Oregon State University
College of Engineering

CS Capstone Design Document

May 27, 2020

Wireless Trackers

Prepared for

Oregon State University

Steve Fowler _____ _____
Signature *Date*

Prepared by

Group 53

Wireless Trackers

Justin Vaughn _____ _____
Signature *Date*

Kenneth Price _____ _____
Signature *Date*

Jordan NG _____ _____
Signature *Date*

Abstract

Currently the ability to monitor the usage of the wireless network across the Oregon State University (OSU) Corvallis campus is limited. As OSU has continued to improve network access for both staff and students they have determined that a new tool was needed. To be able to visually represent the data of the wireless network, the application that is under development will display the usage of a buildings wireless network through a heat map. In order to approach the development of the application in a efficient way this paper will discus some of assumptions we have made in regards to the development of the application. Using the Airwave API to retrieve data as well as InfluxDB to store the data, we will build a User Interface like Google maps to implement the application. Currently there are some products available that do similar things to what we wish to accomplish and we will be attempting to replicate some of the features that excel.

Contents

1 Introduction

The overall goal of this project is to create a program that is able to retrieve and display the usage of Oregon State University's wireless network. The data is to be retrieved using the Airwave API, which is the tool that the Network Operations team currently uses to monitor the network usage. Our application will take that data and convert it into individual data points that can then be mapped to OSU's buildings. A wide variety of heat maps can then be generated using the converted data points. The conversion will be done through a ratio of bandwidth or maximum occupancy, depending on the current user or intended use. This will allow the Network Operations team to more easily monitor wireless usage across the entirety of the OSU campus.

2 Data Retrieval and Storage

2.1 Data Pulling

Having the ability to pull data from the AirWave API is crucial for the success of this project. The AirWave API is what allows us to be able to do our calculations of data and population of the map. Without the ability to pull data from the AirWave API we would be unable to do much of anything. There is not many different ways that we could go about doing this seeing as though the AirWave API has a specific format to pull data. There isn't much debate here as whether or not this is the best option seeing as though it is our only option since it is what is currently being used by the OSU network to pull data. Unless we are opting for a overhaul of the collection of data on the wireless network, using the AirWave API is our only option.

2.2 Data Parsing

The data that we retrieve from the AirWave API will be in the form of an XML object, which we can then parse to retrieve the data that will need to be stored. We will want to store this data based on the most granular data sets that we will want to display. At this point that means storing it by building name and floor. The data that we store will include WAPs up, WAPs down, and connected clients. We can extract this information using a basic Python XML parser.

2.3 Data Storage

We will store this data using InfluxDB, which is an open source time series database [1]. Using InfluxDB, we can pull and store our data at increments of 15 minutes, and store that data for 48 hours. We can achieve this limited time storage using the built in retention policy in InfluxDB. Then, using what InfluxDB refers to as a continuous query, we can average the data at a set time every data for the previous 24 hour period, which can then be stored in an archive for one year. This will allow the network operations team to view the usage history, without using too much data storage.

3 Data Mapping and Display

3.1 Data Mapping

After we have handled the retrieval of the data, we will then need to map the data in a way that it can be easily translated to a visual representation. To do this, we will need some sort of look-up database that contains a max capacity, and number of access points sorted by the most granular information that we are displaying. Our best option for this is probably MongoDB. It allows us to store the information that we need in a JSON-like document, which is easily search-able[2]. It also allows us to increase the granularity more easily than if we were using an SQL database, which would require us to go in and modify the tables for every change we make.

3.2 Color Coding

To generate the color coding for the final data display, we will take the current usage of the building and divide that by the total bandwidth of the whole building.

$$\text{Usage density network} = \frac{\text{Current usage}}{\text{Total bandwidth}}$$

This would show us the usage of the network and give the Networking and Telecommunications team a clear understanding of the health of their network. Later this can be expanded by using the number of connected clients over the buildings maximum capacity.

$$\text{Usage density occupancy} = \frac{\text{Current users}}{\text{Total building/floor capacity}}$$

This would give us the density of users relative to the buildings size. More statistics can be added as we progress depending on the pertinence of the information. We will also include the number of WAPs that are currently down or up at that location. The capacity of the location will be reduced, based on the number of WAPs that are currently down. We will also keep in mind that the bandwidth of the building is not necessarily permanent. These numbers can change and fluctuate depending on the building or on the evolution of current technology. With this in mind we must keep the data that we use as some sort of variable or macro that can be easily changed depending on the limitations of the building.

3.3 Data Display

To create and display the heat map, we may be using the Google Maps API. Currently the Oregon State Campus map is implemented using the Google Maps API. We could use the preexisting API to create a foundation for our application and then use JavaScript in conjunction with the mapped data that we have to create a heat map display of the the wireless network usage. We can do this by simply color coding the overlays that have already been created for each building based on the data. However When looking at other products that are similar to the one we wish to develop, we can see that there are some other options when it comes to displaying data. Like figure 1 we can see a picture of a very detailed but clean looking UI from ElastiFlow, we should look to replicate some of these features when making our own application.

4 System Access

4.1 Web Application Framework

The UI will be built through the Django framework. There will be an open view for general users and overall reporting, a supervisor view with more detailed information of each location, and an administrator view that will allow changes to the color range deliminators and other critical settings.

