# Detecting IoT Botnet Formation using Data Stream Clustering Algorithms

Gabriel de Carvalho Arimatéa[a] and Admilson de Ribamar Lima Ribeiro[b]

*Federal University of Sergipe, Av. Marechal Rondon, s/n, Aracaju, Brazil*

Keywords: Internet of Things, Botnet, Machine Learning, Security.

Abstract: The Internet of Things has gained much importance nowadays due to its applicability to many ecosystems on day-to-day use. However, these embedded systems have several hardware constraints, and theses device's security has been neglected. Consequently, botnets malwares have taken advantage of poor security schemas on these devices. This paper proposes unsupervised machine learning using data streams to detect the botnet formation on the edge of the network. The results obtained by the algorithm includes an average of 98.43% accuracy and taking about 20.07 ms to evaluate each sample from the stream, making it reliable and fast, even in a more constrained device, such as Raspberry Pi 3 B+.

## 1 INTRODUCTION

The use of a botnet to launch some malicious attacks is one of the main concerns in network security due to its destructive potential. It comes from the spreading and infection capabilities, creating and expanding a ”zombie army” with great versatility, being able to launch different attacks (Kambourakis et al., 2017).

These botnets can be formed by a mix of different types of devices, going from small and more simple embedded systems to a very capable desktop. At first, the main target was the traditional desktop and notebook, since it was the mainstream platform worldwide, making more prominent to big scales scenarios. Botnets such as EarthLink Spammer (2000), Storm (2007), Cutwail (2007), Grum (2008), Kraken (2008), and Mariposa (2008) were the most famous botnets from that time.

In the last years, the target os this infection has shifted to the Internet of Things (IoT) scenarios. This change came due to its ubiquity proposal, its low-level security, and distributed nature, make it easier to launch attacks with the risk of damaging the network and the devices itself.

For its ubiquity proposal and evergrowing raise in daily use, the botnets had shifted interest to those networks. Because the botnets serve as platforms to countless and dangerous attacks, detect its formation is essential to network security of IoT.

Although there are some well-known variants of botnet malware known, there are very few researches considering a dynamic scenario, considering, for example, the evolution of these malwares or new variants. Some traditional strategies, which in general are using supervised machine learning algorithms, are not the best suited since it needs prior knowledge from the attack to insert into the dataset to train the models. Besides, most parts being on the cloud raises the time to respond to threats detected.

For this paper, we propose the use of DenStream, an unsupervised machine learning algorithm, in an edge device (instead of cloud) to detect the formation of botnets, improving the IoT network security. The goal is to detect effectively malicious spam attack released by the malware to attempt to control unprotected devices.

The rest of the paper is organized as follows: Section 2 presents the discussion of related works; Section 3 shows the impacts of an IoT botnet in Internet attacks and its relevance; Section 4 explains why was chosen an unsupervised approach; Section 5 presents the proposed solution; Section 6 presents the methodology, metrics, and experimental evaluations; Section 7 presents the conclusions and future works.

[a] https://orcid.org/0000-0003-1725-5934
[b] https://orcid.org/0000-0003-2010-6024

Arimatéa, G. and Ribeiro, A.
Detecting IoT Botnet Formation using Data Stream Clustering Algorithms.
DOI: 10.5220/0010180903950402
In *Proceedings of the 16th International Conference on Web Information Systems and Technologies (WEBIST 2020)*, pages 395-402
ISBN: 978-989-758-478-7

## 2 RELATED WORK

Based on a systematic review, were verified what machine learning algorithms have been used or what has been done about network security considering a restricted environment (such as Wireless Sensor Network or IoT) and using data stream. It showed almost none works have been made for security reasons.

Many works (Zhao, 2005; Amza et al., 2011; Kapoor and Dhavale, 2016; Kanoun et al., 2016; Roopaei et al., 2017; Axenie et al., 2018; Afghah et al., 2018; Bhattacharyya et al., 2018) did not have concerns about restricted environments such as processing time and memory. Donovan's work (Donovan et al., 2018), although had a near restricted environment (fog computing), did not have any concern on security or used any machine learning approach, making it not very suitable for a dynamic scenario.

