CMP408 – Mini Project Report

2003115 Robbie Sangster

1 Contents

1 Contents 2
2 Introduction 3
2.1 Objectives 3
3 Methodology 4
3.1 Raspberry Pi Setup & Configuration 4
3.2 AWS – EC2 Instance Security 6
4 Conclusions 7
4.1 Future Work 7
5 References 8

2 Introduction

The Pi hardware monitor is designed to allow users to view the important hardware data remotely through a webpage and Amazon Web Services (AWS). Uses cases for this project would be users who use a Raspberry pi for controlling process in a smart home or use the Pi as a remote server. This project would allow them to receive meaningful data regarding the state of their pi without the need to manually inspect it. The UK is expected to have over half the number of homes with smart home integration by 2027 (Business Matters, 2023) which could lead to a large uptake in Raspberry Pi users. A Raspberry Pi zero will collect its latest hardware data such as CPU Memory, Disk space and temperature at the push of a button. This is collected using the psutil library and put into json format in a file 'pi_data.json' which is then uploaded to AWS EC2 instance using rsync. The Pi is also connected to a breadboard which has two led lights on it. These lights will activate depending on the success or failure to upload to aws. If the file successfully transfers to aws the yellow led will turn on, if it fails the red led will turn on. The LEDs are activated using a Linux Kernal module (LKM) and will activate the correct GPIO pin matching the upload status. The json data can then be viewed on a HTML webpage hosted on AWS EC2 which will display the most recent hardware data. The EC2 instance has been configured to allow traffic from port 22 and port 80.

2.1 Objectives

1. Raspberry Pi
   - Collecting data relating to CPU (temperature and frequency), count and percentage, memory, and disk diagnostic information with a timestamp of when the data was collected.
   - A button connected to GPIO pins will execute a Python script that will collect the data.
   - Two LEDS will be used to visually display the status of the file uploaded to AWS.
2. AWS
   - AWS EC2 instance will host a HTML page that users can view to see the most recent hardware information and refresh for the latest results.
   - Security Group Policies, Private Keys and Elastic ip address will allow for authorized access and protect the security integrity of the AWS and PI.

3 Methodology

3.1 Raspberry Pi Setup & Configuration

A button connected to the Raspberry Pi when pressed will execute the file 'pi-data.py'. This file will firstly collect the hardware data using the psutil library. Data points for the Pi CPU percent, CPU frequency, CPU count, disk total, memory total, CPU temperature and the current date and time. This data is then put into a JSON format and dumped into the file 'pi_data.json'. With the data collection complete the pi will then upload the json file to aws using a rsync command. The rsync command takes in arguments of the json file in the form of the path which is /home/pi/pi_data.json, The EC2 instance username ec2-user, The elastic ip of the instance assigned to keep the ip from being released after instance reboot and the location on the instance being /home/ec2-user. To secure the EC2 instance the upload will require the Pi to submit a private key alongside the data file. This will prevent unauthorized access to the instance preventing potential attackers gaining access. The command when executed in the terminal looks like this.

```
Rsync -av -e ssh -i /home/pi/CMP408.pem /home/pi/pi_data.json ec2-user@44.207.97.0
:/home/ec2-user
```

Figure 1: Rsync command.

This command is executed using subprocess.check_output(rsync). Which will then get response of "OK" is successful or display the error if failed. Rsync does the same function as scp but works more like a synchronization rather than copy. Rsync transfers the difference of the files making it more efficient transferring data, which reduces the cost of AWS charges of file transfer as it only looks at the difference of the file size. Finally, the script will switch the LED depending on the response. If successful, the yellow LED if it fails the red LED will turn on. The LEDs are activated using a LKM and depending on the response determines which LKM and LED. The following figures 2.0 and 2.1 illustrate the process of the project and physical setup.

