Debugging and Solving Software Problems

1 hour 30 minutesFree

Introduction

You're a member of your company's IT department. A colleague that recently left the company wrote a program that's 90% complete; it's designed to read some data files with information on employees and then generate a report. It's up to you to finish the code -- this includes fixing any errors, bugs, and slowness that might be in the unfinished code.

Prerequisites:

You should have a sound knowledge of the following things prior to performing the lab:

•   Debugging (gathering information, root cause analysis, and remediation)

•   Identifying and understanding system performance (I/O, Network, CPU, Memory)

•   Understanding and troubleshooting the environment around the program (file system, OS, etc.)
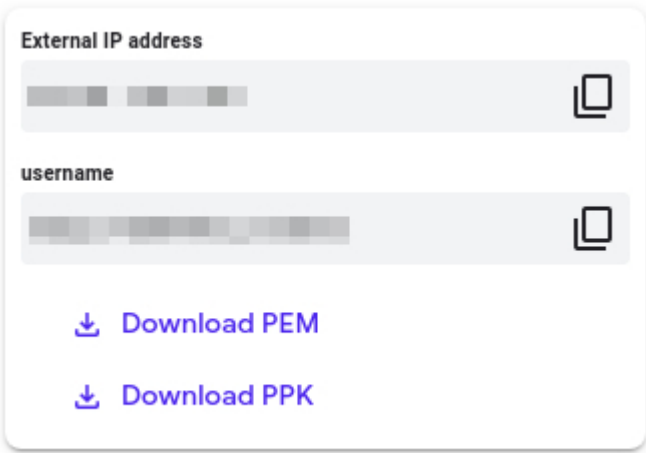
You'll have 90 minutes to complete this lab.

Start the lab

You'll need to start the lab before you can access the materials in the virtual machine OS. To do this, click the green "Start Lab" button at the top of the screen.

Note: For this lab you are going to access the **Linux VM** through your **local SSH Client**, and not use the **Google Console** (**Open GCP Console** button is not available for this lab).

After you click the "Start Lab" button, you will see all the SSH connection details on the left-hand side of your screen. You should have a screen that looks like this:

Accessing the virtual machine

Please find one of the three relevant options below based on your device's operating system.

Note: Working with Qwiklabs may be similar to the work you'd perform as an **IT Support Specialist**; you'll be interfacing with a cutting-edge technology that requires multiple steps to

access, and perhaps healthy doses of patience and persistence(!). You'll also be using **SSH** to enter the labs -- a critical skill in IT Support that you'll be able to practice through the labs.

Option 1: Windows Users: Connecting to your VM

In this section, you will use the PuTTY Secure Shell (SSH) client and your VM's External IP address to connect.

Download your PPK key file

You can download the VM's private key file in the PuTTY-compatible **PPK** format from the Qwiklabs Start Lab page. Click on **Download PPK**.
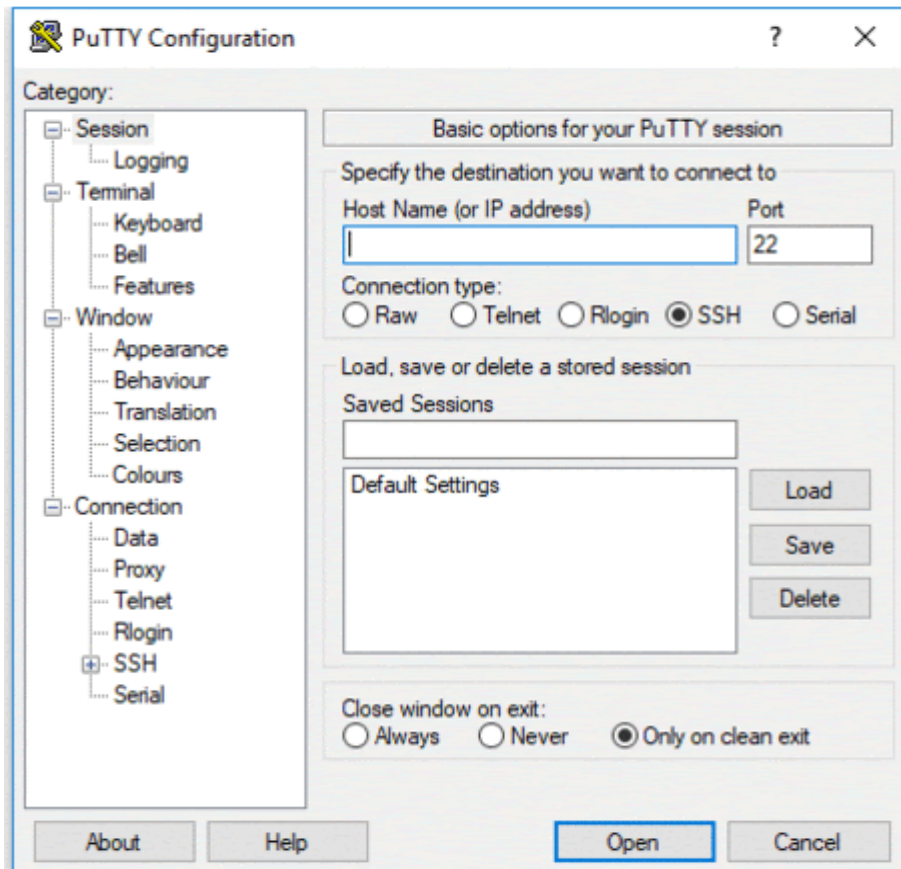
⬇ Download PEM