Dey's paper (Dey et al., 2016) on determine occupancy in a room using Random Forest and sensors showed great accuracy, was not considered time or resource usage. Considering that it uses the Random Forest algorithm, which generates many combinational decision trees, it can consume many resources from the device.

Li proposal (Li, 2014), based on Local and Global Consistency (or LGC) and Support Vector Machine (or SVM), aimed to classify sensors data stream. The authors show the necessity to previously know the patterns to be able to identify correctly due to its supervised nature. It also stated the high cost due to its constant retraining of the model.

Genetic approach (Schmidt et al., 2014) was also considered, but is slower than algorithms such as SVM and Naive Bayes, and to have good accuracy, it is necessary to increase antibodies, which increase computational cost and memory. It also did not consider restricted environments.

AMWR proposal (Akbar et al., 2015; Akbar et al., 2017), based on the moving window technique to analyze data stream flows, is a supervised algorithm, making it necessary to label all initial data before training. Besides, the authors state that its algorithm could have an almost real-time response, but was not measured response time or computational cost to evaluate its behaviour on restricted environments.

*Vertical Hoeffding Tree* (Kourtellis et al., 2016) used in Elkhoukhi's work (Elkhoukhi et al., 2018) used parallelism. It showed promising results in the experimentation present in Elkhoukhi's paper (Elkhoukhi et al., 2018) in occupation detection, but outside security scope.

Singh (Singh et al., 2013), from a Wireless Sensor Network (WSN) scenario, creates a concept of CluStream on the distributed part and Support Vector Machine (SVM) on the centralized part to detect forest fires. It was used Clustream due to its ability to deal with large data streams for being capable of distributing the data. On the second part, it uses SVM to process the groups formed to predict effectively the existence of fire or not.

P-DenStream (Lu et al., 2018) is a variation of the original DenStream, allowing parallelization and better load and operational cost distribution. This parallelization occurs in the initial part, consisting of the creation and weight of the micro-clusters attribution. Due to DenStream nature, it showed a great candidate to restricted environments, such as embedded systems, where the authors claim a reduction from 10 second from DenStream to 200 milliseconds on P-DenStream.

FlockStream (Spezzano and Vinci, 2015) had conceptual similarities with DenStream, also being a great candidate to restricted environments. It was proposed aiming enormous volume of data from streams but focused on low-memory systems. It is an unsupervised algorithm (not needing external supervision), been tested in a WSN.

It was also considered the paper from Meidan (Meidan et al., 2018), which proposed the usage of autoencoders algorithms that can differentiate benign from malicious data flow within IoT botnet attacks. Autoencoders are used as an anomaly detector, setting the first clusters and use these to evaluate new samples as outliers or normal.

All the papers had its importance in its areas, but to use their approaches to detect botnet formation has some caveats. Were considered these caveats to propose a new approach to network security in the IoT scenario.

Although Spezzano (Spezzano and Vinci, 2015), Lu (Lu et al., 2018) and Singh (Singh et al., 2013) papers had impactful contributions, none of them was used in real restricted environments such as IoT networks, neither considered in a security application. Spezzano (Spezzano and Vinci, 2015) did not demonstrate any experiment or evaluate any characteristic to evaluate this approach in IoT or any other restricted environment.

Although P-DenStream (Lu et al., 2018) and Clustream decentralized part on Singh's paper (Singh et al., 2013) has a distributed nature aiming to divide better the processing load, it did not evaluate memory and processing usage. Also, like Spezzano's (Spezzano and Vinci, 2015), there was no consideration in restricted environments and did not considerate security.

The work proposed by Meidan (Meidan et al.,

2018) is used on an IoT network and had considered security application, but using a neural network algorithm, which makes a not so lightweight approach and not so fast alternative, been a great candidate to a cloud application. But this type of application been in the cloud makes this approach more susceptible to latency and communication issues such as packet loss or not having an Internet connection at all.