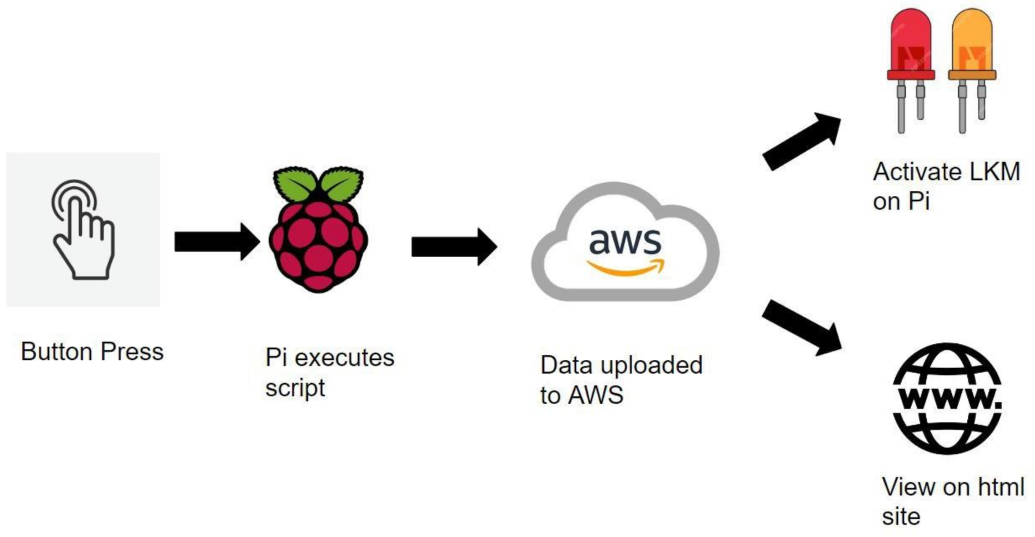


Figure 2: Process flow diagram

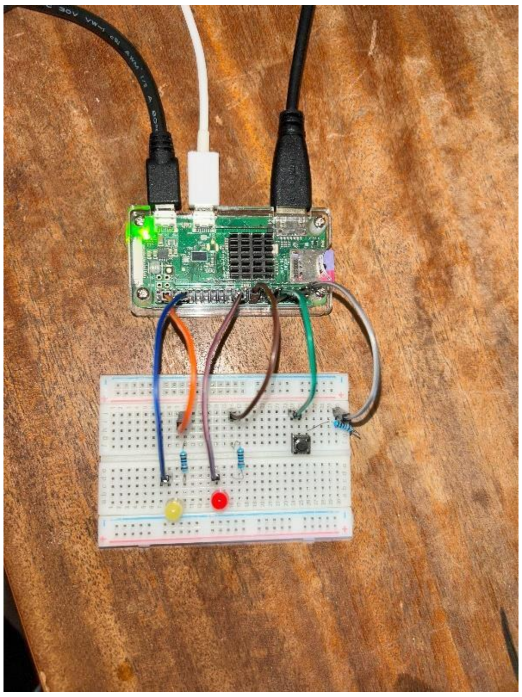


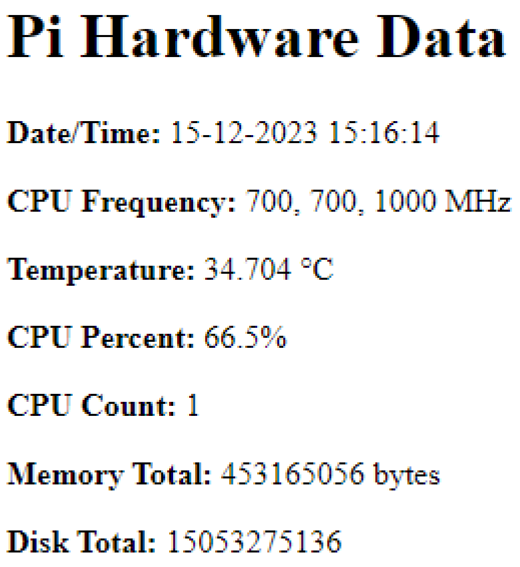
Figure 3: Raspberry Pi Hardware configuration

The Button is connected to pin 10 and file buttonPush.py, the red LED is connected to pin 23 and the yellow LED is connected to pin 20. The file buttonPush.py detects input from the user to begin the process and the LKMs connected to the LEDs were created on the provided module VM and by adapting the lab05 code. The connection is set up through a Python script named buttonPush.py, which is designed to detect user input and trigger the Linked Kernel Module (LKM) process. This process involves interacting with two LEDs: the red LED connected to PIN 23 and the yellow LED to PIN 20. For script detection, the button is connected to PIN 10. The LKM was developed and supplied from the VM module, and the code from Lab05.

