GSoC 2016: Trend Monitoring Analysis Engine for OWASP AppSensor

*Timothy Sum Hon Mun*

19th March 2016

**Summary:**

Over the recent years, machine learning algorithms have been playing an important role in searching for patterns in data and developing innovative ways to utilize the information to solve many types of problems in the world. AppSensor project defines a conceptual framework that provides a backbone to implementing application intrusion detection and automated response. At the moment, it only supports a basic policy driven analysis and this project aims to implement a machine learning based analysis engine to make AppSensor more robust and be able to analyze a wider variety of problems/events.

**Project:**

AppSensor currently supports a basic policy-driven analysis engine to determine if a series of events represents an attack (if a user triggers 5 of this type of event in 10 minutes, it's an attack). While this supports many use cases, there are times when it would be helpful to know trending information. If a particular function of the application begins to see 10 times its normal amount of traffic, that might represent an attack. This project would add an additional analysis engine to support "trend monitoring".

During the course of the Google Summer of Code, I plan to add the new trend monitoring analysis that will use machine learning. In terms of the new analysis engine, it can be broken down into two key areas: the data storage and the algorithm that will process the data to detect trends.

In terms of data storage, I have two ideas:

i) Extend and improve the existing influxDB implementation in AppSensor to handle querying and processing that will be done by the implemented algorithm.

ii) Using spark and cassandra (as a time series db even though its a k-v store) for data analysis. In relation to appsensor, I would have to implement Spark (probably as part of the analysis engine) for its machine learning library and implement a storage provider for cassandra prior to wiring them together. I will have to design a schema for the time series data storage inside cassandra as well.

<http://www.slideshare.net/patrickmcfadin/apache-cassandra-apache-spark-for-time-series-data>

I will mainly be following through with my first idea and would consider an alternative tool and implementation for data storage if influxDB proves to be hard for the use-case. In terms of algorithm implementation, I will select one problem based on my understanding of detection points and will use the most suitable algorithm for its trend analysis. The overall flow and actions are as such:

i) Problem to tackle: Rate of login attempts / Speed of application use / Change in usage of same transaction for the website.

ii) Algorithm: Random Forest, Decision Trees / Support Vector Machines / K-Nearest Neighbour

iii) Algorithm implementation: using existing machine learning libraries such as JavaML, sparkML or quickml(<http://quickml.org/>) or others with regards to the problem.

iv) Training dataset: Will try to obtain dataset from the community during bonding period.

v) Turn on analysis engine once algorithm has been trained and observe the performance of the algorithm in identifying trend.

Ideally, I will be using the first problem and first algorithm suggested above for the my implementation of the trend monitoring analysis engine.

As part of this project, I will also add additional configuration mechanism that will allow users to decide on specific trending rules and policy. This will be done by extending the configuration mode package, and then creating the respective xml and xsd configuration for the trend Monitoring analysis engine.

For the AppSensor demo application using the new analysis engine, I will be creating a Spring boot application using some ideas off an existing demo application that was created for a talk on AppSensor (<https://github.com/dschadow/ApplicationIntrusionDetection>). I will also be creating user documentation on how to setup and configure the demo application as well as how to use it to try out the new analysis engine.

Besides that, I will be creating the test suite for the overall project to ensure that the project will be well-tested and future contributors can make changes without fear of breaking any existing behaviours which will be detected by failing tests/builds.

**Tentative Timeline Plan:**

*22nd April - 22nd May (Community Bonding Period):*

* Familiarize myself with members of OWASP AppSensor team.
* Work on implementation details (for the algorithm and data storage), get feedback from the community and finalize the project plan where the implementation details are clear and well-defined.
* Gather training dataset from the community that is required to train the algorithm.

*23 May - 12 June*:

* Work on the for time series data storage that will be used for metadata tracking (Implementation, Testing)

*13 June - 3 July (Mid terms in between):*

* Work on the algorithm that will be used for the trend monitoring analysis engine which queries the metadata from the database

(Implementation, Testing)

*4 July - 10 July:*

* Implement the different configuration modes that will handle the choice of trending rules/policy

*11 July - 24 July:*

* Create the Spring boot application that will demo the new analysis engine

(Implementation, Testing)

*25 July - 1 Aug:*

* Testing, bug fixing, cleaning up code (collect feedback from mentor and community)

*2 Aug - 8 Aug:*

* Write user documentation

*8 Aug - 14 Aug:*

* (Buffer time)

*15 Aug - 23 Aug (Final evaluation and hard pencil down):*

* (Buffer time)

**Success Criteria:**

i) A machine learning trend monitoring analysis engine to be used either in place of or in addition to the existing policy-driven analysis engine.

ii) Associated configuration mechanism to specify the trending rules/policy

iii) A small full sample demo application showing usage of the trend monitoring feature.

iv) Creating test suite for the trend monitoring analysis engine and documentation on how to setup and configure it.

**Nice-to-Haves:**

i) Extend the machine learning trend monitoring analysis engine by adding new types of algorithms and monitoring trends of different problems.

ii) Include the feature in (i) into the demo application to demonstrate the use of different algorithms for different problems for the users.

**Do you have other obligations from late May to early August :**

* I have a report to complete as part of my industrial placement which is due on the end of June 2016 but it won’t be any hindrance to my GSoC work as I will be working on this report throughout April and May.

**About Me**

* My name is Timothy Sum and I am from Malaysia. I am currently a final year MSc Computer Science student studying at University of Kent in the UK. I have experience in Java, Javascript, Python, Node.js, MongoDB, AWS, Jenkins, Git workflow, Dropwizard, Logstash, Apache Spark (MSc dissertation) and others, I am always keen to learn new technologies and try things outside my comfort zone!
* I have made efforts to learn about the AppSensor project by following and setting up the demo version as per the guide on GitHub (<https://github.com/jtmelton/appsensor>). I also started reading the code to understanding the overall structure of AppSensor and made a first contribution to the project by submitting a pull-request which was accepted.