Based on those points, this work aims to use DenStream as a lightweight unsupervised machine learning algorithm to create cluster groups of normal behaviour and then act as an anomaly detection algorithm (similar to Meidan's approach (Meidan et al., 2018)). These anomalies are considered as malicious traffic, allowing dynamic response to threats. Besides, for being a lightweight data stream processing algorithm, it consumes fewer resources and gives faster responses, enabling many low costs single boards computers to run it effectively.

## 3 IoT BOTNET IN INTERNET ATTACKS

In IoT, thanks to its ubiquity, if many devices were contaminated and taken control, it forms a botnet network, which had great potential to launch devastating attacks. According to Dietz et al. (Dietz et al., 2018), botnets are networks of devices infected with malware, allowing a malicious actor to control them remotely. Especially in IoT, due to low-security schemes and not proper configuration on installation, many devices are not well protected, serving as an entrance to contamination. The steps to forming a botnet and launch of the attacks (which can which are seen in Figure 1) are (Dietz et al., 2018):

1. Scan open ports;
2. Brute force attacks to gain access to IoT devices;
3. End possible concurrents to device's control;
4. Connect the device to Command and Control (C&C);
5. Download and execution of malicious scripts;
6. Spread and attack other vulnerable devices;
7. Launching attack from C&C.

There are attacks specialized in mounting these botnets. Bashlite (Marzano et al., 2018) was one of the firsts botnets to aim to control IoT devices. The dispersion starts with the initial device trying to connect with a public IP, with a Telnet scan. Then, tries to authenticate on found devices using the most common ports and credentials. Once successful, that device will serve as a source to distribute requests on the next attack. But, due to its simplicity, it demands more effort to set up the malware and C&C form the attacker.

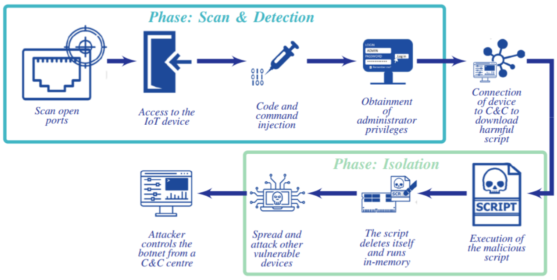
Phase: Scan & Detection
Scan open ports
Access to the IoT device
Code and command injection
Obtainment of administrator privileges
Connection of device to C&C to download harmful script
Phase: Isolation
Execution of the malicious script
SCRIPT
The script deletes itself and runs in-memory
Spread and attack other vulnerable devices
Attacker controls the botnet from a C&C centre

Figure 1: Botnet life cycle (Dietz et al., 2018).

The Mirai was discovered in August 2016. It creates botnets network for DDoS attacks from DVRs, WebIP cameras and low-security Linux servers (Kambourakis et al., 2017). It has the same structure as Bashlite. The few differences are the Mirai capability which is not present in Bashlite (only available through extensions) such as tools for contaminated devices search for more vulnerable devices; DNS use for attacks; binary protocol to send messages to difficult its discovery. The process and agents involved in a Mirai attack can are seen in Figure 2.

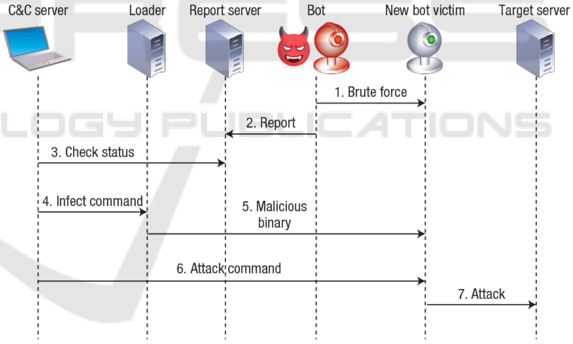

Figure 2: Steps and agents involved on Mirai attack (Kolias et al., 2017).