⬇ Download PPK ⬅

Connect to your VM using SSH and PuTTY

1. You can download Putty from here

2. In the **Host Name (or IP address)** box, enter username@external_ip_address.

Note: Replace **username** and **external_ip_address** with values provided in the lab.

3.   In the **Connection** list, expand **SSH**.

4.   Then expand **Auth** by clicking on + icon.

5.   Now, select the **Credentials** from the **Auth** list.

6.   In the **Private key file for authentication** box, browse to the PPK file that you downloaded and double-click it.

7.   Click on the **Open** button.

Note: PPK file is to be imported into PuTTY tool using the Browse option available in it. It should not be opened directly but only to be used in PuTTY.
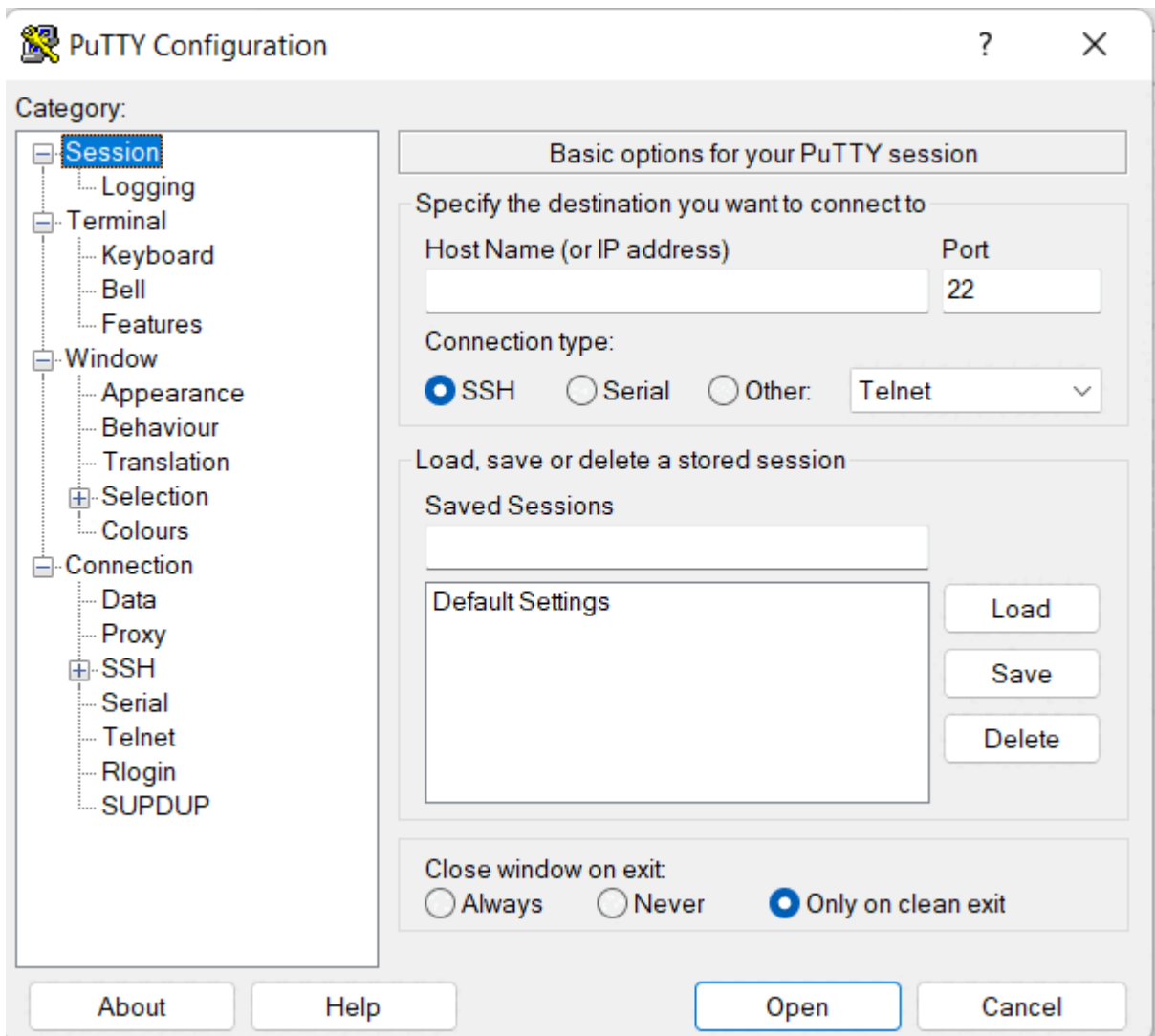
8.   Click **Yes** when prompted to allow a first connection to this remote SSH server. Because you are using a key pair for authentication, you will not be prompted for a password.

