Student Name: **Weight:** **20%**

Student ID: **Marks:** **/30**

Assignment: Designing a Network

Read the scenario below and answer the questions that follow. Your overall goal is to design a network that addresses the needs of the client in the scenario.

Scenario

You are working for client who needs to relay important information from client machines and a server machine to a central location where this information can be processed and monitored. Performing these tasks manually would take a lot of time, so automation is the best approach in this situation. The Client machine has regularly occurring cron jobs that run two python scripts in sequence. The cron jobs should run at boot (the SysAdminTask.py runs followed by Client.py). The company wants you to recommend the best technologies for this purpose.

Instructions

1. You will use 2 virtual machines for this project.
    - Machine 1 = Admin Machine (OS can be Linux or Windows)
    - Machine 2 = Client Machine (OS must be Linux)
2. Proper error handling should be in place to ensure that the SysAdminTask.py has been run and will produce the correct output file before running the Client.py script.
3. You can choose what the SysAdminTask.py script does or may reuse the script developed for the Assignment 2 that creates the JSON file. If you are reusing, make sure to rename file to SysAdminTask.py
4. When the Client Machine is rebooted, it will automatically send a file over the network to the admin machine which has a python server script waiting to receive files. The file will contain data collected from resident script on the Client machine.

Deliverables

- Business document that contains network standards and pseudocode.
- You must use 3 scripts:
    - Server.py
    - Client.py
    - SysAdminTask.py
- Demo will be in-person during class hours.

Tips for Success

- Make sure the Client Machine can send the output for the SysAdminTask.py script to the Server Machine manually. Once this is confirmed, set up the crontab on the client machine to schedule the SysAdminTask.py, followed by the Client.py scripts to run at reboot.
- Ensure that there is a proper handshake & signoff between the sever and client.
- Ensure that the client attempts to transfer the file more than once in case of an error in transfer.
- Establish response codes between server and client to manage effective communication.
- Show the transfer progress with a progress bar using tqdm module.

Marking Criteria

Categories	0	1	2	Score
SysAdminTask.py	Not functioning	Runs without errors but does not run automatically	Runs correctly as a cron job	**/2**
Business Document clearly establishes how Client & Server communicate	None	Not all aspects are taken care of	Well executed	**/2**
Client.py will run automatically on startup & connect to server	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**
Client.py sends the prerequisite info for file transfer	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**

Client.py Sends File to server	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**
Client.py ensures that file is received by Server in its original state	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**
Client.py has sufficient error handling	None	Limited (missing error handling techniques when working with resources, casting data types, etc.)	Well-executed	**/2**
Server.py accepts connections perpetually	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**
Server.py receives the file parameters and process them	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**
Server.py Receives File	Not functioning	Attempted (code exists to accept the file but the file is not created properly)	Runs correctly, as proven in demo	**/2**
Server.py ensures that file received is accurate	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**
Server.py ensures to keep all the historical files from all clients	Not functioning	Attempted (code exists but is not fully implemented and may not be working in all cases)	Runs correctly, as proven in demo	**/2**

Server.py has sufficient error handling	None	Limited (missing error handling techniques when working with resources, casting data types, etc.)	Well-executed	**/2**
Transfer progress bar	Not functioning	Attempted (code exists to accept the file but the file is not created properly)	Runs correctly, as proven in demo	**/2**
Code Readability	Insufficient (very difficult to follow the coding solution, lacks organization and logical structure)	Missing concise pseudocode, missing sufficient code commenting, or missing proper code structure	Code is easy to understand. Provides proper documentation.	**/2**
			Total	**/30**