https://v2.overleaf.com/3859381586sqskzvbqfkpr

Harvesting Field Station Data; Sensors to Jetstream*

Extended Abstract†

Emmanuel Guido
CSU Monterey Bay
eguido@csumb.edu

Jazzly Anderson
University of California Berkeley
jazzlyanderson@berkeley.edu

Tom Slayton
The University of Cincinnati
slaytotd@mail.uc.edu

Sheri Sanders
Indiana University
ss93@iu.edu

Tony Walker
Indiana University
tonlwalk@iu.edu

ABSTRACT

Biological and marine field stations have a long history of collecting environmental information. However, due to a lack of computing resources, this information is not always accessible to those who need it. Fortunately, consumption of information is made easier by modern cyberinfrastructure. Here we outline a proof-of-concept workflow that makes collecting and sharing information easier for researchers. Field stations frequently collect environmental data via remote sensors;to simulate field stations, we collected atmospheric data using a Raspberry Pi. To process and analyze this data, our prototype uses an XSEDE[10] virtual machine (i.e. Jetstream and Wrangler), Mosquitto, Python, PostgreSQL, R, and Drupal. This virtual machine, inlcuding the aforementioned software, can be distributed to researchers as an easy-to-use appliance.

CCS CONCEPTS

• **Information systems** → Cloud based storage; • **Software and its engineering** → *Software functional properties*;

KEYWORDS

Data sharing, Virtual Machines, Data storage, Raspberry Pi, Microcontroller, Database, Internet of Things, Cloud

ACM Reference Format:
Emmanuel Guido, Jazzly Anderson, Tom Slayton, Sheri Sanders, and Tony Walker. 2018. Harvesting Field Station Data; Sensors to Jetstream: Extended Abstract. In *Proceedings of PEARC Conference (PEARC'18).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

As of 2016, there were a recorded 1,286 biological field stations within 120 countries [11].A current issue facing field station researchers today centers around the absence of an easily accessible means of storing and publicly sharing their collections of large-scale data. Field stations serve as a controlled setting where researchers from various scientific backgrounds may come together to study the earth's natural processes and gather data that will hopefully lead to groundbreaking scientific discoveries. To aid in the collection of data from these field stations, we develop a proof-of-concept workflow that uploads sensory data from a Raspberry Pi to a database hosted on a Jetstream virtual machine and visualizes said data on a web page. Jetstream[8, 9] is a cloud-based, high performance computing (HPC) and data analysis tool capable of running virtual machines that offer the simplicity of a researcher's home machine and the power of a modern HPC computer. By introducing a straightforward method of moving and sharing data we hope to broaden the audience of field station researchers, promoting further awareness of current environmental issues.

2 MATERIALS AND METHODS

The central premise of our project involves the transfer of data between four main components: sensor, Raspberry Pi, Jetstream, and website. In an effort to simplify replication, we constructed a publicly accessible Jetstream image pre-installed with the main services utilized throughout our project. With the image being publicly accessible, researchers will be able to launch instances of our image on their own personal machine, providing them with all of the project's necessary services without having to configure them themselves.

The following subsections, as well as Figure 1, serve as an outline of our work-flow and detail the sequential steps taken in the implementation of our project. The specific steps taken include: 1) Prepare the Pi, 2) Build Jetstream Image, 3) Build Database, 4) Connect senstor to Pi, 5) Connect Pi to Jetstream, 6) Build webpage.

2.1 Preparing the Pi

The chosen microcontroller for our project was a Raspberry Pi model 3B with Debian Stretch software and a quad core 1.2GHz Broadcom BCM2837 64 bit CPU. The two main services that required installation or enabling on the Raspberry Pi were Mosquitto and SPI (Serial Peripherial Interface). Mosquitto is a message broker that implements the MQTT (Messaging Query Telemetry Transport) protocol that allows the Pi to publish data to the Jetstream virtual machine. While Mosquitto had to be manually installed onto the device, SPI simply had to be enabled. Enabling SPI allowed us to transfer information between the Raspberry Pi and it's sensor in the later steps of our work flow.