After a deployed botnet, a plethora of attacks can be launched from it, serving as platforms to others attacks, such as spam advertised pharmaceuticals, robbing bank credentials and advertisement, click fraud and distributed denial of service (DDoS) (Putman et al., 2018). Kanich et al (Kanich et al., 2011) wrote about a spam advertised pharmaceuticals exploit, where according to his article, a botnet can earn almost $3.5M per year. These attack spam emails about selling counterfeit medicine, where Viagra and diet pills been the most common.

Click fraud is also another known use of a botnet. After the botnet deployed, it starts to fake clicking advertisement, allowing the host to profit on those counterfeited clicks. Although its simplicity, initiative such as WhiteOps stated that a group based in Russia profits ”$3 to $5 million in counterfeit inventory per day by targeting the premium video advertising ecosystem”.

DDoS are attacks to deplete the resources from a server, leading to interruption of service. According to Marzano et al (Marzano et al., 2018), this attack aims at servers or network devices. Known famous attacks registered were at Krebs on Security (reaching 620Gb/s) and Ars Technica (reaching 1Tb/s).

# 4 UNSUPERVISED X SUPERVISED MACHINE LEARNING

What differentiates an unsupervised from a supervised machine learning algorithm is the need from the previous classification of data for training the algorithm. The unsupervised approach does not need previous categorization, grouping data with similar characteristics.

This type of algorithm is better suited in scenarios with no or little knowledge of the groups in the dataset. On IoT botnet scenarios, where the infections are always adapting and evolving, and zero-day attack makes a supervised approach not so reactive. Therefore was chosen an unsupervised algorithm for this approach.

But a downside that comes with unsupervised algorithms it that they are not so accurate and more susceptible to outliers than a supervised algorithm. Besides, in an unsupervised approach, the most features collected to the algorithm, the better. But associated with that increases processing time and resources. The use of preprocessing techniques to overcome these difficulties are used on this approach as well to reduce or eliminate outliers on training data and identify the most valuable features.

# 5 PROPOSED SOLUTION

## 5.1 DenStream

It was selected to this approach an unsupervised algorithm due to no prior knowledge and more evolving capability. It was also considered algorithms that can use data-stream as input. These characteristics were chosen because the massive data flow in IoT, with evergrowing perspective, which can be difficult and almost impossible to handle all data. The data stream as input make the algorithms more performatic and causes small footprint, which makes them lightweight and fast.

The DenStream algorithm proposed by Cao et al’s paper (Cao et al., 2006) as a specialized data mining unsupervised clustering machine learning algorithm, to adapt to evolving streams with low memory cost. These characteristics were the reasons to be chosen to the solution proposed.

The algorithm has two distinct parts: offline and online phases. A DBScan (Ester et al., 1996) it is performed on the offline part to discover the first clusters. These represent the common and benign behaviour of the network. The parameters needed are:

- epsilon: The maximum distance between two points for being considered as similar;
- minPoints: The minimal quantity of similar points to create a cluster to these points.

The Ozkok’s paper (Ozkok, 2017) proposed an improvement to DBScan, estimating the best value to epsilon due to minimal points passed as parameter. It made this decision automatic by the algorithm, being dependent only by one parameter. This improvement was applied to facilitate the adjustment of its parameters. To its offline phase, it was used 100 benign samples from each device to extract the PCA parameters and the epsilon value.

After the first clusters formed by the offline phase, the algorithm receives new samples, trying to insert in any existing groups. If it is not possible, it will be labelled as an outlier and stored for a while. If other samples become more similar to an outlier, it creates outliers clusters, adapting to new scenarios. The code was written in Python also using scikit-learn package and its available in a public repository [1].

To improve the overall accuracy of this proposal, it was also applied a preprocessing phase prior to its offline and online phases.

## 5.2 Preprocessing Phase

For preprocessing it was applied Local Outlier Factor (LOF) as preprocessing algorithm to reduce noise in the benign traffic contained in the dataset.

