IoT Experiential Learning

Week 1 Journal

1. **Describe the experience and what you hope to gain from participating in the experience.**

   The first week experience was a chance to begin dialogue with my colleagues in our discussion forums. Learning about the different backgrounds of those whom I will be working with in the coming weeks has given me insight into the different perspectives, strengths, and weaknesses of those around me. The first week has also served as a change to begin reviewing programming in Python as well as researching new Ideas for a project with regards to cyber security and the Internet of Things (IOT). Finally, the first week was a time to try out using the Raspi and GrovePi+ together create a weather station with a temperature sensor and LCD output.

   This module presented my first experience with a Raspberry Pi device. This was also my first experience with Python outside of a learning environment. With this experience, I hoped to familiarize myself with the physical Raspberry PI device itself including the on-board components such as the various outputs and ports as well as the Grove Pi+ board and sensors. I also hoped to become familiar enough with Python to create basic coding that would run without errors. Through this experience, I developed a knowledge base sufficient to achieve a successful outcome in project one. With this knowledge base, I hope to be adequately prepared to work on more complex projects in the future.

2. **Provide an overview of tasks and key activities (training, discussions, labs, assessments, etc.) in which you were engaged during the week.**

   This week I took part in setting up my Raspberry Pi and GrovePi+ board for work with the interactive lab material. Originally, the board was failing to work with the GrovePi+. After some research and scouring online forums, did I find that there was not official support for the the new Raspi 3 B+ working with the GrovePi+ kit at this time. I then moved onto downloading and installing the latest edition of Raspbian. After updating and connecting the Raspi, I moved onto looking at the Raspi terminal interface after connecting the GrovePi+. Running a "dmesg" command showed that the Raspi had detected the BCM 2708, i2c interface. Generally on a Linux device, when the kernel detect the hardware but does not load the kernel drivers, running the "modprobe" command in combination with the target Linux kernel module will activate the device (Tutorials Point, n.d.). The information regarding the fix can be found on a tutorial by Adafruit at the following link (https://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio- setup/configuring-i2c).

   I began working during the week prior to the course opening to start working on checking the hardware I purchased as well as a Python course to get a better understanding of the language. In addition to the Raspberry Pi videos located in the Module Resources section, I also watched two "help" videos from raspberrypi.org. These helped me understand what to

expect when setting up my device for the first time. As the device I purchased came with an SD card pre-installed with NOOBS, these videos provided some clarification as to how to handle the initial boot of the device. Once the device was set up, I took some time to explore the Raspbian OS. I reviewed some discussion boards at raspberrypi.org and raspberrypi.stackexchange.com for some basic operating information such as changing the time zone and navigating the command prompt.

5 Feb 2018: I opened the hardware that I purchased for the course. The first step was to unpack the Raspberry Pi 3 (RPi3) and get it started under Raspbian (which was supposed to be installed on the included SD card.) However, this was not the case, the included SD card would not boot and finish configuring. After several attempts, I decided to forgo that install and use the recommended "Raspbian for Robots" spin from Dexter Industries. Following the procedures outlined on the Dexter Industries site, I was able to download, install, and boot their image on my RPi3 (Dexter Industries, 2018). After that success, I installed the GrovePi daughter board, connected the LCD screen and Temp/Humidity sensor and got the initial Project to run with no modifications.

8 Feb 2018: After looking through the python project files that were included with the GrovePi installation I quickly realized that my Python was not up to par and I would need to dive deeper to be able to complete these projects. I used my team account at Linux Academy (Linux Academy, 2018) to start taking some lessons on Python. I was able to complete several lessons from the Python 2.7 for System Administrators course on the following topics:

- Creating and Running Python Scripts
- Using Comments in Python
- Data Types – Strings and Numbers
- Data Types – Booleans and None

9 Feb 2018: I continued with the less from the Python Course on Linux Academy. I was able to complete the following lessons:

- Working with Variables
- Lists and Tuples
- Dictionaries (dicts)
- Control Flow – Conditionals and Comparisons

12 Feb 2018: This evening I wrote my discussion board assignment for the week. This was the initial inquiry into "What is IoT and how does it impact our lives". Initially I thought I knew the answer to this question, however, after thinking about it and doing some reading, I realized that I was only partially correct in my assumptions. I initially assumed that IoT covered sensors and lower powered devices, however, it is very inclusive of assistants, televisions, and even cars.