Common issues

If PuTTY fails to connect to your Linux VM, verify that:

*   You entered **<username>@<external ip address>** in PuTTY.

*   You downloaded the fresh new PPK file for this lab from Qwiklabs.

*   You are using the downloaded PPK file in PuTTY.

Option 2: OSX and Linux users: Connecting to your VM via SSH

Download your VM's private key file.

You can download the private key file in PEM format from the Qwiklabs Start Lab page. Click on **Download PEM**.

Connect to the VM using the local Terminal application

A **terminal** is a program which provides a **text-based interface for typing commands**. Here you will use your terminal as an SSH client to connect with lab provided Linux VM.

1.   Open the Terminal application.

•   To open the terminal in Linux use the shortcut key **Ctrl+Alt+t**.

•   To open terminal in **Mac** (OSX) enter **cmd** + **space** and search for **terminal**.

2.   Enter the following commands.

Note: Substitute the **path/filename for the PEM** file you downloaded, **username** and **External IP Address**.

You will most likely find the PEM file in **Downloads**. If you have not changed the download settings of your system, then the path of the PEM key will be **~/Downloads/qwikLABS-XXXXX.pem**

chmod 600 ~/Downloads/qwikLABS-XXXXX.pem
Copied!
content_copy

ssh -i ~/Downloads/qwikLABS-XXXXX.pem username@External Ip Address
Copied!

content_copy

Option 3: Chrome OS users: Connecting to your VM via SSH

Note: Make sure you are not in **Incognito/Private mode** while launching the application.

Download your VM's private key file.

You can download the private key file in PEM format from the Qwiklabs Start Lab page. Click on **Download PEM**.

Connect to your VM

1.   Add Secure Shell from here to your Chrome browser.

2.   Open the Secure Shell app and click on **[New Connection]**.

[New Connection]

username@hostname or free form

username   hostname

SSH relay server options

Identity: [default]

SSH Arguments: extra command l

Current profile: default

Mount Path: the default path

[DEL] Delete   Options

3.	In the **username** section, enter the username given in the Connection Details Panel of the lab. And for the **hostname** section, enter the external IP of your VM instance that is mentioned in the Connection Details Panel of the lab.

[New Connection]

username@hostname or free form text

username   hostname

SSH relay server options

Identity: [default]

SSH Arguments: extra command line arg

Current profile: default

Mount Path: the default path is th

[DEL] Delete   Options

4.   In the **Identity** section, import the downloaded PEM key by clicking on

the **Import…** button beside the field. Choose your PEM key and click on the **OPEN** button.

Note: If the key is still not available after importing it, refresh the application, and select it from the **Identity** drop-down menu.

5.   Once your key is uploaded, click on the **[ENTER] Connect** button below.

[New Connection]

username@hostname or free form text

username | hostname

SSH relay server options

Identity: [default]

SSH Arguments: extra command line argu

Current profile: default

Mount Path: the default path is the

[DEL] Delete | Options

6.   For any prompts, type **yes** to continue.

7.   You have now successfully connected to your Linux VM.

You're now ready to continue with the lab!

Debug issue

You have a start_date_report.py Python script with a bunch of functions
like **get_start_date()**, **list_newer()** and others. This script will operate on the data file employees-
with-date.csv, which is generated from a file URI within the script. The script then generates a report
of all employees that started on the given start date.

To list the files on the home directory, use the following command:

ls
Copied!
content_copy

Output:

```
student-04-a8a259bc22ec@linux-instance:~$ ls
start_date_report.py
```

Grant the executable and editable file permission to the start_date_report.py

sudo chmod 777 ~/start_date_report.py
Copied!
content_copy

Now, run the python program start_date_report.py

./start_date_report.py
Copied!
content_copy

Enter the values for the year, month, and day respectively as the prompt appears.