The LOF was proposed in 2000 by Breunig et al (Breunig et al., 2000), using concepts of ”core distance” and ”reachability distance”, present on DBScan and DenStream as well, to determine outliers. This algorithm tries to eliminate noise on the benign dataset, allowing a better characterization of the data

[1] https://github.com/gabriel-arimatea/unsupervised-ml-botnet

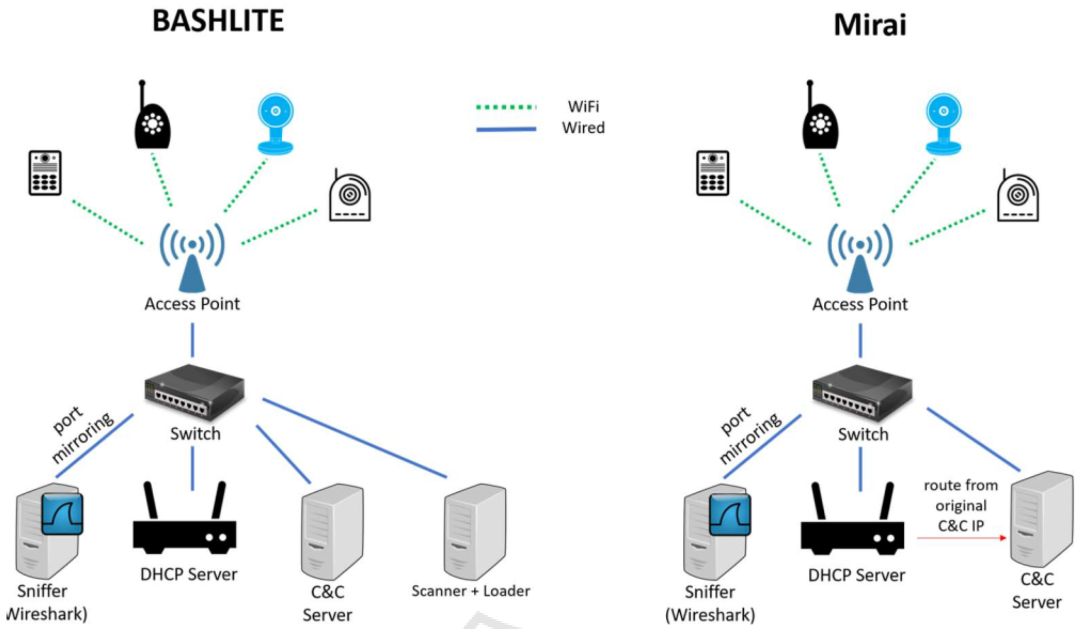

Figure 3: Architecture proposed by Meidan et al. 2018.

at the training phase. But some outliers can be misjudged by the algorithm due to those noises removed by the LOF.

Principal Component Analysis (PCA) was also applied to reduce dimensionality to make the algorithm more efficient and reduce processing time. The PCA algorithm was proposed in 1901 by Pearson, where it receives some correlated variables and transforms it into linearly uncorrelated variables. It creates fewer variables with more value to the algorithms, eliminating redundancy by the correlation.

### 5.3 Architecture

Using the dataset made available from Meidan's paper (Meidan et al., 2018), were developed an hypothetical architecture (seen on Figure 4) based on Meidan's architecture, seen on Figure 3. The proposed architecture made the use of the dataset possible.

This architecture uses a message broker in publish/subscribe to store the data stream until it processed by the device. It ensures that every data stream will be processed accordingly.

## 6 EXPERIMENTAL EVALUATION

### 6.1 Dataset

The dataset used was proposed in Meidan's paper (Meidan et al., 2018). On his work, his architecture showed in Figure 3 infected with two major botnets variants: Mirai and Bashlite. The dataset generated has benign traffic and five types of attacks executed from the botnets. The spam attack used to infect other devices was the only one considered from these attacks, due to be the only attack used to form the botnet network.

