Lab Instructions: Retrieve and Display Remote Directory Contents Using Fabric

1. Ensure that you are in your python virtual environment, it it is not active, use the following command to activate it:

   source my_python_env/bin/activate

2. Remove the fabfile.py that you have created in last lab, to do so execute the following command:

   rm fabfile.py

3. Verfiy if the file is removed using 'ls' command.

Step 1: Prerequisites

1. Install Python **(ignore if it is already done):**

- Ensure Python (3.6 or later) is installed on your system.

- Verify the installation:

  python3 --version

2. Install Paramiko **(ignore if this already done):**

- Paramiko is required for SSH automation. Install it using pip:

  pip install paramiko

3. Create a Virtual Environment **if it is not created** :

- Use a Python virtual environment to isolate your project.

  python3 -m venv venv

- Activate the virtual environment:

  source venv/bin/activate # For Linux/macOS

- Install fabricwithin the virtual environment:

  pip install fabric

Step 2: Create the Fabfile

1. Create a Fabfile.

    1. In this step, we will create a text file called sftpupload.txt in our local machine. To do so, execute the following command:

        nano fabfile.py

```
from fabric import task, Connection

REMOTE_HOST = "192.168.1.166"
USERNAME = "rps"
PASSWORD = "rps@123"

@task
def get_remote_directory_contents(c):
    """Retrieve and display the contents of a remote directory dynamically."""
    # Accept remote directory as user input
    remote_directory = input("Enter the remote directory path: ")

    print(f"Querying the contents of the remote directory: {remote_directory}\n")
    try:
        # Establish connection to the remote machine
        conn = c.Connection if hasattr(c, 'connection') else Connection(
            host=REMOTE_HOST,
            user=USERNAME,
            connect_kwargs={"password": PASSWORD},
        )
        conn.open()  # Explicitly open the connection

        # Command to list directory contents (Windows-specific)
        command = f'dir "{remote_directory}"'
        result = conn.run(command, hide=True)

        # Display the output
        print(f"Contents of {remote_directory}:\n")
        print(result.stdout.strip())

        conn.close()  # Close the connection

    except Exception as e:
        print(f"Error obtaining remote directory contents: {e}")
```

Step 3: List Tasks in Fabfile

1. List Tasks:

- Use the fab command to list available tasks in the fabfile.py:

fab --list

2. Expected Output:

- You should see the following task listed:

get-remote-directory-contents Retrieve and display the contents of a remote directory dynamically.

Step 4: Execute Tasks

1. Run the Task:

- Execute the g**et-remote-directory-contents** task:

fab get-remote-directory-contents

2. Provide Input:

- When prompted, enter the remote directory path you want to query (e.g., **home/rps/Downloads**).

Step 5: Verify Virtual Environment

1. Check Active Virtual Environment:

- Ensure the virtual environment is active. The prompt should include (venv).

2. Deactivate When Done:

- Deactivate the virtual environment to exit:

deactivate