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document

PEARC'18, July 2018, Pittsburgh, PA USA

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

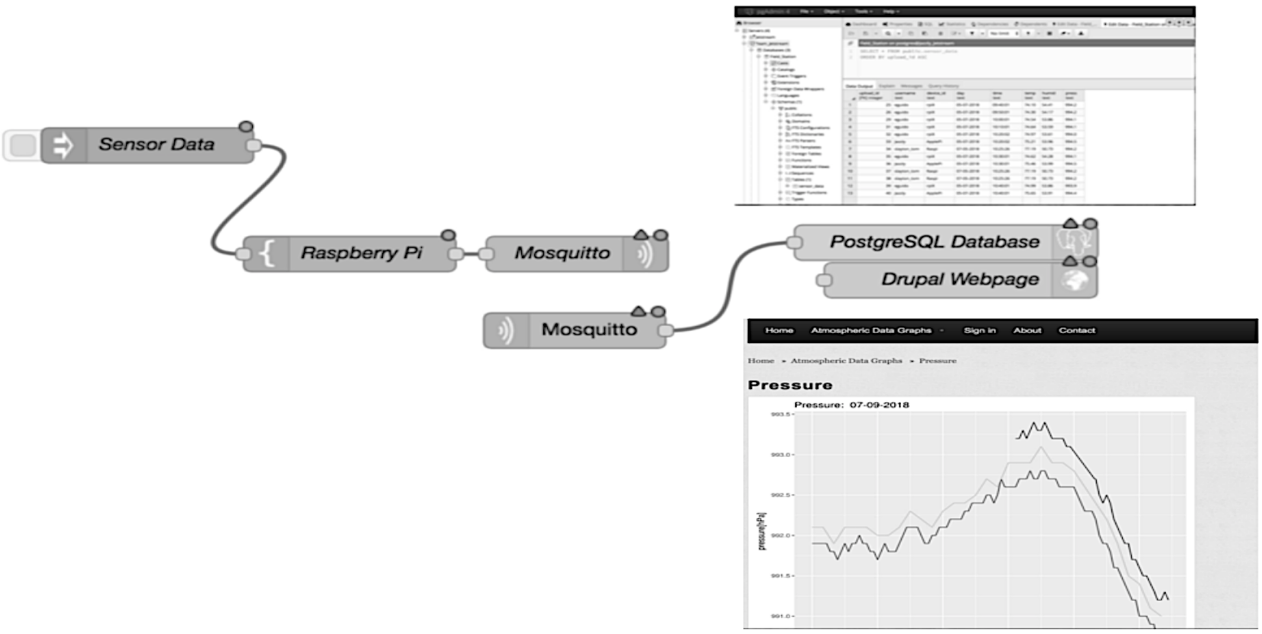

Figure 1: Our work-flow

2.2 Generating a Jetstream Image

Once the Pi was configured, we then launched an instance of a Jetstream image preloaded with the Ubuntu 16.04 operating system,Drupal 7, and LAMP [5] stack services necessary for creating our web page. Mosquitto version 1.4.8[6],PostgreSQL 9.5[4], and Node-Red were also uploaded to our instance which we have since imaged for public use [1]. "Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways"[1]. In our particular project, Node-Red was used to intercept data sent from the Pi and transfer it to the PostgreSQL database.

2.3 Constructing a Database

For this project, we used PostgreSQL to store our sensor data. There are multiple strategies to managing your data in PostgreSQL, including directly through terminal. We chose to use "pgAdmin"[7] as a platform to administrate our database. After initializing PostgreSQL and connecting with pgAdmin, we created and designed our table to fit our data requirements. As well as a time stamp, temperature, humidity and pressure, PostgresSQL includes a "serial" data type, and we used this as a primary key to uniquely identify every new entry. We also included a "Device ID" to distinguish multiple devices owned by one user. With a database running on our Jetstream instance, using crontab and Python scripts, we were able to automatically populate our database at whichever interval we chose. This is the database that our web page will pull data from to generate data views suited to the user.

[1]The image may be found on Jetstream under the name "Harvesting Field Station Data"

2.4 Enabling Sensor-Pi Line of Communication

In order to collect data from the BME280 sensor breakout board we first had to establish a connection between the board and the Raspberry Pi. The breakout board consists of 10 pins. However, we only required the six on the right side which provide SPI functionality. These six pins are ground, power, serial clock(SCK), MISO(SDO), MOSI(SDI), and slave select(CS). To establish the connection we began by soldering a red coated wire to the power pin and a black coated wire to the ground pin. We then soldered white coated wires to the remaining pins. After soldering, we stripped and crimped connectors to the ends of each wire in order to connect them to their corresponding GPIO pins, Table 2, on the Raspberry Pi. Once this was completed we were able to collect sensor data from the board using an open-source pre-written python class found online[2] [2]. This python class reads byte data from the sensor and converts it to human readable units(i.e. Celsius, pascals, etc.) Using this class we wrote a python script that takes the sensor data and inserts it into a string in the form of an SQL insert statement. This SQL statement is then written to a file which is sent through the Mosquitto client on the Pi to the Mosquitto broker on the Jetstream image.

2.5 Enabling Pi-Jetstream-Database Line of Communication

In order to enable our micro-controllers to talk to Jetstream, we used Node-RED, with an added PostgreSQL node to write into our database, Figure 1. After gathering atmospheric sensor data, the Raspberry Pi sends out an SQL file via MQTT which is then

[2]The python sensor class may be downloaded at
$http://abyz.me.uk/rpi/pigpio/code/BME280_py.zip$

Table 1: Sensor to Pi Pin Connections

Sensor	Raspberry Pi
GND	GROUND
3.3v	3.3v
SCK	SCLK (GPIO11)
SDO	MISO (GPIO9)
SDI	MISO (GPIO10)
CS	CE0 (GPIO8)

sent to our PostgreSQL database through Node-RED. As opposed to manually gathering and sending data throughout the day, we established a constantly running cron job that runs the Python script and "mosquitto_pub" command that respectively gather and send data to the Jetstream image.

