Journal of Internet Services and Applications, 2022, 12:1, doi: 10.5753/jisa.2022.2327

Detecting Attacks and Locating Malicious Devices Using Unmanned Air Vehicles and Machine Learning

Evilasio C. Junior [**Universidade Federal do Ceará** | ***evilasiojunior@great.ufc.br***]
Wanderson L. Costa [**Universidade Estadual do Ceará** | ***wanderson.leonardo@ifpi.edu.br***]
Ariel L. C. Portela [**Universidade Estadual do Ceará** | ***ariel.portela@aluno.uece.br***]
Leonardo S. Rocha [**Universidade Estadual do Ceará** | ***leonardo.sampaio@uece.br***]
Rafael L. Gomes [**Universidade Estadual do Ceará** | ***rafa.lopes@uece.br***]
Rossana M. C. Andrade [**Universidade Federal do Ceará** | ***rossana@ufc.br***]

Received 30 November 2021 • **Accepted** 18 March 2022 • **Published** 22 September 2022

Abstract
Internet access in both private and public environments allows users to broadly access their data what makes possible the deployment of new services based on Internet of Things. This fact created Smart Environments (SEs) that are composed of a huge amount of heterogeneous devices, for example, personal devices (smartphones, notebooks, tablets, etc) and IoT devices (sensors, actuators, and others). However, these environments can facilitate the action of malicious agents interested in promoting Distributed Denial of Service (DDoS) attacks to the network, and, when they are public places, it is challenging to locate these attackers. In this way, it is necessary to deploy solutions that can detect DDoS in SEs and to determine the physical location of the attacker, which is essential to prevent future attacks. Within this context, this article presents an Intelligent System for detection of DDoS and physical location of devices in SEs, applying Machine Learning (ML) and trilateration techniques. The experiments performed, using real network traffic and simulation, suggest that the proposed system is capable of detecting attacks and finding malicious devices.

Keywords: Location, Detection, Machine Learning, Unmanned Aerial Vehicles.

1 Introduction

The ease of accessing Internet over public wireless networks allows people to access their data online from anywhere, what makes possible the provision of services and the deployment of smart cities (Chamoso *et al*., 2018). This kind of environment has been called Smart Environments (SEs), which can be implemented in several contexts: Smart Campus, Smart Homes, Smart Cities, Industry 4.0, Smart Hospitals, etc. (Doshi *et al*., 2018).

The presence of Internet of Things (IoT) devices in these public environments brought several security issues. One important category of attacks in these environments is Distributed Denial of Service (DDoS), where an attacker targets the availability of the servers by flooding the communication channel with requests coming from distributed devices (Vishwakarma and Jain, 2020). The DDoS attacks come from numerous security vulnerabilities in the devices (Andrea *et al*., 2015; Diro and Chilamkurti, 2018) that directly affect Quality of Service (QoS) and Quality of Experience (QoE). As a result, in the last few years, several cyber-attacks were performed in the Internet through the infection of both personal and IoT devices (Brun *et al*., 2018).

A suitable approach for DDoS detection is the usage of Machine Learning (ML) techniques, which understand the network flows and progressively improve the understanding of them (Doshi *et al*., 2018). Nevertheless, it is necessary to use the most relevant network traffic characteristics to train the DDoS attack detection mechanism using ML, since the consideration of unsuitable characteristics harms the accuracy of ML techniques. Once the attack is detected, it is possible to identify information such as the MAC address and the signal strength between the access point and the malicious device (Alotaibi and Elleithy, 2016). Then, the malicious device can be blocked, preventing the attacking machine's access to the network. Additionally, it is important to identify the location of these malicious devices, which is important to prevent further attacks (Halder and Ghosal, 2016).

One way to physically locate devices in public environments is to use unmanned aerial vehicles (UAV) such as drones. According to (Mozaffari *et al*., 2019), these vehicles are advantageous when used in solutions for public wireless network environments due to their characteristics, such as mobility, path flexibility, and adaptive altitude. Additionally, drones can locate targets using only digital signals (Halder and Ghosal, 2016).

Within this context, this article proposes a system to detect DDoS attacks and to locate immobile devices used to attack a network, using a drone and the information obtained from the attack detection module. Both attack detection and location processes work together to improve the security level of SEs.

The attack detection module is based on network monitoring with a trained ML model. The detection module starts with a labeled dataset, with traffic corresponding to attacks being distinguished from normal traffic. Then, starting with 80 important network flow features for DDoS from (Sharafaldin *et al*., 2019), the module performs features selection and training the ML model, which is then used for detecting DDoS attacks. The training phase identifies the main features for detecting DDoS in SEs, and several ML algorithms are used to train the model for detecting DDoS attacks.

With the information of the attacker's signature and the

Table 1. Location Process Related Work

Related Work	Calculates distance based on signal strength	Uses Access Point metering	Types of environments covered	Prior knowledge of network elements required	Target type
Halder (2016)	Yes	No	Indoor/Outdoor	Locating sensors data	Locating sensors
Sorbelli (2018)	Yes	No	Indoor/Outdoor	Locating sensors data	Locating sensors
Acuma (2017)	Yes	No	Indoor/Outdoor	Not need	Lost devices
Sun (2018)	Yes	No	Indoor/Outdoor	Not need	Lost devices
Nobles (2011)	Yes	Yes	Only Indoor	Location of access points	Network attacker

network signal strength obtained at the final of the detection process, the location process seeks to find the distance from the drone to the target and from the target to the access point used by the attacking device. The attacker signature and the network signal strength, along with the location of the access point and the drone, are used in the trilateration calculation to determine the approximate location of the attacking machine.