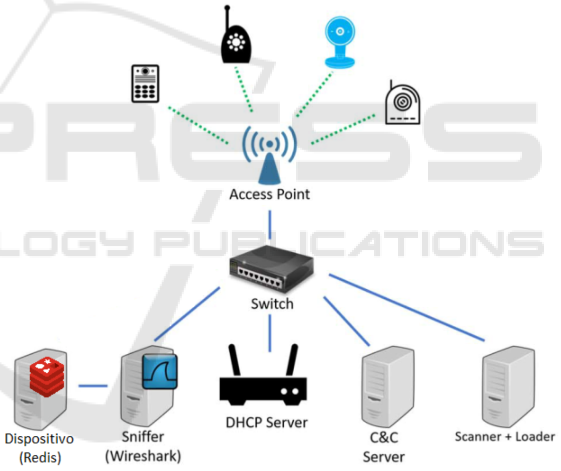

Figure 4: Proposed Architecture architecture.

The dataset was created using four characteristics, aggregated by different sources about the stream packets, shown in Table 1. There were applied some statistics, generating 23 features. Those create 115 characteristics, considering the decay factor of a damped window.

Table 1: Meidan's dataset features.

| Characteristics | Statistic | Aggregated by | Number of Features |
|---|---|---|---|
| Outbound Packet size | Mean, Variance | Source IP and MAC-IP, Channel, Socket | 8 |
| Packet count | Number | Source IP and MAC-IP, Channel, Socket | 4 |
| Packet jitter | Mean, Variance, Number | Channel | 3 |
| All Packet size | Magnitude, Radius, Covariance, Correlation coefficient | Channel, Socket | 8 |

## 6.2 Metrics

It was supplied random samples from benign and malicious spam attack data to the generated model to evaluate its capability, and some characteristics were analyzed. In this scenario, the objective would be to block all the malicious data (labelled as a negative) and pass all benign data (marked as a positive). Considering these labels, metrics such as true positives and negatives ratings are essentials.

Considering these metrics and false positives and negatives ratings (which are complementary), the confusion matrix was chosen, generating more metrics, such as accuracy. Also, considering how fast has to be a response to stop propagation, time to create the first clusters and to analyze each new sample is essential. The metrics used to evaluate the results are:

- True Negative Rate (TNR): benign data classified as benign;
- True Positive Rate (TPR): malicious data classified as malicious;
- False Negative Rate (FNR): malicious data classified as benign;
- False Positive Rate (FPR): benign data classified as malicious;
- Accuracy;
- Offline training time: time used to train the model;
- Online classification time: time to analyse a new data.

The algorithm was ported to a Raspberry Pi 3 B+, running the algorithm and a Redis instance in a publish/subscribe configuration to store and serve the data collected by the Wireshark Sniffer showed on Figure 4.

To evaluate were made 40 rounds with random data from the dataset, using all nine devices, with 100 random samples for each device for offline training and a balanced dataset to test the final model generated by the algorithm.

## 6.3 Results

The confusion matrix generated by the sum of all executions can be seen in Table 2. The accuracy of each run is shown in Figure 5. The results are in range:

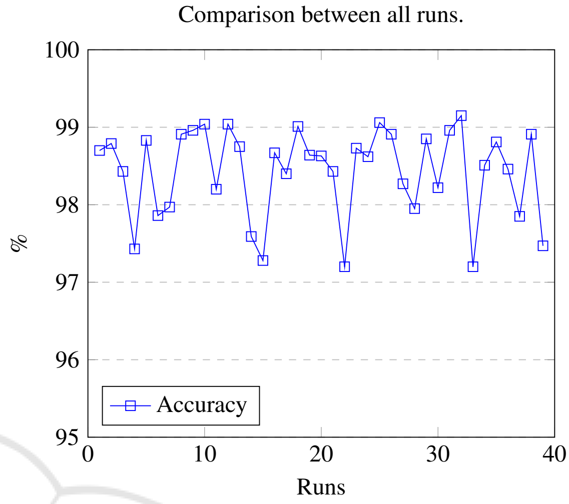

Figure 5: Accuracy between runs.

- TNR: Between 97.52% and 100% (mean of 99%);
- TPR: Between 94.41% and 99.87% (mean of 97.85%);
- FNR: Between 5.59% and 0.13% (mean of 2.15%);
- FPR: Between 0% and 2.48% (mean of 1%);
- Accuracy: Between 97.2% and 99.15% (mean of 98.43% - Figure 5);
- Offline training time: mean of 45 ms;
- Online classification time: Between 17.99 and 23.96 ms (mean 20.07 ms).

