International Journal of Engineering &Technology, 13 (2) (2024) 341-350

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Proposal for a new architecture for detecting intrusion clusters in IoTs

Kanga koffi [1] *, BROU Aguié Pacôme Bertrand [2], Kamagaté Beman Hamidja [3]

[1] Doctor in computer science specializing in software and database engineering: INPHB doctoral school Teacher – researcher at ESATIC (African Higher School of ICT: Republic of Ivory Coast)
[2] Doctor of Computer Science Teacher – researcher at ESATIC (African Higher School of ICT: Republic of Ivory Coast)
[3] Doctor in computer Science specializing in network and cybersecurity INPHB doctoral school, Teacher – researcher at ESATIC (African Higher School of ICT: Republic of Ivory Coast)
**Corresponding author E-mail:kanga.koffi@esatic.edu.ci / kangamiage@gmail.com*

Abstract

In this paper, we are proposing a new architecture to detect intrusions in an IoT environment. To achieve this, we reviewed the various works to our knowledge, relating to intrusions in IoT. As a contribution, our proposed architecture has 5 components (sensors - data storage - preprocessing and associated classification algorithms). Also To provide intelligence and evaluate the performance of our architecture, we initially implemented in our architecture the DBSCAN and K-means algorithms separately. Secondly, we proceeded with a hybridization of these algorithms (DBSCAN and K-means) . In terms of results, this hybridization allowed us to use DBSCAN to identify dense clusters of arbitrary form of data as well as intrusions in this data. As for K-means, it made it possible to refine the globular clusters found by DBSCAN in order to best detect and predict the sources of intrusion. These results show that our solution makes it possible to detect intrusions in IoTs in an efficient manner compared to the 2 algorithms (DBSSAN AND K-means) applied separately in our architecture.

Keywords: *IoT Intrusion Detection Classification Algorithm; Dbscan; K-Means.*

1. Introduction

Talking about intrusion detection architecture in IOTs deserves some explanations. Indeed, according to the IEEE, the Internet of Things (IoT) is a complex, adaptive and self-configuring network that connects "objects" to the Internet via standardized communication protocols. Interconnected objects have a physical or virtual representation in the digital world, a detection and actuation capacity, a programming function. These connected objects are uniquely identifiable. Their representation takes into account information such as the identity of the object, its status, its location or any other relevant commercial, social or private information. These objects provide services, with or without human intervention, through the exploitation of unique identification, data capture and communication and their actuation capability.

1.1. operating principle of IoT tools

In their operation, IoT tools are based on several basic principles which are:

- Interconnection: IoT objects are interconnected via wireless or wired networks, allowing them to communicate with each other and with remote computer systems.
- Data collection: Sensors embedded in IoT objects collect various data about the physical environment, such as temperature, humidity, location, etc.
- Real-time processing: Collected data is often processed in real time by IoT devices or transmitted to remote servers for analysis and interpretation.

1.2. Basic architecture of IoTs

The architecture of an IoT solution (fig 1) varies from one system to another based on the type of solution to be implemented. The most basic architecture is a three-layer architecture:

- The perception layer has sensors and actuators that detect and collect information about the environment.
- The network layer is responsible for connecting, transporting and processing data from sensors and actuators.
- The application layer is responsible for providing the user with specific services and intelligent applications.

Copyright © Kanga koffi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

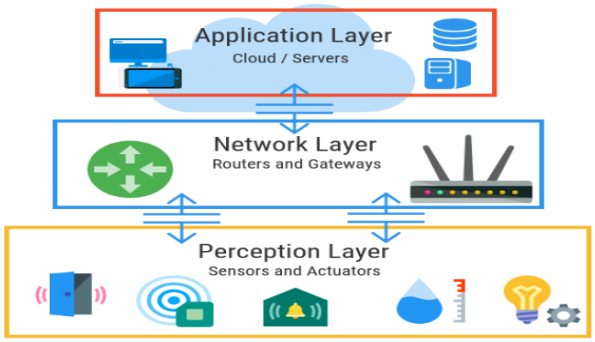


Fig. 1: IoT Architecture.

