 ksoftirqd/0
    8 root      20   0       0      0      0 I   0.0  0.0   0:00.07 rcu_sched
    9 root      20   0       0      0      0 I   0.0  0.0   0:00.00 rcu_bh
   10 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 migration/0
   11 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 watchdog/0
   12 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/0
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/1
   14 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 watchdog/1
   15 root      rt   0       0      0      0 S   0.0  0.0   0:00.20 migration/1
   16 root      20   0       0      0      0 S   0.0  0.0   0:00.03 ksoftirqd/1
   17 root      20   0       0      0      0 I   0.0  0.0   0:00.01 kworker/1:0
   18 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/1:0H
   20 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kdevtmpfs
   21 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 netns
  118 root      20   0       0      0      0 S   0.0  0.0   0:00.00 khungtaskd
  202 root      20   0       0      0      0 S   0.0  0.0   0:00.00 oom_reaper
  203 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 writeback
  204 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kcompactd0
  206 root      25   5       0      0      0 S   0.0  0.0   0:00.00 ksmd
  207 root      39  19       0      0      0 S   0.0  0.0   0:00.00 khugepaged
  208 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 crypto
  209 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kintegrityd
  211 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kblockd
  319 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 md
  322 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 edac-poller
  327 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 watchdogd
  451 root      20   0       0      0      0 S   0.0  0.0   0:00.02 kauditd
  457 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kswapd0
  547 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 xfsalloc
  548 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 xfs_mru_cache
  602 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kthrotld
```

29. To quit top, hit **q** and press ENTER.
30. You can also run top with the following options to find the usage and version information:

```
top -hv
```

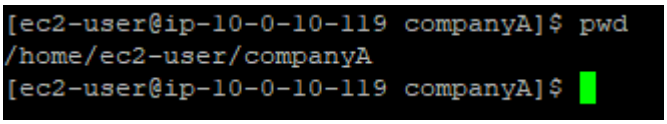
Task 4: Exercise - Create a Cron Job

In this exercise, you will create a cron job that will create an audit file with ##### to cover all csv files:

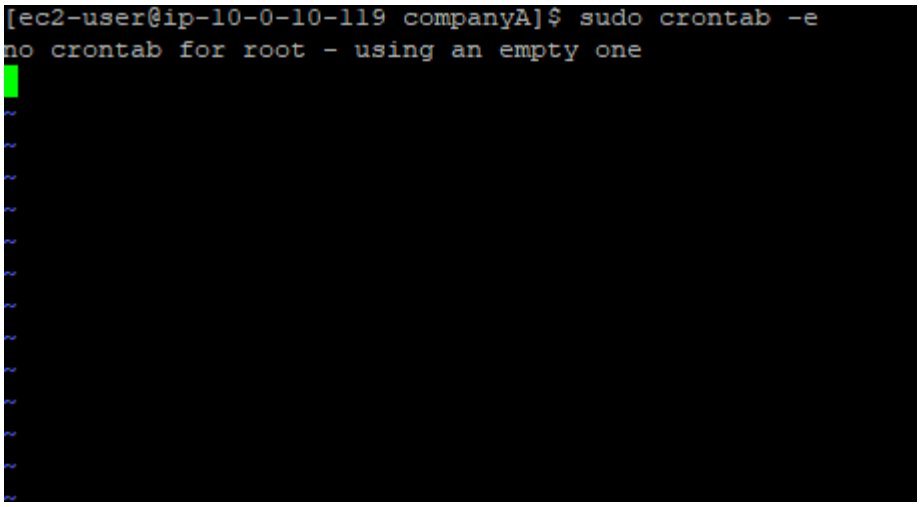
Note: You may have to use sudo to complete this exercise if you are not root.

Remember that **cron** is a command that runs a task on a regular basis at a specified time. This command maintains the list of tasks to run in a crontab file, which you create in this task. You create a job that creates the audit file with ##### in order to cover all .csv files. When you enter the **crontab -e** command, you are taken to an editor where you then enter a list of steps of what the cron daemon will run. The crontab file includes six fields: minutes, hour, day of month (DOM), month (MON), day of Week (DOW), and command (CMD). These fields can also be denoted with asterisks. Once this command runs, you can verify your work.