Table 2: Confusion Matrix considering all rounds.

| | | Original Classfication | |
|---|---|---|---|
| | | Benign | Malicious |
| Prediction | Benign | 337,303 | 7,315 |
| | Malicious | 3,401 | 333,623 |

To the autoencoder proposed by Meidan (Meidan et al., 2018), the metrics cited by his paper are true positives rating, false positives rating and time to detect the infection. The table comparing the two results can be seen in Table 3.

The proposed algorithm had less accuracy than Meidan's autoencoder but still have a high percentage. But in comparison, the DenStream is much

Table 3: Comparison between algorithms.

| Metrics | (Meidan et al., 2018) | DenStream | |
|---|---|---|---|
| | | Mean | Variance |
| TPR | 100% | 97.85% | 94.43% ∼ 99.87% |
| FPR | 0.007% ∼ 0.01% | 1% | 0.01% ∼ 2.48% |
| Time | 174 ∼ 212 ms | 20.07 ms | 17.99 ∼ 23.96 ms |

faster, taking almost 90% less time to differentiate benign and malicious data. Since autoencoder is a neural network, it has a much costly footprint than DenStream, and need much more data to train also.

## 7 CONCLUSION

In this paper was showed that more lightweight algorithms, such as DenStream, can be a great candidate to detect botnet formation, making possible to run this algorithm in more simple and low-cost devices, such as a Raspberry Pi 3B+ (used in the experiment). It also showed that, due to its light and efficient way of dealing with training and predicting, it could respond to a threat much sooner.

In this paper was used DenStream as an unsupervised machine learning algorithm, but the CluStream showed as an option as well. As future work, it will be tested using the CluStream and will be verified which one is more effective to the problem.

It will also be studied applications for the algorithm, which can be ported to an IoT specialist device or inserted in an SDN context. For this, an analysis of minimum hardware requirements to perform well had to be made. It will also be studied possibles measures to apply when the algorithm detects an attack.

## REFERENCES

Afghah, F., Cambou, B., Abedini, M., and Zeadally, S. (2018). A ReRAM Physically Unclonable Function ( ReRAM PUF )- Based Approach to Enhance Authentication Security in Software Defined Wireless Networks. *International Journal of Wireless Information Networks*, 25(2):117–129.

Akbar, A., Carrez, F., Moessner, K., and Zoha, A. (2015). Predicting complex events for pro-active iot applications. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pages 327–332.

Akbar, A., Khan, A., Carrez, F., and Moessner, K. (2017). Predictive analytics for complex IoT data streams. *IEEE Internet of Things Journal*, 4(5):1571–1582.

Amza, C., Leordeanu, C., and Cristea, V. (2011). Hybrid network intrusion detection. In *2011 IEEE 7th International Conference on Intelligent Computer Communication and Processing*, pages 503–510.

Axenie, C., Tudoran, R., Bortoli, S., Al Hajj Hassan, M., Foroni, D., and Brasche, G. (2018). Starlord: Sliding window temporal accumulate-retract learning for online reasoning on datastreams. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1115–1122.

Bhattacharyya, S., Katramatos, D., and Yoo, S. (2018). Why wait? let us start computing while the data is still on the wire. *Future Generation Computer Systems*, 89:563–574.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). LOF. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data - SIGMOD '00*. ACM Press.

Cao, F., Estert, M., Qian, W., and Zhou, A. (2006). Density-based clustering over an evolving data stream with noise. In *Proceedings of the 2006 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics.

Dey, A., Ling, X., Syed, A., Zheng, Y., Landowski, B., Anderson, D., Stuart, K., and Tolentino, M. E. (2016). Namatad: Inferring occupancy from building sensors using machine learning. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. IEEE.

Dietz, C., Castro, R. L., Steinberger, J., Wilczak, C., Antzek, M., Sperotto, A., and Pras, A. (2018). IoT-botnet detection and isolation by access routers. In *2018 9th International Conference on the Network of the Future (NOF)*. IEEE.