The detection experiments, using a dataset of real network traffic with DDoS attacks, indicate that the proposal reaches 99% of accuracy when the most suitable features are selected. Regarding the location experiments, we executed simulations using Network Simulator 3 (NS-3) (Riley and Henderson, 2010), suggesting that the proposal was able to the approximate position of the target with a time of ten seconds in all the situations evaluated.

Thus, the main contribution of this paper is the investigation of the performance and viability of using combined ML techniques and algorithms for the detection and location of an attacker in a Smart Environment. In particular, it is evaluated how fast and accurately it is possible to detect and locate devices performing DDoS attacks.

The remainder of this article proceeds as follows. Section 2 presents related work. Section 3 describes the proposed system. Section 4 details the performance evaluation and results. Finally, Section 5 concludes the article.

2 Related Work

This section summarizes related works on cybersecurity and DDoS detection and physical location of the attacking device. We initially describe the existing proposals related to location, and, later, we detail the solution for attacks detection.

Nobles *et al*. (2011) present a strategy for tracking a network attacker in an indoor environment. This strategy uses wireless signal strength to calculate the distance between the target and the various access points. This distance is then used to locate the target using trilateration.

Halder and Ghosal (2016) present different strategies for locating sensors in a wireless sensor network. These strategies use trilateration and moving and static targets in open environments. Aligned to the previous study, Betti Sorbelli *et al*. (2018) propose two algorithms for locating static targets in open environments, using directional and omnidirectional antennas. This work compares the results of the two proposed algorithms through a simulation made in the MATLAB programming language.

Acuna *et al*. (2017) propose a solution that identifies the wifi signal of smartphones captured by the network and a drone that captures this signal from the network, maps the environment into zones based on the received signals, and applies a Random Forest algorithm to find the zone where the target is. Sun *et al*. (2018) uses a similar idea, but seeks to filter the zones using new drone measurements, trilateration, and Kalman filters, thereby increasing the accuracy of the found location.

Table 1 summarizes the main characteristics of the aforementioned related works. We can observe that the studies by Halder and Ghosal (2016) and Betti Sorbelli *et al*. (2018) focus on locating target nodes in wireless sensor networks, which communicate with the drone. On the other hand, Acuna *et al*. (2017) and Sun *et al*. (2018) search for devices capturing signals from the wifi network to map specific zones. Our work distinguishes from them since we propose a technique to locate attacking devices in indoor or outdoor smart environments using the network information obtained by our attacker detection process. We trigger the drone, which uses the information provided by the detection process to find the target. In addition, we consider information from the access point used by the target device, which decreases the number of measurements the drone needs to take.

Regarding the works related to the detection of attacks, several proposals use Artificial Intelligence (AI) techniques. Vinayakumar *et al*. (2020) propose a botnet detection system based on a two-tier ML structure to semantically distinguish botnets from legitimate behavior in the application layer of DNS domain name system services. In the first level, scores are used to define the similarity. When reaching a difference established by the authors, the domain name is passed to the second level that uses a deep learning architecture to detect and classify DDoS occurrences. This work focuses on detecting DDoS exclusively on DNS servers, preventing its application in other types of IoT network services.

Sharafaldin *et al*. (2019) present a study on the traffic characteristics of the most important networks for detecting different types of DDoS attacks on traditional networks, that is, TCP/IP networks. Two networks with traditional computers were designed and implanted in the carried out experiments. In short, the behavior extracted from the dataset samples becomes different compared to networks designed with IoT devices. The behavior of IoT networks communicates with a small finite set of endpoints. It is prone to have repetitive network traffic patterns (small packages at fixed time intervals for registration purposes, for example).

Yamauchi *et al*. (2019) describes a model for detecting anomalous operations of IoT devices in smart homes (SHs) based on user behavior. The model learns the sequence of activities performed by the hour of the day and then compares the current sequence with the sequences learned for the condition corresponding to the current condition. If it has any predefined changes, the method classifies the operation as an IoT device anomaly. Thus, this model proposed by the authors is limited to understanding SHs.

Table 2 summarizes the characteristics of the detection process from the related works.

Table 2. Detection Process Related Work

Related Work	Problem	Approach
Vinayakumar (2020)	DDoS em DNS	Two-way ML botnet detection layers
Sharafaldin (2019)	DDoS em TCP/IP	Testbed experiments for feature analysis
Yamauchi (2019)	Anomaly Detection in SHs	Identification of anomalies based on device behavior

To the best of our knowledge, there is no proposal in the literature focused on the study of the performance and viability of a system to detect DDoS in heterogeneous smart environments integrating this detection with a location process to identify the physical location of the attacking devices. The results obtained by the system are encouraging and evolve the state of the art regarding security and network management in smart environments.

3 Proposal

This section details the proposed system to detect DDoS attacks and locate immobile devices used to attack a network, using a drone and the information obtained from the detection process. Thus, the proposed solution has two main modules: Detection and Location.

The detection module encompasses the task of features extraction, feature selection, and ML training, where, in the end, the trained model is able to perform network monitoring and detection of DDoS attacks. When an attack is detected, the detection module passes the information about the malicious devices (MAC Address) to the module Location, which executes the tasks of getting signal strength in the AP, defining the drone flight plan, and calculates the location using trilateration. An overview of the organization of the proposed system is illustrated in Figure 1.

