Homeland Security

Anomaly Detection Analysis

Jennah Fayyaz, Parham Neyzari, Michael Wilson

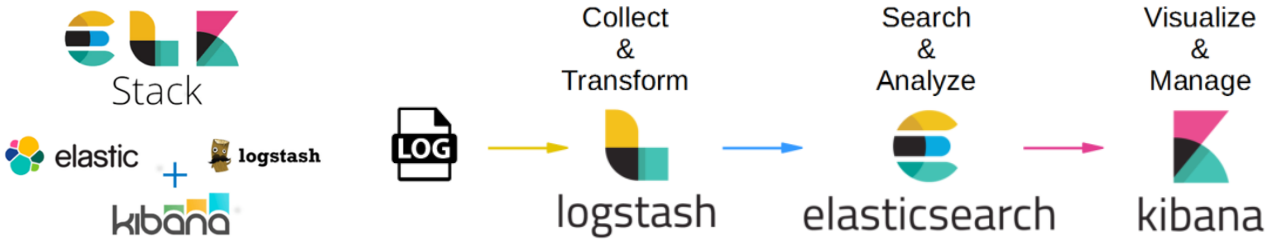
GEORGE MASON UNIVERSITY

Background:

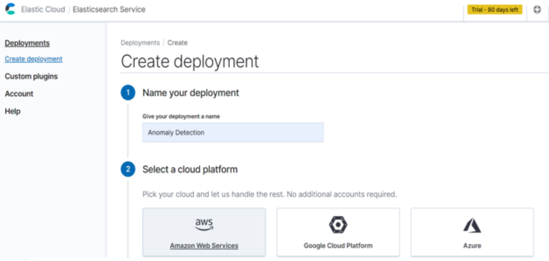
The Department of Homeland Security is the department within the federal government that is responsible for an amalgamation of different tasks, but most importantly, and simply put, for securing the nation. Now that threats are not only physical, but also coming from the cyber web, the Cybersecurity and Infrastructure Security Agency was established in 2018 within the Department of Homeland Security, in order to defend the nation’s critical infrastructure from incoming cyber threats. Their incident response and security testing capabilities in order to ensure the successful and smooth operation of the many departments and agencies within the government, that are dependent on having and maintaining safe networks in order to operate successfully. Anomaly detection would be of interest for an organization who has so much at stake at the risk of a breach and is adamant on a quick and equally correct response.

Purpose:

Data is recorded for every event or interaction that occurs within operating systems or software. This data is called logs, of which there are many different types. The various types of logs include, but are not limited to event logs, security logs and management logs. Different types of software exist in order to hold this data, and to condense it into a readable format for it to be useful. If it wasn’t, within seconds there would be hundreds of thousands of logs that no one would ever sort through and interpret their meaning. Data visualization is an additional capability provided by the software. However, having a system, or a method of being able to detect anomalies through an algorithm, would greatly relieve the strain of having to sort through all the data that is being rendered. The purpose of this project is 3to analyze relevant data and alerting, and creating a corresponding dashboard, which allows for visualization of the data, we would be creating a system in which the output could easily be examined and understood. However, our dashboard would not only display results, but also detect any anomalies within the data logs and bring those to the surface in order to make sure they receive the proper attention that is necessary and required for it. When this occurs, the correct person, preferably the SOC analyst on duty, is notified so that the issue that is causing the anomaly can be resolved.



Our project started off with a need for data. We considered the different options we had, including finding an online dataset, obtaining logs from a source, or by generating our own data set using virtual machines. We began by generating log files using two virtual machines that we created and using those to create a JSON file. We then wrote python scripts in order to parse the data into the information we needed. After creating our deployment, which is hosted on AWS cloud platform, we tested our JSON log file within our deployment by uploading them and letting elasticsearch analyze the information. However, the data was not in the format that elasticsearch recognized, and therefore we were left with two options: to manipulate the data into the format that was required, or to find new data, a set that was compatible with the elastic wizard, which accepts the data and transforms it. As was recommended by the Department of Homeland Security, we chose the second route, where we decided to find a different set of data. We experimented with many different variations, using Windows system logs, Windows security logs, and finally, we found what worked best with our project. We used the logs that were being recorded in real-time, by utilizing the stream live feature. After doing this, we were able to record the data as it came, and test whether it was accurate. When an anomaly was detected within our data, it became clearly evident and was brought to the surface. In order to visualize our results, we created an integrated dashboard by utilizing the particular part of ELK stack, which is known as Kibana. Within Kibana, we were able to create a dashboard and choose the metrics and graphs that best displayed our data for visualization.