2.6 Constructing Drupal Digital Platform

Once the database was collecting data from three different sensors, we created a Drupal [3] web page to display graphical views of our data for easier analysis. In order to display the images, we developed an R (version 3.5.0) script that pulls and plots data from the PostgreSQL database and displays it on the Drupal web page. By way of a cron job, the web page's graphs are updated every 24 hours.

3 RESULTS

By the end of our project, we were able to develop a running website that displays sensor data sent from a Raspberry Pi to a PostgreSQL database stored on a Jetstream instance. To make the replication of our work-flow easier for field station researchers or scientist in general, we created a Jetstream image pre-installed with the necessary services and a detailed documentation of our process.

4 CONCLUSION

We were successful in developing a functional website that automatically displays data collected and sent by easily accessible and affordable devices. Furthermore, our work flowâĂŹs inclusion of free services such as Jetstream, Node-RED, Drupal, and PostgreSQL is financially advantageous in that it allows field station researchers to allocate their funds to other aspects of their studies.

4.1 Future Directions

Our current work-flow was developed and demonstrated with data being collected from only three sensors. However, we understand that at many field stations, data may be generated by a multitude of devices resulting in large amounts of data that surpass JetstreamâĂŹs current storage capacity. As a result, the next proposed step of our work -flow would be to attach our Jetstream instance to the Wrangler storage system. Wrangler is a data analysis and storage system hosted on XSEDE designed to assist in the storage and maintenance of massive collections of data. By connecting our Jetstream instance to a Wrangler allocation, we would be able to dump data every 24 hours from the database stored on Jetstream to a long term database hosted on Wrangler. After the data is transferred from the Jetstream database and displayed on the web page, the data would be deleted from Jetstream.

Further additions to our project include the allowance of citizen scientists to access the PostgreSQL database in order to contribute to data collection. Our database currently requires specification of usernames to associate each input of data with the uploader. This enables those analyzing data to filter out potentially unreliable sources or use narrower views for more specific analysis.

A APPENDIX

A.1 Abstract

A.2 Introduction

A.3 Materials and Methods

A.3.1 Preparing the Pi.

A.3.2 Generating a Jetstream Image.

A.3.3 Constructing a Database.

A.3.4 Enabling Sensor-Pi Line of Communication.

A.3.5 Enabling Pi-Jetstream-Database Line of Communication.

A.3.6 Constructing Drupal Digital Platform.

A.4 Results

A.5 Conclusion

A.5.1 Future Directions.

A.6 References

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1445604. This work was also supported by Indiana University. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Indiana University.

This work used the Extreme Science and Engineering Discovery Environment (XSEDE),which is supported by National Science Foundation grant number ACI-1548562, Jetstream research cloud at Indiana University through allocations TG-CIE170025 and TG-BI0170103.

REFERENCES

[1] [n. d.]. https://nodered.org/
[2] [n. d.]. http://abyz.me.uk/rpi/pigpio/examples.html#Python_BME280_py
[3] [n. d.]. Drupal - Open Source CMS. https://www.drupal.org/
[4] [n. d.]. The World's Most Advanced open source relational database. https://www.postgresql.org/
[5] 2004. http://www.apache.org/licenses/LICENSE-2.0
[6] Roger A Light. 2017. Mosquitto: server and client implementation of the MQTT protocol. *Journal of Open Source Software* 2, 13 (2017).
[7] Dave Page. [n. d.]. pgAdmin. https://www.pgadmin.org/
[8] Craig A Stewart, Timothy M Cockerill, Ian Foster, David Hancock, Nirav Merchant, Edwin Skidmore, Daniel Stanzione, James Taylor, Steven Tuecke, George Turner, et al. 2015. Jetstream: A self-provisoned, scalable science and engineering cloud environment. (2015).
[9] Craig A. Stewart, David Y. Hancock, Matthew Vaughn, Jeremy Fischer, Tim Cockerill, Lee Liming, Nirav Merchant, Therese Miller, John Michael Lowe, Daniel C. Stanzione, James Taylor, and Edwin Skidmore. 2016. Jetstream: Performance, Early Experiences, and Early Results. In *Proceedings of the XSEDE16 Conference*

on Diversity, Big Data, and Science at Scale (XSEDE16). ACM, New York, NY, USA, Article 22, 8 pages. https://doi.org/10.1145/2949550.2949639

[10] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, and N. Wilkins-Diehr. 2014. XSEDE: Accelerating Scientific Discovery. *Computing in Science Engineering* 16, 5 (Sept.-Oct. 2014), 62–74. https://doi.org/10.1109/MCSE.2014.80

[11] Laura Tydecks, Vanessa Bremerich, Ilona Jentschke, Gene E Likens, and Klement Tockner. 2016. Biological field stations: a global infrastructure for research, education, and public engagement. *BioScience* 66, 2 (2016), 164–171.