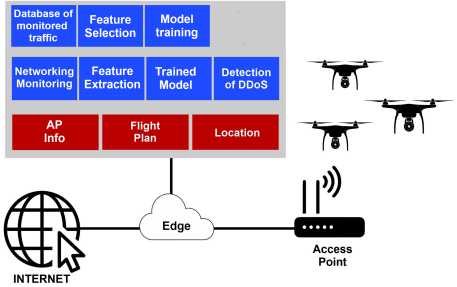

Figure 1. Overview of the Proposed System

In Figure 1, the modules in blue represent the functionalities of the detection process, while the modules in red illustrate the steps performed during the location process. The detection process monitors the network to collect data to detect possible attacks. When an attack is identified, the location process is triggered to physically locate the malicious device using a UAV.

Next, the detection process and the location process are detailed in Subsections 3.1 and 3.2, respectively.

3.1 Detection Process

The detection module performs the following steps: Database Construction; Selection of Features; Training of Model and Detection of DDoS; Network Monitoring; Feature Extraction and Detection of DDoS Attacks use the trained model.

Before the network monitoring, we need to train the ML model that is used to identify the DDoS attack. At the beginning, we need a labeled dataset with network traffic containing normal (benign) traffic and traffic corresponding to DDoS attacks. From this raw data in the database, we extract 80 features that are important for DDoS detection, following the work from Sharafaldin *et al*. (2019). We then perform automatic feature selection techniques that choose the features to be used as input to training a ML model. After training, the generated classifier acts to detect DDoS attacks. Various techniques can be used during the features selection and model training phases. We tested different combinations of techniques, as described in the next subsections.

With the trained ML model, it is possible to monitor the network and detect new DDoS attacks. When the DDoS attack is detected, the information is logged, and the localization process is started.

3.1.1 Selection of Features

All the information extracted from de database represents some aspect of IoT Networks. However, the use of many features will result in certain noises that may interfere with the process of the ML model. In addition, noise affects the functioning of ML techniques unevenly. That is, a certain noise may or may not affect another technique. Therefore, it is important to select the most relevant characteristics to achieve the best performance of the classifiers used in DDoS detection.

The features selection regarding network traffic analysis is a challenging process, since it becomes even more complex when it comes to DDoS attacks due to the variety of types and the complexity of its action timing. The selection purpose of characteristics is to enable the construction of ML models that make it possible to understand the data and maximize the detection capacity. Therefore, the selection of characteristics helps to know irrelevant and redundant attributes that can have a negative impact on the model performance, decreasing the accuracy of the model. Another important issue is the computational resources, since reducing the number of features brings important benefits to this issue. Less data means reduced training time, less misleading data that improve model performance, faster processing, less memory

consumption, easier data extraction, less storage space, and, mainly, dimensionality reduction. In this way, the usage of most suitable features about the network traffic enables the optimization of the time for training and detection of these ML models.

Based on these facts, in this research, we analyze the following techniques for selecting characteristics: (1) Maximum Relevance Minimum Redundancy (mRMR) (Peng *et al*., 2005), uses Fisher's test scores and Pearson's correlation; (2) Low Variance (LV), removes all characteristics whose variation does not reach a certain limit; (3) Extra-Tree (EA) (Geurts *et al*., 2006), builds a set of the non-pruned decision or regression trees according to the classic top-down procedure; (4) SVC (Chang and Lin, 2011), a linear model that estimates sparse coefficients based on important characteristics; and, (5) Lasso (Friedman *et al*., 2010), a linear model that estimates sparse coefficients.

These techniques can be used to select the DDoS data set's relevant characteristics that improve the model's performance. Nevertheless, the different strategies applied by them (filter methods, wrapping methods, or embedded methods) lead to different selected characteristics (Kaushik, 2016).

In this way, we present in Section 4 a study to define the most suitable selection technique that improves the accuracy of the existing ML models to detect DDoS in SEs. Additionally, timing issues are evaluated: Training time and detection time. This timing evaluation allows the identification of the tuple selection technique and the faster ML model, allowing its application in time-constrained scenarios.

3.1.2 Model Training and Detection of DDoS Attacks

After processing the processed data, the Knowledge Dataset is fed, and it is used as the basis for the ML training. ML training encompasses the input of the data in the Knowledge Dataset and the execution of the ML technique. Later, the detector (ML model trained) is used to identify possible DDoS attacks.

The DDoS attack behavior is not defined but rather learned from the training dataset, whose data points are labeled "normal" or "attack". Depending on the quality and diversity of the dataset characteristics, the system will correctly detect when an attack occurs.

The proposed system was designed to enable the usage of any ML technique. This Independence allows the proposed system to execute the most suitable ML technique in the training stage. Thus, we evaluated the ML techniques that have distinct singularities: K-Nearest Neighbor (KNN), Naive Bayes (NB), Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), and Support Vector Machines (SVM).

The defined process flow is repeated constantly to keep the detector updated according to the behavior of the devices in the SE. This continuous feedback process enables the recurrent information knowledge about the smart environment. Consequently, the proposed solution can adapt and understand the usual behavior of the network flows and detect DDoS attacks.

3.2 Location Process

The location process is initiated upon detecting an attack, triggering the drone. Then, the detection module sends to the drone and to the access point the malicious device is connected to, the MAC address of the attacker. The server also calculates the trajectory that the drone should follow. In addition, the server maintains a T-tuple with three positions to store the information coming from the access point and the drone.

When receiving the attacker's MAC address, the access point sends the server the signal strength between itself and the attacking machine and the access point's location. While following its trajectory, the drone uses the broadcast strategy of the wi-fi signal and the MAC address provided by the server. This strategy consists in sending the drone's wi-fi signal and trying to detect all nodes within its range as it moves until the drone finds the attacking node, which is identified based on the MAC address information provided by the detection system. When the drone finds the attacking machine, it sends to the server the information of the drone's location and the signal strength between it and the target. Until the location of the attacking machine is undetermined, the drone continues to search for the target machine and send the information to the server. Finally, the location algorithm calculates the attacker's position when all tuple positions are filled.