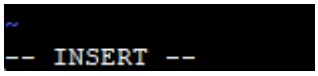
31. To validate that you are in the **/home/ec2-user/companyA** folder, enter `pwd` and press Enter.



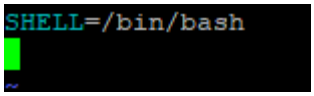
32. To create a cron job that creates the audit file with ##### to cover all .csv files, enter `sudo crontab -e` and press Enter to enter the default text editor.



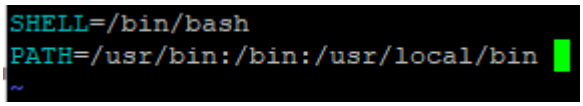
33. Press `i` to enter insert mode, and press Enter.



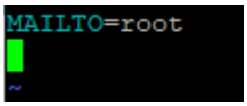
34. For the first line, enter and press the Space bar.



35. For the second line, eter `PATH=/usr/bin:/bin:/usr/local/bin`

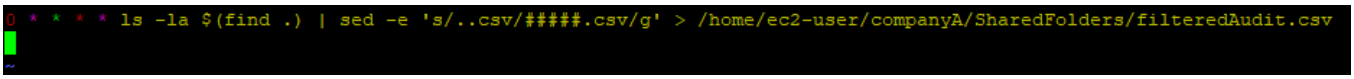


36. and press Enter.
37. For the third line, enter `MAILTO=root` and press Enter.



38. For the last line, enter `0 * * * * ls -la $(find .) | sed -e 's/..csv/#####.csv/g' > /home/ec2-user/companyA/SharedFolders/filteredAudit.csv`

Figure: In the terminal, it shows how the cron job with the SHELL, PATH, MAILTO, and a script that was referenced earlier in the lab.



38. To save and close the file, press ESC. Then enter `:wq` and press Enter.

39. To validate your work, enter `sudo crontab -l` and press Enter. Inspect the crontab file to ensure that it matches the text exactly, as the following output shows:

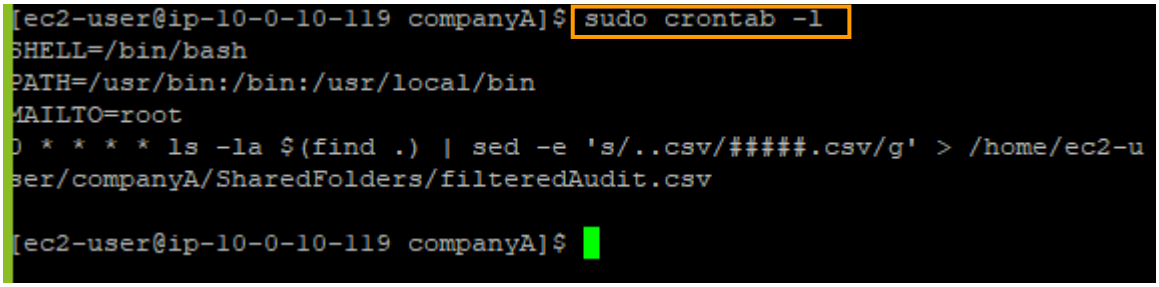


Figure: A validated cron job is shown by entering the command sudo crontab -l. The output of the command will be from the file that was entered from earlier in the lab.

Commands Used:

On this lab we used several commands to perform different tasks. Here is a summary of the commands used:

Command	Description
ls	List the contents of a directory.
pwd	Print the current working directory.
cd	Change the current working directory.
mkdir	Create a new directory.
touch	Create a new file.
cp	Copy files or directories.
mv	Move or rename files or directories.
rm	Remove files or directories.
cat	Display the contents of a file.
head	Display the first few lines of a file.
tail	Display the last few lines of a file.
grep	Search for a specific pattern in a file.
chmod	Change the permissions of a file or directory.
chown	Change the owner of a file or directory.
sudo	Execute commands with superuser privileges.