Output:

```
student-04-a8a259bc22ec@linux-instance:~$ ./start_date_report.py

Getting the first start date to query for.

The date must be greater than Jan 1st, 2018
Enter a value for the year: 2019
Enter a value for the month: 1
Enter a value for the day: 1

Traceback (most recent call last):
  File "./start_date_report.py", line 83, in <module>
    main()
  File "./start_date_report.py", line 79, in main
    start_date = get_start_date()
  File "./start_date_report.py", line 23, in get_start_date
    return datetime.datetime(year, month, day)
TypeError: an integer is required (got type str)
```

The program crashes with a **TypeError**. This is because it reads the value entered at prompts as a string. Refer to the function datetime.datetime() within the script. The arguments passed to the datetime.datetime() function should be of integer type, but in our case, the input values are strings.

In order to fix this **ERROR**, open start_date_report.py by using the following command:

nano ~/start_date_report.py
Copied!
content_copy

Now, search for **get_start_date()** function and typecast the string variable that's taken from user input to the integer. Here, we have to explicitly cast the data type of these three variables: year, month, and day from string to integer.

Eg. year = int(input('Enter a value for the year: '))

Similarly, you can cast the values of month and day to an integer.

The **get_start_date()** function should now looks like this:

```
def get_start_date():
    """Interactively get the start date to query for."""

    print()
    print('Getting the first start date to query for.')
    print()
    print('The date must be greater than Jan 1st, 2018')
    year = int(input('Enter a value for the year: '))
    month = int(input('Enter a value for the month: '))
    day = int(input('Enter a value for the day: '))
    print()

    return datetime.datetime(year, month, day)
```

Save the start_date_report.py script file by clicking Ctrl-o, the Enter key, and Ctrl-x.

Run the start_date_report.py Python script:

./start_date_report.py
Copied!
content_copy

Output:

```
student-04-a8a259bc22ec@linux-instance:~$ ./start_date_report.py

Getting the first start date to query for.

The date must be greater than Jan 1st, 2018
Enter a value for the year: 2019
Enter a value for the month: 10
Enter a value for the day: 4

Started on Oct 04, 2019: ['Quamar Stewart']
Started on Oct 06, 2019: ['Darius Goodman']
Started on Oct 15, 2019: ['Idola Warren']
Started on Oct 16, 2019: ['Hu Hyde', 'Lesley Fuentes']
```

Click Check my progress to verify the objective.

Debug and fix issue

Check my progress

Improve performance

Once you debug the issue, the program will start processing the file but it takes a long time to complete. This is because the program goes slowly line by line instead of printing the report quickly. You need to debug why the program is slow and then fix it. In this section, you need to find bottlenecks, improve the code, and make it finish faster.

The problem with the script is that it's downloading the whole file and then going over it for each date. The current script takes almost 2 minutes to complete for 2019-01-01. An optimized script should generate reports for the same date within a few seconds.

To check the execution time of a script, add a prefix "time" and run the script.

Example:

```
time ./test.py
```
Copied!
content_copy

In order to fix this issue, open the start_date_report.py script using nano editor. Now, modify the **get_same_or_newer**() function to preprocess the file, so that the output generated can be used for various dates instead of just one.

```
nano ~/start_date_report.py
```
Copied!
content_copy

This is a pretty challenging task that you have to complete by modifying

the **get_same_or_newer**() function.

Here are few hints to fix this issue:

1.   Download the file only once from the URL.

2.   Pre-process it so that the same calculation doesn't need to be done over and over again. This can be done in two ways. You can choose any one of them:

- To create a dictionary with the start dates and then use the data in the dictionary instead of the complicated calculation.

- To sort the data by start_date and then go date by date.

Choose any one of the above preprocessing options and modify the script accordingly.

Once you've completed modifying the Python script, save the file by clicking Ctrl-o, the Enter key, and Ctrl-x.

Run the start_date_report.py python script:

./start_date_report.py
Copied!
content_copy

Output:

```
student-04-a8a259bc22ec@linux-instance:~$ ./start_date_report.py

Getting the first start date to query for.

The date must be greater than Jan 1st, 2018
Enter a value for the year: 2019
Enter a value for the month: 1
Enter a value for the day: 1

Started on Jan 05, 2019: ['Lucy Calhoun']
Started on Jan 11, 2019: ['Macon Livingston']
Started on Jan 12, 2019: ['Curran Farley']
Started on Jan 13, 2019: ['Lucius Glass']
```

Now, you've improved the performance of the script.

Click Check my progress to verify the objective.

Improve performance

Check my progress

Congratulations!

Congrats! You've successfully fixed errors, bugs, and increased the performance of execution. Debugging an issue from a program and reducing execution time by fixing a repeatable call will be beneficial as an IT Specialist.

End your lab

When you have completed your lab, click **End Lab**. Qwiklabs removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied
- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.