The drone's trajectory follows the strategy SCAN (Koutsonikolas *et al*., 2007), in which the area where the drone will travel is divided into a square area and equidistant subsquares with centers connected using straight lines. We use this strategy as the uniform coverage of the lattice field helps to ensure a small location error (Koutsonikolas *et al*., 2007).

It is worth noting that the analysis of the drone's energy cost is not part of the scope of this proposal. Furthermore, we assume that there are no aerial obstacles and that we can continuously determine the drone's location. The impact of the number of messages exchanged between all nodes can be an important aspect but it is not in the scope of this study.

3.2.1 Trilateration

Initially, we will define the trilateration technique, which is a calculation that uses distance measurements to determine a position in three-dimensional space (Murphy and Hereman, 1995). This calculation makes it easy for real-time positioning systems to find targets without the need to measure angles. In a simplified way, to calculate the coordinates of a fixed target, we can use a simple system of three equations, as defined in Equation 1 (Courtay *et al*., 2019).

$$\begin{cases} d_{N1}^2 = (x_A - x_{N1})^2 + (y_A - y_{N1})^2 + (z_A - z_{N1})^2 + e_{N1} \\ d_{N2}^2 = (x_A - x_{N2})^2 + (y_A - y_{N2})^2 + (z_A - z_{N2})^2 + e_{N2} \\ d_{N3}^2 = (x_A - x_{N3})^2 + (y_A - y_{N3})^2 + (z_A - z_{N3})^2 + e_{N3} \end{cases} \quad (1)$$

In Equation 1, d_{Ni} represents the distance between the node i, whose position is known (for example, a drone), and the target. X_{Ni}, Y_{Ni}, and Z_{Ni} are the X, Y, and Z coordinates of nodes whose location is already known. X_A, Y_A, and Z_A are the coordinates of the target. And e_{Ni} is the error

due to the signal noise used to calculate the distances from node i to the target.

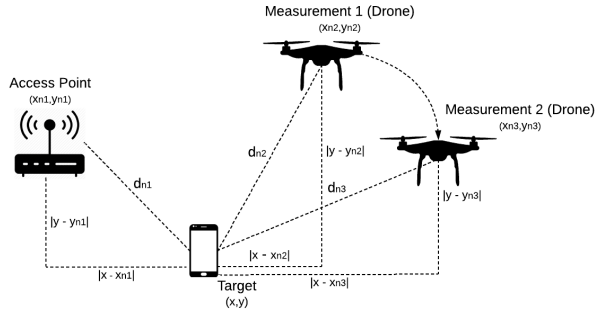

Figure 2. Example of Trilateration with drones

Figure 2 presents an example of trilateration in the context of this work, in which two drone measurements and an access point are used. Note that the z axis is not considered in the example. Then, to simplify the calculation, we assume that the attacker and the access point are at the same reference height, therefore their values for the z axis are zero and, for the drone, this value is equal to the drone's height in the time of measurement.

The calculation of trilateration seeks to solve the system represented in Equation 1 to determine the coordinates of the target (Courtay *et al*., 2019). One way to do this is to transform the Equation 1 into the Vector Equation $u = V.a + e$, which can be achieved by developing the terms of the system of equations and subtracting the distances in Equation 1, where:

$$u = \begin{pmatrix} (d_{N1}^2 - d_{N2}^2) - (x_{N1}^2 - x_{N2}^2) - (y_{N1}^2 - y_{N2}^2) - (z_{N1}^2 - z_{N2}^2) \\ (d_{N1}^2 - d_{N3}^2) - (x_{N1}^2 - x_{N3}^2) - (y_{N1}^2 - y_{N3}^2) - (z_{N1}^2 - z_{N3}^2) \\ (d_{N2}^2 - d_{N3}^2) - (x_{N2}^2 - x_{N3}^2) - (y_{N2}^2 - y_{N3}^2) - (z_{N2}^2 - z_{N3}^2) \end{pmatrix} \tag{2}$$

$$V = 2 * \begin{pmatrix} (x_{N2} - x_{N1})\ (y_{N2} - y_{N1})\ (z_{N2} - z_{N1}) \\ (x_{N3} - x_{N1})\ (y_{N3} - y_{N1})\ (z_{N3} - z_{N1}) \\ (x_{N3} - x_{N2})\ (y_{N3} - y_{N2})\ (z_{N3} - z_{N2}) \end{pmatrix} \tag{3}$$

$$e = \begin{pmatrix} e_{N1} - e_{N2} \\ e_{N1} - e_{N3} \\ e_{N2} - e_{N3} \end{pmatrix} \tag{4} \qquad a = \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix} \tag{5}$$

Given this transformation, we can determine the values of x_A, y_A, and z_A using some mathematical artifice. In our proposal, we use Cramer's rule (Courtay *et al*., 2019), due to the simplicity of implementation and the fast response time (in the place of milliseconds), which is important for our context of public environments. Thus, we calculate the value of x_A, solving the equations 6 and 7.

$$v1 = \begin{pmatrix} 2 * (x_{N2} - x_{N1}) \\ 2 * (x_{N3} - x_{N1}) \\ 2 * (x_{N3} - x_{N2}) \end{pmatrix} \tag{6}$$

$$x_A = \frac{det((u - e).v1)}{det(V)} \tag{7}$$

Analogously, we can use Cramer's rule to calculate the values of y_A and z_A and thereby find the target's location.

