Detection and Prevention of Attacks in Sensor Networks

Enzo Lebrun
Universite Claude Bernard
Lyon - France
Email: enzo.the@gmail.com

Gwenael Ambrosino-Ielpo
Universite Claude Bernard
Lyon - France
Email: gwenael.ambrosino@etu.univ-lyon1.fr

***Abstract*—How to handle the variety, the velocity and the volume of data a network of many different kind of sensors deliver?**
How can we verify the trustworthiness of data coming from a sensor?
This article aim to give leads to the answers of those questions, by analyzing data from a camera network. The main goal is to detect and prevent attacks in a sensor. The results shows that the following method detect attacks, it needs to be tested in a larger scale.

Keywords*—*Big Data Analytics, Machine Learning, Image Analysis, Kalman Filter, IoT

I. Introduction

With the increase of IoT applications and the apparition of smart cities, analysis based on sensor networks has become more and more ubiquitous. The needs in this area requires more processing power than a single computer can provide. The amount of data delivered by sensors is a very important source of information used to take accurate decisions. Hence detecting anomalies such as malfunctions or cyberattacks has become a serious issue. Tracking a person on multiple camera is also more and more complex. Thats why its necessary to distribute data on multiple machines. Moreover, the process time must be fast enough in order to detect anomalies in real time. The development of smart cities will require scalable systems to be feed by new sensors and (perhaps) evolving population of data without suffering of a degraded performance.

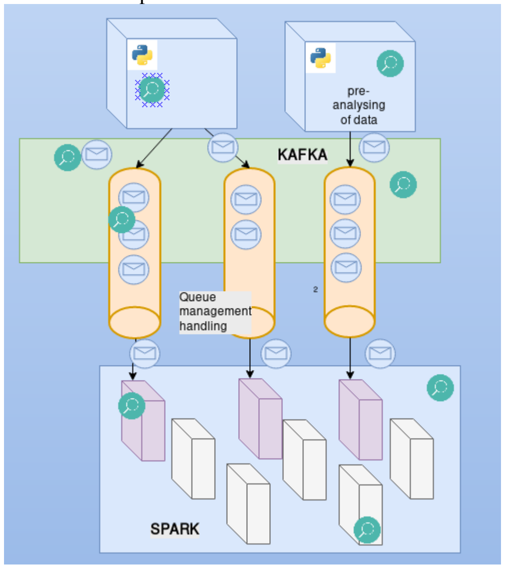
A. Related Work

There is already some anomaly detection in sensor networks in real time (for video surveillance system or meteo,) but the most of them are based on the Apache framework Storm wich possess lesser processing power compared to Apache Spark. In real time analysis, the processing power is crucial so we chosed to use Spark and its Streaming API. [1]

II. Architecture

The architecture of the project will be divided in three part, one pre-analysis section which will receive the flows of data coming from the sensors. The Queue section which take the flows and give them to the last section which will analyze in a distributed system. As

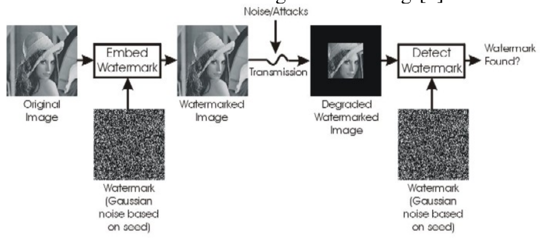
Fig. 1. Structure of the producer/consumer model



A. Anomalies prevention

A first approach to detect attacks is to apply a dynamic watermarking on the data we are sending. Then the fraudulent data which doesnt come from the sensor they pretend to belong, will be detected before any deep analysis.

Fig. 2. Simulation of an attack during watermarking [2]

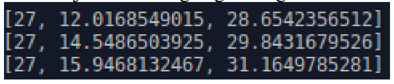

Noise/Attacks
Original Image
Embed Watermark
Watermarked Image
Transmission
Degraded Watermarked Image
Detect Watermark
Watermark Found?
Watermark (Gaussian noise based on seed)
Watermark (Gaussian noise based on seed)

B. Pre-Treatment

The baseline model will be feed with divers type of data collected from many kind of sensors. When the different kind of data are collected, they are pretreated, as an example for a video flow, some data will be extracted from the images such as the number of person, the euclidean distance between the previous image.