13 Feb 2018: This evening I started my first real work with the IoT week 1 project. After working to modify and run code using IDLE on the RPi3's desktop I decided that it was too cumbersome. I decided that I would forgo using the Python IDLE environment on the RPi3 and use PyCharm's capabilities for using remote python interpreters and deployments to modify and run the project's code. I made sure the SSH port was open on the RPi3 and that its wireless adapter was connected to my home network. From here I was able to connect to from my Windows PC through SSH and setup PyCharm to use the RPi3 as the remote interpreter. Making this switch took most of my time for the evening but in the end, I believe it was worth it as I would have a more comfortable workflow. Without jumping between devices, I could have the IDE open along with documents and websites for research.

14 Feb 2018: I continued work on the week one project by starting to modify the code as per step 15 of the assignment. I will be honest, that I had to look up the formula for converting Celsius to Fahrenheit. Once I had the formula I was able to easily translate that into Python. I started by placing the snippet of code in the try/catch of the program. However, I realized that it would be better if I created a function to convert it and simply call the function where I needed it. So I defined a function *celToFahr(temp)* that would return the a value based on this evaluation: *tempf = int((temp * 9 / 5) + 32).* I began having issues with NaN errors that would intermittently pop up and cause the screen to blank out. I tried slowing the refresh rate of the screen by extending the sleep timer but they were still getting through. This was a problem that I needed to research.

15 Feb 2018: I wasn't able to make any headway on the NaN issue so I referenced some of the work that a fellow student posted on the discussion board. It seems I was on the correct path with slowing the polling but I adapted his technique of moving the sleep timer into the try so it would only sleep on success. With that part completed, I moved on to the optional extra practice portion of the project. After some research and going back through the Python lessons I came up with initializing a WARM and COOL variable and set them to 74/73 respectively (the ambient temperate in my room). I then added an if/elif that compares temperature variable to the WARM/COOL value and runs the *setRGB* function to change the color of the screen. The image below shows the LCD screen and my setup for PyCharm running code on the RPi3.

Figure 1 Running Week 1 IoT project from PyCharm

16 Feb 2018: I finished off this week's work with responding to discussion board posts. The most interesting part of this was Shelby's comments about how outdated IPv4 is. This got me thinking about IPv6 and how that fits into IoT. I found some really great resources about IPv6 and how it can actually increase security through native IPSec tunnels and better name resolution. This gave me some ideas to try to implement IPv6 in my project if I am able to. I would like to try to implement an IPSec tunnel between the device and the server component (I would use a DigitalOcean Droplet.)

3. **How did the activities in which you were involved demonstrate your knowledge, skills, and abilities in meeting your selected program competencies?**

This first week was a good way to get introduced to the GrovePi platform and conduct some basic manipulation of Python code to establish the sensors for the project. Getting the RPi3 up and running I was able to use my "day-job" skillset of Linux Administration and previous experience with RPis in general. I have several RPis already and have been familiar with getting SD cards ready for use as well as installing distros like Raspian. Setting up the remote code execution, I was able to use previous experience with SSH and the JetBrains IDEs to be able to deploy code from my PC to the RPi3 and run it. Overall this week was suited to my strengths as a systems administrator. Though I feel that subsequent weeks I won't be able to rely on that as much and will have to dig into the developer toolkit more.