3.2.2 Distance Calculation

To calculate the attacker's location based on trilateration, you need three measurements from different sources. In each measurement, we obtain the coordinates of the location of the source (access point or drone) and calculate the distance from the source to the attacker using the signal strength between source and target. The signal strength is generated based on the attenuation of the signal energy between the two devices. This calculation is directly related to the (Abhayawardhana *et al*., 2005) signal propagation model, which has to be adequate to the environment configuration.

In this work, we use the Log Distance model, which models the propagation of wireless networks in indoor and outdoor environments with and without obstacles (Al-Hourani and Gomez, 2017). The signal strength for the Log Distance model is calculated following Equation 8. In this Equation, P and P_0 represent the signal strength (in dBm) for the distance d and d_0. The value α is the attenuation coefficient, which is a constant and depends on the medium and frequency used in the network. Finally, n is the signal strength measurement error. This measurement error impacts the signal noise of the trilateration formula. Thus, to calculate the distance, we transform the Equation 8 into the Equation 9.

$$P = P_0 - 10 * \alpha * \log_{10}\left(\frac{d}{d_0}\right) + n \tag{8}$$

$$d = 10^{(P_0 - P + 10 * \alpha * \log_{10}(d_0) + n)} \tag{9}$$

We can obtain the values P and P_0 during measurement. The value d_0 is equal to 1 meter and the α, in terrestrial environments, is equal to 3 (Al-Hourani and Gomez, 2017). Moreover, empirically, we were observed that in an open environment covered by a home router, there is a little variation for the signal strength measurement error between the access point, the drone, and the attacker. Then, for our current solution, we calculate the value of n based on the signal measurement between the drone and the access point. For this, we just isolate the value of n in Equation 9 and calculate the distance between the access point and the drone, based on their locations, which are known.

3.2.3 Location Algorithm

Algorithm 1 presents our solution to calculate the location of the malicious device applying Cramer's rule. The T tuple contains the P and P_0 signal strengths and the coordinate vectors of the location of the device that sent the information to the server (access point or drone). The n value is the signal strength measurement error. As we are using the value n, which we calculated previously, we will not use the error vector e (Equation 4), as we are already handling the errors during the distance calculation.

In lines 2 and 3, we initialize the auxiliary variables responsible for allocating the distances and the vector with the location of the attacking machine. Line 4 contains the initialization of the auxiliary counter. In lines 5 to 8, the distances

Algorithm 1: Calculation of location

Data: T e n
Result: Location of attacking machine
1 **begin**
2 $D = \{\}$
3 $vloc = (0, 0, 0)$
4 $i = 0$
5 **for** *each* $t \in T$ **do**
6 $D[i] =$ CalculateDistance(t, n)
7 i++
8 **end**
9 $u =$ GeneratesVectoru(T, D)
10 $V =$ GeneratesMatrixV(T)
11 $v1 = 2 * ((T[2].x - T[1].x), (T[3].x - T[1].x), (T[3].x - T[2].x))$
12 $v2 = 2 * ((T[2].y - T[1].y), (T[3].y - T[1].y), (T[3].y - T[2].y))$
13 $v3 = 2 * ((T[2].z - T[1].z), (T[3].z - T[1].z), (T[3].z - T[2].z))$
14 $x_A = \frac{det(u.v1)}{det(V)}$
15 $y_A = \frac{det(u.v2)}{det(V)}$
16 $z_A = \frac{det(u.v3)}{det(V)}$
17 $vloc = (x_A, y_A, z_A)$
18 **end**
19 **return** $vloc$

are calculated, and lines 9 and 10 use the values of the distances and locations of the access point and the drone to calculate the result related to Equations 2 and 3. Lines 11 to 16 show the direct application of Cramer's rule. Finally, in line 17, the calculated coordinates are assigned to the malicious device's location vector.

4 Results

This section presents the experiments performed to evaluate the proposed system for DDoS detection and devices location in smart environments. The experiments focus on evaluating the designed system to detect the attacks correctly and locate the devices in a suitable time and approximation.

4.1 Detection Experiments

The network traffic dataset that we use in the experiments is based on two datasets that were merged to represent an SE composed of heterogeneous IoT and Personal devices. The former is the dataset "BoT-IoT" [1] developed by Koroniotis *et al*. (2018), which contains both normal (benign) traffic and traffic related to the latest DDoS attacks. The latter is the "UNSW-IoT" created by Sivanathan *et al*. (2018), that has normal (benign) traffic of IoT and Personal devices. Both datasets are formatted in real-world monitoring data (PCAPs). Altogether, the dataset used in the experiment contains 100.000 traffic records, where half of the records contain normal traffic and the other half traffic related to DDoS attacks. From this database, 80 features were extracted using the CICFlowMeter tool pattern from Sharafaldin *et al*. (2019).

The methodology used for the experiment consists of dividing the database into two sub-bases, one for training the ML algorithms, containing 70% of the records and the other part, with the remaining 30% of records for testing. After splitting the dataset, the features are extracted and then selected. The ML models are trained with the subset of selected features. The tests are then carried with trained model applied to the testing data, where again we perform feature extraction and selection so that the trained model can classify the datapoints.

During the detection experiments, the following selection techniques were evaluated: Extra-Tree (EA), SVC, Lasso, Low Variance (LV), and mRMR (cases of 5, 10, 20, 30, and 40 features). These techniques selected the features used in the ML techniques trained: DT, SVM, KNN, RF, and LR. Thus, we evaluated all the possible combinations of selection and ML techniques, allowing a complete analysis of the possible performances.

