Smart Thermostat Project Submission

Scott Nguyen

August 8, 2025

1 How to Run the Project

1. Extract the provided `project.tgz` archive:

   ```
   tar -xzvf project.tgz
   cd thermostat_project_folder
   ```

2. Make sure `project.sh` is executable:

   ```
   chmod +x project.sh
   ```

3. Launch the host server and QEMU virtual machine:

   ```
   ./project.sh
   ```

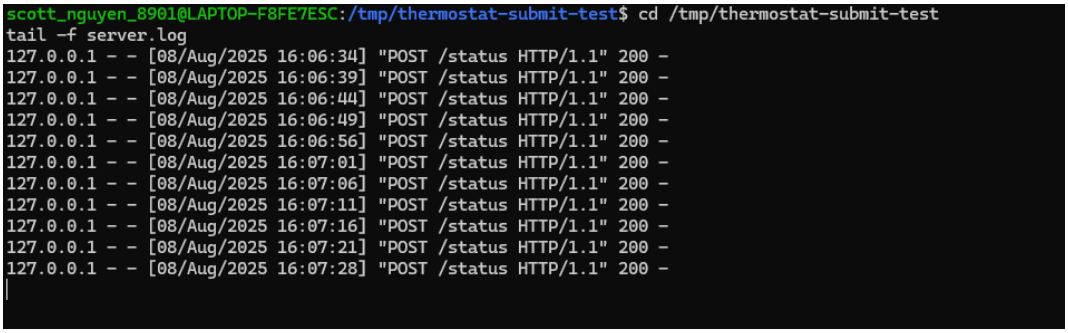
   This script:

   - Starts `server.py` on the host (port 8000).
   - Boots the QEMU ARM VM with the thermostat client running automatically.

2 Verification Steps

2.1 1. Server Receives Status Updates

The host `server.log` shows regular POST requests from the VM client to the `/status` endpoint every 5 seconds.



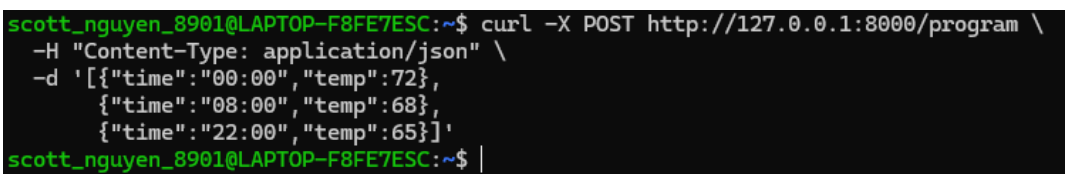
```
scott_nguyen_8901@LAPTOP-F8FE7ESC:/tmp/thermostat-submit-test$ cd /tmp/thermostat-submit-test
tail -f server.log
127.0.0.1 - - [08/Aug/2025 16:06:34] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:06:39] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:06:44] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:06:49] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:06:56] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:07:01] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:07:06] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:07:11] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:07:16] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:07:21] "POST /status HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2025 16:07:28] "POST /status HTTP/1.1" 200 -
```

Figure 1: Regular status updates sent from VM client to server.

2.2 2. Remote Schedule Update via HTTP POST

A new temperature program is sent from the host to the VM using:

```
curl -X POST http://127.0.0.1:8000/program \
-H "Content-Type: application/json" \
-d '[{"time":"00:00","temp":72},
{"time":"08:00","temp":68},
{"time":"22:00","temp":65}]'
```



```
scott_nguyen_8901@LAPTOP-F8FE7ESC:~$ curl -X POST http://127.0.0.1:8000/program \
  -H "Content-Type: application/json" \
  -d '[{"time":"00:00","temp":72},
       {"time":"08:00","temp":68},
       {"time":"22:00","temp":65}]'
scott_nguyen_8901@LAPTOP-F8FE7ESC:~$
```

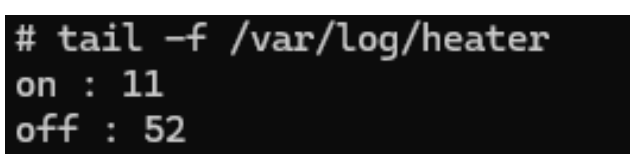
Figure 2: Host sends updated program to the thermostat client.

2.3 3. Heater Control Log in VM

Inside the VM:

```
tail -f /var/log/heater
```

The log shows the heater turning `on` and `off` based on temperature readings.



```
# tail -f /var/log/heater
on : 11
off : 52
```

Figure 3: Heater on/off events with timestamps.

2.4 4. Simulated Temperature Readings in VM

The thermostat reads from `/tmp/temp` (simulated thermocouple). This can be observed with:

```
watch -n 1 cat /tmp/temp
```

Figure 4: Simulated temperature readings inside VM.

2.5 5. Post-Update Confirmation in Server Log

After sending a new program, the server log confirms receipt with a `POST /program` entry.

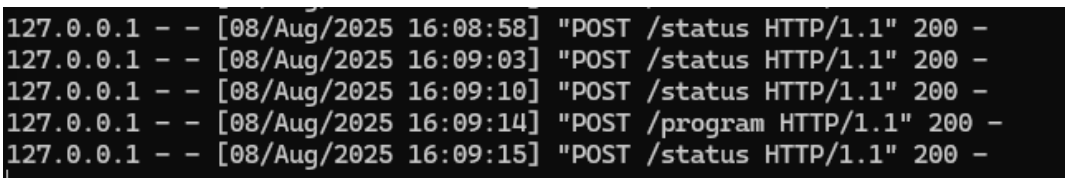


Figure 5: Server confirms program update from host.

3 Conclusion

The tests confirm that:

- The thermostat client starts automatically in the VM.
- The client reads simulated temperature data.
- Heater control logic works and logs actions.
- Status is reported to the host server.
- Remote schedule updates are accepted and applied.

This satisfies the core requirements for the Smart Thermostat IoT project.