4. **Have you implemented any changes this week in your approach to interpersonal communication, collaboration, project/time management or other aspects as a result of your experiences from the prior week?**

Since this is the first week, I have not made any changes to my interpersonal communication, collaboration, and time management skills. In the past, throughout the course of my time at university, I have always regulated my time for work by doing all my work on Wednesday, Saturday and Sunday. This course will give me the opportunity to look into better time management by spreading out my work load. I would like to try this as an option, but I have always been the type of person to sit down and pound to through a mountain of work, rather than spread it out over a large period of time.

While it is not directly related to this course, I have been playing around with creating my own online Wiki for sharing knowledge. I have found the environment, MediaWiki, the same engine used by Wikipedia to be very helpful for the next Cisco Instructor training course that I am offering in late May. This online platform would be an excellent way to better share knowledge and information among my peers in a collaborative project.

5. Reflections on Experiences

This section lays out the key challenges and outcomes of research regarding the experience of the week.

Key challenges

The first challenge I had to face was the pre-configured SD card not wanting to complete the Raspbian install. This would be a blocker as the RPi3 is the basis of the IoT platform. Luckily for me, the week one project gave alternative instructions for installing “Raspbian for Robots” which was successful and installed all the needed software to make the GrovePi work. Overall my general systems administration skills and experience with Linux as well as RPis helped me overcome this issue with little effort. The fact that the installer of Raspbian failing hadn’t occurred to me before. This was due to the high success rate of installs that I had done. I am glad that I unpacked and prepped the hardware before the course started so I wouldn’t get behind on the issue. This challenge reinforced the notion of “pre-flight checks” of hardware so that any issues can be worked out prior to needing the equipment or butting up against a deadline.

In week 1, the key challenge was mostly working with the Raspi and getting it to fully function with the GrovePi+. As state above, the device would not fully work. The primary problem was partial communication with the i2c bus. At first, only a LED light would interact with the board. There was no output for the LCD screen, no input from the digital DHT 11, or the analog light sensor.

Another challenge I faced was the occurrence of the NaN errors that were appearing in the data being received from the temperature and humidity sensor. This was a problem that I

could not solve from previous knowledge. I had to start looking into first what a NaN error meant for the sensor and then I would be able to understand how to mitigate it. While I could understand that it was receiving a value that wasn't a number, I wasn't sure how to mitigate it. I tried slowing the polling of the sensor which seemed to help but it would still cause the screen to flash and errors to slip through. Eventually, I saw a post on the discussion board from a fellow student that not only did he slow the polling, but moved the sleep statement inside the try/catch so it would only print on a successful run. At first, I wasn't satisfied with how I came to the answer, but I then realized, its no different than going to a colleague at work and getting assistance with a problem, which seems to be in the spirit of this course. My general IT problem-solving skillset and knowing where and how to find answers led me to a partial solution. This problem caught me off-guard, I didn't expect the given project files to have an error like this. However, I believe this is more from the lack of precision in the sensor and the inability for the baseline example project to handle the error. Overall, I learned that I need more practice with Python, while I can read, understand it, and write out some basic concepts I need to focus more on problem-solving and debugging when things don't go as planned. While I don't think this problem is avoidable given the inexpensive nature of the sensors in question. However, I feel that this will force me to write better "error trapping" and will start to understand typically errors that can occur with these types of sensors and compensate for them early in the project.

Identification and evaluation of key challenges

These problems were identified as I began testing the different sensors as part of the GrovePi+ kit.

Resolution of key challenges

In order to solve the problems of the Raspi not communicating in full with the GrovePi+ board, I tried by simply turning it off and on again. Sometimes the simplest solution and most obvious can go overlooked. The problem persisted. I then tried installing the "Raspbian for robots" package onto the SDcard. After loading Raspbain for robots, I found the Raspberry Pi was unable to even load the OS. At that point I did another wipe of the SDcard and formatted the newest up to date version of Raspbian. Since the Raspiberry Pi I purchased was the newest and latest around, there is always the change that bleeding edge hardware will not always work. After loading and updating the latest version of Raspbian on the Raspi, I set out to manually install the GrovePi+ program and libraries. As I stated above, it was then that I began to tinker around with the Linux OS. After detecting the board's kernel module, I then researched the manual modprobe commands. After issuing the correct manual commands and adjusting the proper text files, I reinitiated my tests on the sensors stated above. After successfully capturing data and posting text to the LCD, I determined that the problem was fixed.