Donovan, P. O., Gallagher, C., Bruton, K., and Sullivan, D. T. J. O. (2018). A fog computing industrial cyber-physical system for embedded low-latency machine learning industry 4.0 applications. *Manufacturing Letters*, 15:139–142.

Elkhoukhi, H., NaitMalek, Y., Berouine, A., Bakhouya, M., Elouadghiri, D., and Essaaidi, M. (2018). Towards a real-time occupancy detection approach for smart buildings. *Procedia Computer Science*, 134:114–120.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press.

Kambourakis, G., Kolias, C., and Stavrou, A. (2017). The mirai botnet and the IoT zombie armies. In *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*. IEEE.

Kanich, C., Weavery, N., McCoy, D., Halvorson, T., Kreibichy, C., Levchenko, K., Paxson, V., Voelker, G. M., and Savage, S. (2011). Show me the money: Characterizing spam-advertised revenue. In *Proceedings of the 20th USENIX Conference on Security*, SEC'11, page 15, USA. USENIX Association.

Kanoun, K., Tekin, C., Atienza, D., and v. d. Schaar, M. (2016). Big-data streaming applications scheduling based on staged multi-armed bandits. *IEEE Transactions on Computers*, 65(12):3591–3605.

Kapoor, A. and Dhavale, S. (2016). Control flow graph based multiclass malware detection using bi-normal separation. *Defence Science Journal*, 66(2):138.

Kolias, C., Kambourakis, G., Stavrou, A., and Voas, J. (2017). Ddos in the iot: Mirai and other botnets. *Computer*, 50:80–84.

Kourtellis, N., Morales, G. D. F., Bifet, A., and Murdopo, A. (2016). VHT: Vertical hoeffding tree. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 915–922. IEEE.

Li, F. (2014). A pattern query strategy based on semi-supervised machine learning in distributed WSNs. *Journal of Information and Computational Science*, 11(18):6447–6459.

Lu, J., Feng, J., Zhang, J., Xia, P., and Xiao, X. (2018). A parallel approach on clustering traffic data stream based on the density. In *2018 Sixth International Conference on Advanced Cloud and Big Data (CBD)*. IEEE.

Marzano, A., Alexander, D., Fonseca, O., Fazzion, E., Hoepers, C., Steding-Jessen, K., Chaves, M. H. P. C., Cunha, I., Guedes, D., and Meira, W. (2018). The evolution of bashlite and mirai IoT botnets. In *2018 IEEE Symposium on Computers and Communications (ISCC)*. IEEE.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Shabtai, A., and Elovici, Y. (2018). N-baiot: Network-based detection of iot botnet attacks using deep autoencoders. *CoRR*, abs/1805.03409.

Ozkok, F. O. (2017). A new approach to determine eps parameter of DBSCAN algorithm. *International Journal of Intelligent Systems and Applications in Engineering*, 4(5):247–251.

Putman, C. G. J., Abhishta, and Nieuwenhuis, L. J. M. (2018). Business model of a botnet. *CoRR*, abs/1804.10848.

Roopaei, M., Rad, P., and Jamshidi, M. (2017). Deep learning control for complex and large scale cloud systems. *Intelligent Automation & Soft Computing*, 23(3):389–391.

Schmidt, B., Kountanis, D., and Al-Fuqaha, A. (2014). Artificial immune system inspired algorithm for flow-based internet traffic classification. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*. IEEE.

Singh, Y., Saha, S., Chugh, U., and Gupta, C. (2013). Distributed event detection in wireless sensor networks for forest fires. In *2013 UKSim 15th International Conference on Computer Modelling and Simulation*. IEEE.

Spezzano, G. and Vinci, A. (2015). Pattern detection in cyber-physical systems. *Procedia Computer Science*, 52:1016–1021.

Zhao, Q. (2005). Learning with data streams – an NNTree based approach. In *Embedded and Ubiquitous Computing – EUC 2005 Workshops*, volume 3823, pages 519–528. Springer Berlin Heidelberg.