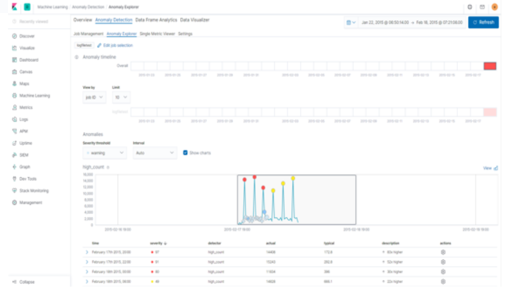
Create deployment

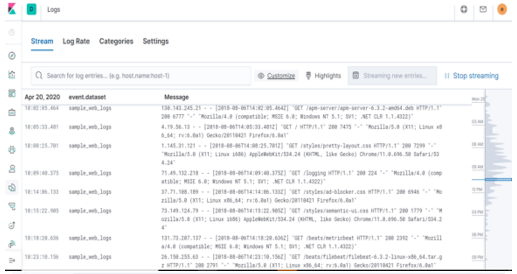
1 Name your deployment

Give your deployment a name: Anomaly Detection

2 Select a cloud platform

Pick your cloud and let us handle the rest. No additional accounts required.





Search for log entries... (e.g. host.name:host-1)

```
Apr 20, 2020  event.dataset     Message
10:02:05.464  sample_web_logs   138.143.245.21 - - [2018-08-06T14:02:05.464Z] "GET /apm-server/apm-server-6.3.2-amd64.deb HTTP/1.1"
                                200 6777 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
10:05:33.481  sample_web_logs   4.19.56.13 - - [2018-08-06T14:05:33.481Z] "GET / HTTP/1.1" 200 7475 "-" "Mozilla/5.0 (X11; Linux x8
                                6_64; rv:6.0a1) Gecko/20110421 Firefox/6.0a1"
10:08:25.701  sample_web_logs   1.145.31.121 - - [2018-08-06T14:08:25.701Z] "GET /styles/pretty-layout.css HTTP/1.1" 200 7299 "-"
                                "Mozilla/5.0 (X11; Linux i686) AppleWebKit/534.24 (KHTML, like Gecko) Chrome/11.0.696.50 Safari/53
                                4.24"
10:09:40.375  sample_web_logs   71.49.132.218 - - [2018-08-06T14:09:40.375Z] "GET /logging HTTP/1.1" 200 224 "-" "Mozilla/4.0 (comp
                                atible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
10:14:06.133  sample_web_logs   37.71.108.189 - - [2018-08-06T14:14:06.133Z] "GET /styles/ad-blocker.css HTTP/1.1" 200 6946 "-" "Mo
                                zilla/5.0 (X11; Linux x86_64; rv:6.0a1) Gecko/20110421 Firefox/6.0a1"
10:15:22.905  sample_web_logs   73.149.124.79 - - [2018-08-06T14:15:22.905Z] "GET /styles/semantic-ui.css HTTP/1.1" 200 1779 "-" "M
                                ozilla/5.0 (X11; Linux i686) AppleWebKit/534.24 (KHTML, like Gecko) Chrome/11.0.696.50 Safari/534.2
                                4"
10:18:20.636  sample_web_logs   131.73.207.137 - - [2018-08-06T14:18:20.636Z] "GET /beats/metricbeat HTTP/1.1" 200 2392 "-" "Mozill
                                a/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
10:23:10.156  sample_web_logs   26.150.255.63 - - [2018-08-06T14:23:10.156Z] "GET /beats/filebeat/filebeat-6.3.2-linux-x86_64.tar.g
                                z HTTP/1.1" 200 2791 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:6.0a1) Gecko/20110421 Firefox/6.0a1"
```

Detector: high_count

Single time series analysis of count