Lab Instructions: Automate Remote Host Login using Paramiko

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
```

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 Paramiko within the virtual environment:

```
pip install paramiko
```

Step 2: Create the Python Script

1. Now, lets create a python script with name **'Automate_Remote_Host_Login_using_Paramiko.py'** using nano editor. To do so, please enter the following command:

nano Automate_Remote_Host_Login_using_Paramiko.py

- Create a file named **Automate_Remote_Host_Login_using_Paramiko.py** in your working directory and add the following content:

```
import paramiko
import time

def connect_to_remote_host(hostname, port, username, password):
    try:
        # Initialize SSH client
        ssh_client = paramiko.SSHClient()

        # Automatically add the host key if it's not already in known_hosts
        ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())

        # Connect to the remote host
        print(f"Connecting to {hostname}...")
        ssh_client.connect(hostname, port=port, username=username, password=password)
        print(f"Connected to {hostname}!")

        # Open an interactive shell session
        shell = ssh_client.invoke_shell()
        print("Interactive shell started. You can now run commands on the remote host.")
        return ssh_client, shell

    except Exception as e:
        print(f"An error occurred during connection: {e}")
        return None, None

def interact_with_remote(shell):
    try:
        while True:
            # Prompt the user for commands to execute on the remote host
            command = input("Enter command to execute (or type 'exit' to quit): ").strip()
            if command.lower() == 'exit':
                print("Exiting interactive shell...")
                break

            # Send the command to the remote shell
            shell.send(command + '\n')

            # Wait for the command to execute
            time.sleep(1)

            # Read all available output
            output = ""
            while shell.recv_ready():
                output += shell.recv(1024).decode()
            # Display the command output
            print(output)
    except KeyboardInterrupt:
        print("\nExiting interactive session...")
    except Exception as e:
        print(f"An error occurred: {e}")


if __name__ == "__main__":
    # Replace these with your remote host details
    hostname = "192.168.1.166"  # Remote host's IP or domain
    port = 22                   # Default SSH port
    username = "rps"          # Your username
    password = "rps@123"          # Your password

    # Connect to the remote host and start the interactive shell
    ssh_client, shell = connect_to_remote_host(hostname, port, username, password)

    if ssh_client and shell:
        # Interact with the remote host
        interact_with_remote(shell)

        # Close the connection after the session
        ssh_client.close()
        print("Connection closed.")
```

Step 3: Run the Script

1. Execute the script:

- Run the script using the following command:

```
python Automate_Remote_Host_Login_using_Paramiko.py
```

Once the script execution is successful, you have your remote host terminal to interact with

Step 4: Verify the Output and close the remote host connection

1. Ensure that the commands are executed on the remote host: (e**g. Ifconfig, ls, ls –l, mkdir)**

- Check the output displayed in the terminal.

2. Verify the execution of each command by logging into the remote server manually (if needed).

3. To close the remote host connection, please enter the following command:

```
Exit
```

You will see the output stating that ‘**Connection closed’**

4. Now, you are in your local host, to verify type the following command:

```
ifconfig
```

You will see the IP address of the local host.

Step 5: Deactivating 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
```