4.1.1 Performance Metrics

The performance of the proposed intelligent system (including the combination of selection and ML techniques) considered the following evaluation metrics:

- Accuracy (in percentage): Rate of correct classifications

[1] https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/bot_iot.php

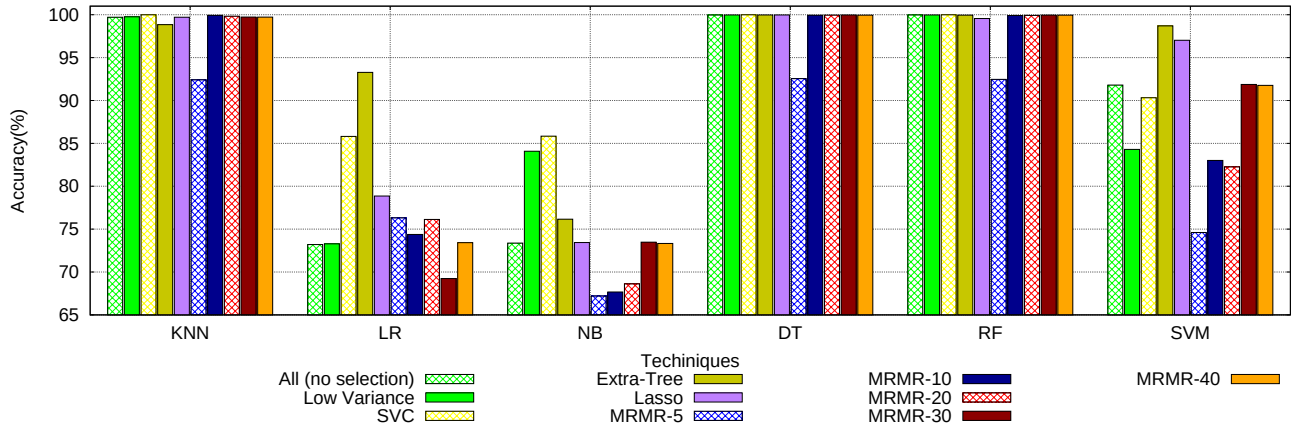

Figure 3. Accuracy for DDoS Detection

according to the Equation 10, i.e., the True Positive (TP) and True Negative (TN) cases about all other cases (TP, TN, False Positive - FP and False Negative - FN).

$$ACC = \frac{TP + TN}{\text{TP} + FN + FP + TN} \tag{10}$$

- Training Time (in seconds): time required to train the DDoS detector (ML model) with the selected input features.
- Detection Time (in seconds): time spent by the DDoS detector to define whether a case is a DDoS attack or not.

4.1.2 Detection Results

From the results shown in Figure 3, it can be seen that the accuracy of the ML techniques varies according to the applied selection technique, especially when these ML techniques are based on approaches that focus on dimensionalities, such as KNN, RL, and SVM classifiers. On the other hand, the ML techniques based on subset division (DT and RF) have almost no effect on selection techniques. It happens because of the recursive derivation process of the subsets, mitigating the variation on the performance of features and resulting in possible noises for the ML training.

Regarding NB and LR performance, both ML techniques present the worst results of the other approaches, regardless of the selection technique. Thus, NB and LR appear as unsuitable solutions for DDoS detection in SEs compared to the other approaches of the experiment.

Tables 3 and 4 show the time spent to perform the ML model training (creating the DDoS detector) and for the detectors to identify the cases of DDoS attacks, respectively. The results presented in both tables represent the feasibility of the ML techniques to be deployed in distinct contexts of SEs.

Based on the results presented in Table 3, the KNN and RF classifiers and, mainly, SVM have a higher training time than the other approaches. Nevertheless, the application of the mRMR selection technique (with 5 and 10 characteristics) reduces the training time of the RF classifier, enabling its deployment for SEs, achieving a time closer to the DT classifier.

Table 3. Training Time (in seconds)

Technique	KNN	LR	NB	DT	RF	SVM
80 Features	13.29	2.34	0.53	1.96	22.99	1625.04
LV	11.74	2.14	0.41	1.88	10.10	273.34
SVC	36.30	1.27	0.21	0.28	5.27	1226.96
Extra-Tree	18.55	2.96	0.19	0.68	13.33	351.01
Lasso	11.91	1.20	0.14	0.74	6.73	304.38
mRMR 05	21.78	2.91	0.13	0.16	4.11	1055.70
mRMR 10	16.01	3.43	0.14	0.38	5.45	708.88
mRMR 20	10.67	3.42	0.30	0.81	10.12	1177.78
mRMR 30	7.91	0.93	0.29	0.49	10.68	385.93
mRMR 40	9.16	1.23	0.28	1.19	15.53	467.02

Table 4. Detection Time (in seconds)

Technique	KNN	LR	NB	DT	RF	SVM
80 Features	9.02	0.02	0.53	0.02	0.53	162.57
LV	7.98	0.03	0.03	0.01	0.34	15.24
SVC	15.62	0.02	0.02	0.01	0.35	144.13
Extra-Tree	12.95	0.01	0.06	0.02	0.44	21.15
Lasso	8.15	0.01	0.14	0.01	0.45	142.08
mRMR 05	16.29	0.02	0.01	0.01	0.48	146.50
mRMR 10	12.43	0.02	0.01	0.01	0.53	142.84
mRMR 20	7.72	0.01	0.02	0.01	0.58	198.56
mRMR 30	5.46	0.02	0.03	0.01	0.49	86.29
mRMR 40	6.26	0.05	0.04	0.01	0.52	102.73

One explanation for the SVM algorithm to take much more time than others is that it needs to work in the high dimensional feature space, where it can be computationally challenging to separate the points for the 70.000 training datapoints. As an evidence of this, we see that the feature selection technique has a big impact in reducing the time it takes to train and detect using SVM as a model.