4.2 General User

The general user view for this will display the full campus map with the most current color coding for each building. This view will not allow any changes or customization to the view. Potential future development will include the ability to select a building and see each floor's color code.

4.3 Supervisor

The supervisor will have the color codes as well as a detailed menu for each building listing its particular bandwidth status, WAPs up and down, and other key details.

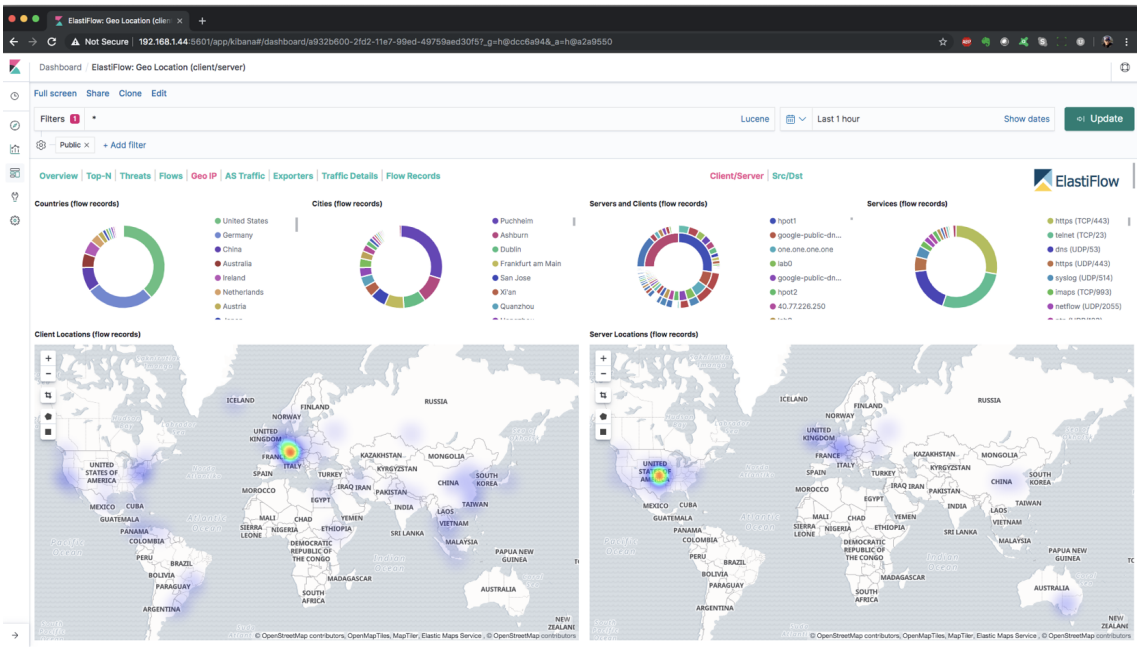

Figure 1. An image of ElstiFlow's UI which clearly shows pertinent data as well as a heat map of the specified filters [3]

4.4 Administrator

The administrator view will show pertinent information to the Networks and Telecommunications department. This may include things like total bandwidth and the usage of that bandwidth or information that should not be available to the public. It will allow for changes to be made to the points at which the color shifts through the selected spectrum to show bandwidth availability as WAP's are updated over time.

5 Timeline

5.1 Fall

In the fall term we plan to continue working on good documentation, and figuring out what we should be accomplishing moving into the winter term. The application should be researched and possibly find products similar to the one that we trying to help develop. This would aid in the creation of this application and we have been taking a look at products such as ElastiFlow.

5.2 Winter

During the Winter term we plan to implement the application and start working on the coding of the project. We will be taking some time in this term to get familiar with the tools at our disposal and possibly finding better ways to implement the application. Also we will attempt to meet the current goal of implementing a wireless heat map of a single building. The building will be two dimensional and display the wireless usage of the building as a whole. The usage will be displayed as a heatmap and shall clearly convey the stress of the network on the building. The data will be stored in a database for use in the future. The basic UI will be flushed out during this term and a finished product should be delivered. The application should be fully functional for a single building, this includes pulling, displaying, and the storing of data.

5.3 Spring

During the spring term we will focus on polishing and optimizing the current application. From there we will aim to add features to the existing application such stress on the network by floor as well as including multiple if not all the buildings on the OSU campus. The UI can contain much more information that can be toggled on or off using overlays so as to not overwhelm the user. Such information could be number of users, usage, or WAPs up or down. We could also implement a student user view which could display only the density of usage relative to the number of users in that building or floor.

References

[1] (2019) Influxdb 1.7 documentation | influxdata documentation. [Online]. Available: https://docs.influxdata.com/influxdb/v1.7/

[2] w3schools. Python mongodb. [Online]. Available: https://www.w3schools.com/python/python_mongodb_getstarted.asp

[3] (2019) Elastiflow. [Online]. Available: https://github.com/robcowart/elastiflow

6 Acronyms

OSU	Oregon State University
API	Application Programing Interface: a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.
Airwave API	The API that we will be using to pull data from the schools network
WAP	Wireless Access Point
UI	User Interfacethe means by which the user and a computer system interact, in particular the use of input devices and software.