Prior course work used

As stated above, my prior experience with Linux and IoT devices has proved to be very useful in diagnosing Linux based device issues. Our program that I am in has a introductory course in IoT systems. There we work with Arduino's and Raspi's on a regular basis. Working with students to help them determine hardware to software issues is a regular occurrence.

Resolutions/Outcomes that challenged prior understanding

At this point in time, none of my prior understanding of these devices has been challenged this week. Working with IoT devices can also be an uncertain experience as each device will implement library and systems functions in a different manner. What is certain is that IoT devices are becoming even for prevalent in our day to day lives and will continue to invade every inch of society. Securing these systems, while providing a base model for a stable platform will be key in the future.

What was learned as a result of the experience

This experience has shown me what are the common kernel drivers that are used on the Linux kernel on a Raspi. It has also re-enforced the notion of using brand hardware with unsupported systems can have mixed results. The online video portion of this experience has showed me the broader scope of how to better handle IoT security and the threat landscape that we face. It is also given the idea of creating a IoT device management system. So far I was thinking this would consist of a Django page over an Ansible server for remote device management. Since Ansible primarily functions over an SSH connection, communication between all devices would be encrypted. Ansible allows for the creation of templates for mass deployments and would work well with the Raspi. Also it has the ability to manage IoT Gateway devices that smaller weaker devices could connect to for secure remote management. I will need to look into using both Ansible and Django in more depth to find out the likelihood of this being a possible combination.

Below is a copy of the final version of the code I used. I commented out unused lines of code rather than deleted them so that I could use them for reference if need be.

```
from grove_rgb_lcd import *
from time import sleep
from math import isnan
```

```
dht_sensor_port = 7 # connect the DHt sensor to port 7
dht_sensor_type = 0 # use 0 for the blue-colored sensor and 1 for the white-colored sensor

# set green as backlight color
# we need to do it just once
# setting the backlight color once reduces the amount of data transfer over the I2C line
setRGB(0,255,0)

while True:
    try:
     # get the temperature and Humidity from the DHT sensor

            [ temp,hum ] = dht(dht_sensor_port,dht_sensor_type)
            ftemp = ((temp * 1.8) + 32)
            #if isnan(temp) is False or isnan(hum) is False:
            print("temp =", ftemp, "F\thumidity =", hum,"%")

              # check if we have nans
              # if so, then raise a type error exception
            #if isnan(temp) is True or isnan(hum) is True:
                  #raise TypeError('nan error')

            t = str(temp)
            h = str(hum)
              #ftemp = ((temp * 1.8) + 32)
            f = str(ftemp)

      # instead of inserting a bunch of whitespace, we can just insert a \n
      # we're ensuring that if we get some strange strings on one line, the 2nd one won't be
affected
            if ftemp > 80:
               setRGB(255,0,0)
            if ftemp < 50:
               setRGB(0,0,255)
            setText_norefresh("Temp:" + f + "F\n" + "Humidity :" + h + "%")
            sleep(1.5)
    #except (IOError, TypeError) as e:
              #print(str(e))
              # and since we got a type error
              # then reset the LCD's text
              #setText("")
    except KeyboardInterrupt as e:
              print(str(e))
```

```
                # since we're exiting the program
                # it's better to leave the LCD with a blank text
                setText("")
                break

        # wait some time before re-updating the LCD
    ##  sleep(1.5)
```

References

Dexter Industries. (2018). *Install Raspbian for Robots on an SD Card for the Raspberry Pi*. Retrieved from https://www.dexterindustries.com/howto/install-raspbian-for-robots-image-on-an-sd-card/

Linux Academy. (2018). *Linux Academy*. Retrieved from https://linuxacademy.com/