Similar to the training time, the detection time (presented in Table 4) of the KNN and SVM classifiers are longer than the other ML techniques. However, differently from the training time, the impact of the selection techniques is lower. In general, the LR, NB, and DT techniques spend very little time performing the detection. Close to them is the RF, proving to be a feasible solution.

In general, the DT classifier presented an overall most suit-

able performance since it can detect attacks with high accuracy and in a short time (mainly with methods mRMR 10 and SVC), while it can be trained fast when compared to other approaches.

4.2 Location Experiments

4.2.1 Simulation Configuration

The simulation was performed using Network Simulator 3 (NS3) (Riley and Henderson, 2010). We wrote the simulation code in C++ language, and the analysis of the results was done using scripts written in python 3.5[2].

We use a server host for the network topology, connected by a peer-to-peer connection to the host used as a wi-fi access point (AP), a mobile host (drone) with a wi-fi connection, and several other static hosts connected to the wireless network. At each simulation run, one of the unidentified static *hosts* is randomly chosen to act like an attacker.

The area covered during the simulation is approximately $100m^2$, the same coverage area as a home wi-fi router. We have also entered a package shipping error rate of 0.001%. This rate is suggested in one of the examples in the simulator's tutorial. The interval between sending packages is 1s. Finally, the drone's displacement speed is constant and equal to 10m/s.

Thirty runs were made for each of the four setups used. For each setup, the number of unidentified fixed wi-fi hosts were varied, as shown in Table 5. Note that we use a drone (mobile node), a server (fixed node), and a fixed AP (fixed node) in all configurations.

Table 5. Simulation Setup

Setup	Number of fixed nodes	Number of mobile nodes	Total of nodes
ID 1	12	1	13
ID 2	52	1	53
ID 3	102	1	103
ID 4	252	1	253

4.2.2 Location Results

We evaluate the simulation in two aspects: (i) regarding the distance between the location found and the actual location of the attacking machine, and (ii) the time taken until the approximate location of the target is calculated.

For distance, the results showed an accuracy of mm in all evaluated configurations. The greatest distances occurred in simulations of types 1 and 2. The time spent during the simulations ranged between 9 and 14 seconds. Again, type 1 simulations had the worst results. Table 6 summarizes the means and standard deviations ($\pm$) for each setup.

The very positive results of the proposal concerning distance are directly related to the measurement error n used. As the simulator uses a reference value n that practically does not vary, when calculating our own n, we obtained a value very close to the simulator's reference value, which, consequently, allowed the high precision of the algorithm.

Table 6. Average Simulation Results

Setup	Distance (mm)	Runtime (s)
ID 1	0.74 ± 0.57	10.28 ± 1.49
ID 2	0.5 ± 0.37	9.8 ± 0.84
ID 3	0.32 ± 0.26	9.63 ± 0.3
ID 4	0.19 ± 0.11	9.61 ± 0.19

We believe that the nature of the evaluated scenarios, which allowed us to obtain the value for n in the way previously mentioned, made it possible to obtain the very positive results in the experiments. However, we do not guarantee that this behavior will be the same in a real environment. In a more realistic scenario, we might need to calculate the value of n in another way. However, since the proposal allows other types of calculations for this value to be used, we believe that the results obtained in the simulation are sufficient to validate that our proposal can calculate the approximate location of the target.

The time taken to locate the target, independent of the location calculation, indicates that the time required to locate the attacking machine using the proposal is small, close to 10s, for the specified area. Also, despite the minimal variation in the results, the simulations indicated that the number of nodes has little impact on the proposal's performance. We believe the results are related to greater spread with fewer nodes within the environment.

Finally, it is important to note that the step that consumed the most processor, memory, and time resources during the experiments was the training of the ML models. But as it runs on the server host, it is assumed to have reasonable resources, and therefore memory and processor were not considered critical in our study. The critical resource considered was the times needed to train the ML model, detect the attacks and locate the attacking device, that we presented in our results.

5 Conclusion and Future Work

Internet access in public environments allows users to access their data online from anywhere and the deployment of new services based on IoT. This fact created SEs composed of both personal and IoT devices. Nevertheless, these environments can facilitate the action of malicious agents performing DDoS attacks on the network. In this context, this work presented a system to detect attacks to wireless networks in public environments and locate the devices used to do that. This system focuses on an underexplored scenario using measurements from a drone and an access point to calculate the target's location through trilateration.

Regarding our future work, we intend to extend the proposed system to use more than one drone and moving targets in the network and evaluate other solutions to calculate trilateration, such as methods based on Gaussian elimination. Furthermore, the detection can be improved using deep learning techniques.

[2]The simulation codes are available at https://bit.ly/2LU55st

Acknowledgment

The authors would like to thank both Coordination of Improvement of Higher Level Personnel - Brazil (CAPES) that provided to Evilasio Costa Junior a Ph.D. scholarship, as well as CNPQ for the Productivity Scholarship DT-1D of Rossana M. C. Andrade (N^o 306362/2021-0) and DT-2 of Rafael L. Gomes (N^o 303877/2021-9).

References

Abhayawardhana, V., Wassell, I., Crosby, D., Sellars, M., and Brown, M. (2005). Comparison of empirical propagation path loss models for fixed wireless access systems. In *2005 IEEE 61st Vehicular Technology Conference*, volume 1, pages 73–77. IEEE. DOI: 10.1109/VETECS.2005.1543252.

Acuna, V., Kumbhar, A., Vattapparamban, E., Rajabli, F., and Guvenc, I. (2017). Localization of wifi devices using probe requests captured at unmanned aerial vehicles. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE. DOI: 10.1109/WCNC.2017.7925654.