3.2 AWS – EC2 Instance Security

An AWS EC2 instance was created to not only store the data uploaded from the pi but also host the HTML page. The EC2 instance was created with the default Amazon linux AMI and created a unique private key to allow only users with the key to connect via ssh. The PEM file created containing the key was sent to the Raspberry Pi as it needs to ssh to the instance to upload the data. The private key can be a security concern as anyone with the key has access to the instance. This requires careful use of who has access to it and the permissions it has. The PEM file has only access from the owner which is us to read the file. Should the PEM file get into the wrong hands the AWS management console can revoke key access and generate new keys. To use the key to ssh to the EC2 instance the -i followed by the key seen in figure 1 to connect using the key to authenticate.

An Elastic Ip address is also required when connecting to an EC2 instance via ssh protocols. AWS will release the associated ip address when stopping an instance and assign a new one when you restart it. An elastic ip is required to reserve a single ip address to the instance. This is because the ip address is defined in the python script as a constant, so it needs to always stay the same. A security group was also created to allow the correct traffic to the instance. Ports 22 for ssh and port 80 for http web hosting are needed. Port 22 can be configured with the Raspberry Pi ip to allow only it to ssh in with the private key. In order to host the html site on the EC2 instance an Apache server was installed using the httpd service. The site uses a JavaScript fetch command to collect the contents of the json file and output them on the site.
 This was then configured inside the httpd.conf file to direct traffic to the create site seen below in figure 4.


Pi Hardware Data

Date/Time: 15-12-2023 15:16:14

CPU Frequency: 700, 700, 1000 MHz

Temperature: 34.704 °C

CPU Percent: 66.5%

CPU Count: 1

Memory Total: 453165056 bytes

Disk Total: 15053275136

Figure 4: HTML Site

4 Conclusions

In conclusion, the Pi hardware monitor can successfully retrieve and store appropriate hardware data inside a json file, connect and upload data to AWS EC2 then display this information on an HTML website. The project makes use of libraries such as psutil and subprocess, hardware components to listen for inputs and visually reflect status of processes via LEDs and LKMs. AWS services EC2 and appropriate configurations using security groups and elastic ip addresses. Web hosting and Apache server configurations on EC2 to view data from the web.

Data upload is done securely using ssh and rsync alongside the private key to allow only authorized access. The project is also very cost effective using less than a dollar during development and testing. Overall, the project has met its goals and is a working cost-effective solution.

4.1 Future Work

The project in future could look towards automating the data acquisition process by implementing a timer system which would automatically make data requests hourly or daily. This would make the process completely hands free and the user would be able to view the most recent data whenever they need. This could also lead to adding alerts to the project where the user would receive a notification if certain data points exceeded limits such as CPU temperature gets too high, or the CPU percentage exceed limitations. Another potential feature would be improving the html site with graphs and charts to display data in a more visually appealing way. Graphs and charts can help users understand what they are seeing better and would make the Hardware Monitor easier to use and benefit from it. Finally, an archiving option to store each request to allow users to see how their Pi performed over time period or where usage spiked for example.

5 References

(2013) *AWS Academy*. Available at: https://aws.amazon.com/training/awsacademy/ (Accessed: 16 December 2023).

PSUTIL documentation (2023) *psutil documentation - psutil 5.9.6 documentation*. Available at: https://psutil.readthedocs.io/en/latest/ (Accessed: 15 December 2023).

Linuxize (2020) *Rsync command in linux with examples*, *Linuxize*. Available at: https://linuxize.com/post/how-to-use-rsync-for-local-and-remote-data-transfer-and-synchronization/ (Accessed: 15 December 2023).

Martin, C. (2023) *How to build a Raspberry Pi system monitor for PC*, *Medium*. Available at: https://medium.com/@chritin111/how-to-build-a-raspberry-pi-system-monitor-for-pc-90079df66713 (Accessed: 16 December 2023).

Paralkar, K. (2023) *Build a raspberry pi monitoring dashboard in under 30 minutes*, *Opensource.com*. Available at: https://opensource.com/article/23/3/build-raspberry-pi-dashboard-appsmith (Accessed: 16 December 2023).

Marquez, E. (2022) *How to create an EC2 instance from AWS console: TechTarget*, *Cloud Computing*. Available at: https://www.techtarget.com/searchcloudcomputing/tutorial/How-to-create-an-EC2-instance-from-AWS-Console (Accessed: 16 December 2023).

Business Matters (2023) *More than half of UK homes to be Smart Homes by 2027*, *Business Matters*. Available at: https://bmmagazine.co.uk/in-business/more-than-half-of-uk-homes-to-be-smart-homes-by-2027/ (Accessed: 16 December 2023).