Objects tracking: This part of the pre-treatment aims to build a list of all the positions of the people in our videos. The Kalman filter is an algorithm that can predict and follow the position of a given object by computing some parameters of this object, like its position or its velocity, at a moment t. Applied to our videos, we use it to track each person's position with and store it in our database.

Fig. 3. Example of Objects tracking logs using Kalman Filter [3]



```
[27, 12.0168549015, 28.6542356512]
[27, 14.5486503925, 29.8431679526]
[27, 15.9468132467, 31.1649785281]
```

We can see here a sample of the logs that are return by the algorithm, an ID for each person and its coordinates x and y. Furthermore, we obtain coordinates with an accuracy of 10 decimals, which is precise enough to detect if a person already follow this exact path which will allow to detect loop in our videos.

subpopulation:

Considering two populations of Data P and P', from them we can extract many different sub-populations respectively named$P_0, P_1, .., P_n$ and $P'_0, P'_1, .., P'_m$

If there is a sub-population from P and a sub-population from P' that can be treated the same way, then we can merge them in a new population P". So those two sub population will be treated with the same algorithm, this will prevent repetition in the analysis algorithm. For example in the video analysis we use a method in order to detect loop or different kind of attack in the flow. The same method can be use in many other sensors such as temperature sensors, light, etc..

This pre-treatment will reduce the amount of data we send to the model so it prevent the drop of performance, and it also ease data traffic in the network.

C. Queue manager

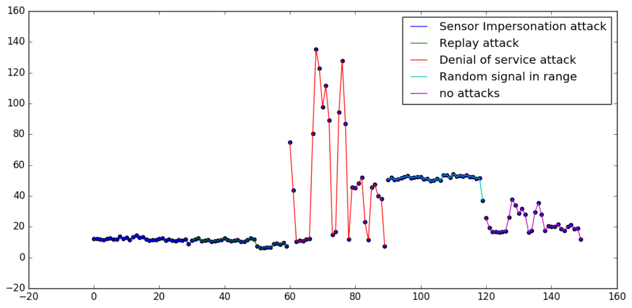
A population P of data will be sent to a queue manager. The Queue handle side effect that the distributed system is not able support. For example, the notion of producer consumer, when the distributed system suffer global drop of performance, the queue will be to able to hold back the flow of data coming from the sensors.

D. Distributed Analysis

A population P is collected from the queue in many Batches. Those batches are spread to every node of the cluster. Data from a Batch B is collect and group into a samples $S_0, ...S_n$ we apply the wavelet transform to the sample and we compute the euclidean distance between the current sample S_i and the previous one S_{i-1} in order to determined the variations on the video. For example, if we take a 1 month sample size we will be able to detect spider webs on video camera by seeing the euclidean distance between two samples.

Detecting attacks:
Considering n samples $S_0, ..., S_{n-1}$ from a population P, if there is a replay attack, with a repetition window of m $\leq (n-1)/2$. The euclidean distance between $S_0, ..S_m$ and $S_m, ..., S_{2m}$ will be 0. The closer we get to this correlation sample/windows, the better we will detect replay attacks. But in a real application case, we never know the size of the window. So we make different size of samples in order to get close to the size of the looping window. The problem is for huge repetition time such as 1 month loop or bigger, it's starting to be very expensive in process power.

Fig. 4. result for different types of attack



III. Conclusion

Real time analysis brings a lot of side effects to be handle by the system, The variety of data is handle by finding similar sub-population between data. The volume is handle by the distributed system and a pre-analysis of the different data it receive. The Velocity is handle by the queue management system that can hold-back the flow of data when the distributed system suffers global drop of performance. This subject hopes to prove that this system is very generic and can be applied to many situation, lots of different methods has been use during pre-analysis before distributing data but we can add deep learning.

Acknowledgment

All this work would not have been possible without the initiative of Dr. Mohand-Said Hacid and Dr. Salim Hariri. And the collaboration between the University of Claude Bernard - Lyon and the University of Arizona -Tucson.
We also thank Jesus Pacheco Ramirez for his help during the project.

References

[1] J. Pacheco, and S. Hariri. "Anomaly behavior analysis for IoT sensors." Transactions on Emerging Telecommunications Technologies (2017).

[2] The figure 2 as been taken from http://www.psiva.ca/