Al-Hourani, A. and Gomez, K. (2017). Modeling cellular-to-uav path-loss for suburban environments. *IEEE Wireless Communications Letters*, 7(1):82–85. DOI: 10.1109/LWC.2017.2755643.

Alotaibi, B. and Elleithy, K. (2016). Rogue access point detection: Taxonomy, challenges, and future directions. *Wireless Personal Communications*, 90(3):1261–1290. DOI: 10.1007/s11277-016-3390-x.

Andrea, I., Chrysostomou, C., and Hadjichristofi, G. (2015). Internet of things: Security vulnerabilities and challenges. In *2015 IEEE Symposium on Computers and Communication (ISCC)*, pages 180–187. DOI: 10.1109/ISCC.2015.7405513.

Betti Sorbelli, F., Das, S. K., Pinotti, C. M., and Silvestri, S. (2018). Range based algorithms for precise localization of terrestrial objects using a drone. *Pervasive and Mobile Computing*, 48:20–42. DOI: 10.1016/j.pmcj.2018.05.007.

Brun, O., Yin, Y., Augusto-Gonzalez, J., Ramos, M., and Gelenbe, E. (2018). Iot attack detection with deep learning. In *ISCIS Security Workshop*.

Chamoso, P., González-Briones, A., Rodríguez, S., and Corchado, J. M. (2018). Tendencies of technologies and platforms in smart cities: a state-of-the-art review. *Wireless Communications and Mobile Computing*, 2018. DOI: 10.1155/2018/3086854.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27. DOI: 10.1145/1961189.1961199.

Courtay, A., Le Gentil, M., Berder, O., Scalart, P., Fontaine, S., and Carer, A. (2019). Anchor selection algorithm for mobile indoor positioning using wsn with uwb radio. In *2019 IEEE Sensors Applications Symposium (SAS)*, pages 1–5. IEEE. DOI: 10.1109/SAS.2019.8706113.

Diro, A. A. and Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for internet of things. *Future Generation Computer Systems*, 82:761–768. DOI: 10.1016/j.future.2017.08.043.

Doshi, R., Apthorpe, N., and Feamster, N. (2018). Machine learning ddos detection for consumer internet of things devices. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 29–35. IEEE. DOI: 10.1109/SPW.2018.00013.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1):3–42. DOI: 10.1007/s10994-006-6226-1.

Halder, S. and Ghosal, A. (2016). A survey on mobile anchor assisted localization techniques in wireless sensor networks. *Wireless Networks*, 22(7):2317–2336. DOI: 10.1007/s11276-015-1101-2.

Kaushik, S. (2016). Introduction to feature selection methods with an example (or how to select the right variables?). https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/.

Koroniotis, N., Moustafa, N., Sitnikova, E., and Turnbull, B. (2018). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *CoRR*, abs/1811.00701. DOI: 10.1016/j.future.2019.05.041.

Koutsonikolas, D., Das, S. M., and Hu, Y. C. (2007). Path planning of mobile landmarks for localization in wireless sensor networks. *Computer Communications*, 30(13). DOI: 10.1016/j.comcom.2007.05.048.

Mozaffari, M., Saad, W., Bennis, M., Nam, Y.-H., and Debbah, M. (2019). A tutorial on uavs for wireless networks: Applications, challenges, and open problems. *IEEE Communications Surveys & Tutorials*. DOI: 10.1109/COMST.2019.2902862.

Murphy, W. and Hereman, W. (1995). Determination of a position in three dimensions using trilateration and approximate distances. *Department of Mathematical and Computer Sciences, Colorado School of Mines, Golden, Colorado, MCS-95*, 7:19.

Nobles, P., Ali, S., and Chivers, H. (2011). Improved estimation of trilateration distances for indoor wireless intrusion detection. *JoWUA*, 2(1):93–102. DOI: 10.22667/JOWUA.2011.03.31.093).

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238. DOI: 10.1109/TPAMI.2005.159.

Riley, G. F. and Henderson, T. R. (2010). The ns-3 network simulator. In *Modeling and tools for network simulation*, pages 15–34. Springer. DOI: 10.1007/978-3-642-12331-3_2.

Sharafaldin, I., Lashkari, A. H., Hakak, S., and Ghorbani, A. A. (2019). Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In *2019 International Carnahan Conference on Security Technology (ICCST)*, pages 1–8. IEEE. DOI: 10.1109/CCST.2019.8888419.

Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., and Sivaraman, V. (2018). Classifying iot devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18(8):1745–1759. DOI: 10.1109/TMC.2018.2866249.

Sun, Y., Wen, X., Lu, Z., Lei, T., and Jiang, S. (2018). Localization of wifi devices using unmanned aerial vehicles in search and rescue. In *2018 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, pages 147–152. IEEE. DOI: 10.1109/ICCChinaW.2018.8674518.

Vinayakumar, R., Alazab, M., Srinivasan, S., Pham, Q.-V., Padannayil, S. K., and Simran, K. (2020). A visualized botnet detection system based deep learning for the internet of things networks of smart cities. *IEEE Transactions on Industry Applications*. DOI: 10.1109/TIA.2020.2971952.

Vishwakarma, R. and Jain, A. K. (2020). A survey of ddos attacking techniques and defence mechanisms in the iot network. *Telecommunication Systems*, 73(1):3–25. DOI: 10.1007/s11235-019-00599-z.

Yamauchi, M., Ohsita, Y., Murata, M., Ueda, K., and Kato, Y. (2019). Anomaly detection for smart home based on user behavior. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6. IEEE. DOI: 10.1109/ICCE.2019.8661976.