1.3. Intrusion detection [dagorn, n. (2006)]

As for intrusion detection, it falls within the field of IT security. Indeed, it represents all the actions making it possible to highlight any actions compromising the confidentiality, the integrity or the availability of a resource.
In their operation, IDS can be classified into four (4) families which are:

- Network IDS (NIDS) : The main role of a NIDS (Network-based Intrusion Detection System) is used to analyze and interpret malicious data packets passing through the network using signatures. NIDS use detectors, often simple hosts, to monitor traffic and trigger alerts if necessary. They operate at different levels of the network (network, transport, application layers) and are capable of detecting packets designed to bypass the rules of an overly permissive firewall, as well as spotting signs of attacks at various locations on the network .
- Host IDS (HIDS) : HIDS focus on analyzing the activities of a single host, making them more accurate in detecting attacks
- Network Node IDSs (NNIDS) analyze network traffic packets, but only those destined for a specific node. Unlike traditional NIDS, NNIDS does not operate in promiscuous mode , which improves scanning performance because only relevant packets are checked
- Application-based IDSs (ABIDS) are a subgroup of host IDSs, which monitor the interaction between a user and a program by adding log files to provide detailed information about activities. They are effective at filtering suspicious behavior and detecting potentially dangerous commands. However, they have two major drawbacks: they are unlikely to detect threats such as Trojans because they do not operate in kernel space, and the log files they generate are vulnerable to attacks. .

“Honeypots” are intentionally exposed and vulnerable computer tools designed to attract and trap hackers, while observing their behavior and recording their attack methods. Their main task is to analyze traffic to create detailed profiles of potential attackers [Kuwatly, I., Sraj and al][Bhagat , N and al]
Also, among the intrusion detection techniques used by these IDSs, we could cite:

- Misuse detection [4, its involves the analysis of data by the IDS to identify patterns corresponding to known attacks. This process, called pattern matching or scenario-based approach, compares the collected information to a database of signatures representing explicit characteristics of documented attacks. Any corresponding activity is considered an attack, with varying levels of severity. An example of an abuse detection system is STAT (State Transition Analysis Toolkit) .
- Anomaly detection (anomaly Behavior detection is a method that dates back quite a long time and is used in various fields, including telephony to detect suspicious activities such as phreaking . Its fundamental principle consists of establishing a model of the "normal" behavior of a system, a program or a user during a learning period, thus defining a reference (called baseline or profile). Then, during the detection phase, any unusual behavior is considered suspicious, exhibiting significant deviations from the normal behavior pattern. Anomaly detection models often use statistical techniques, such as those employed in NIDES (NIDES/STAT) or Haystack . [Guleria , D.and al]

1.4. Why implement intrusion detection in IOTs

Since its advent, the IoT field has repeatedly been the target of large-scale attacks on its infrastructure [Kuwatly, I., Sraj and al]. Indeed, one of the recent attacks is the Bonet Mirai attack in 2016 [Affinito , Aand al] . This robot network has compromised many IoT devices, including routers and IP cameras.
Faced with this situation, making use of IOTs associated with IDS for data capture, monitoring network traffic, detecting anomalies, analyzing signatures in a secure context is necessary, but equipping ourselves with a viable and secure architecture would be even better.
The remainder of our paper is organized as follows:

- Section 2, we will present the state of the art
- section 3, we willidentify our problem
- Section 4, we will illustrate our contribution
- Section 5 is devoted to a discussion and we will end with a conclusion in Section 6. In this section, we will identify some perspectives.

2. State of the art

In this chapter, we will explore the state of the art of intrusion detection in IoTs, examining the different types of attacks in IoTs, the techniques, methodologies and tools used to protect these systems against threats and malicious attacks. We will analyze recent advances, emerging trends, and ongoing challenges in this crucial area of cybersecurity , as well as the best practices and recommendations.

2.1. The main attacks in IOTs [tournier, j and al.]

Physical attacks and cyber attacks in IoT systems can be passive or active. Passive attacks involve spying on data flows, compromising the authenticity and confidentiality of communications. Active attacks go beyond eavesdropping, allowing the attacker to modify or cut off communications, with techniques such as DoS or DDoS attacks .
These attacks pose serious threats to the security of IoT devices and data.
Additionally, we present the most common cyber attacks in IoT below:

2.1.1. DDoS attacks [bouzoubaa , k and al]

DoS (Denial of Service) attacks aim to block legitimate users' access to a service by exploiting resource limitations of IoT systems, such as bandwidth, computing power or energy. They involve overwhelming these systems with a massive flow of harmful data, requiring only an Internet connection and a target system to launch the attack.
DDoS (Distributed Denial of Service) attacks are similar to DoS attacks , but they are orchestrated by botnets , networks of Internet-connected devices controlled by the attacker and geographically dispersed. Due to the large number of poorly secured IoT devices, DDoS attacks are particularly prevalent in IoT and can be carried out in a variety of ways:
Flood attacks (HTTP, TCP SYN and UDP or ICMP) are based on a huge packet that floods the connection , that the victim does not support in order to compromise their bandwidth and prevent legitimate users from accessing the servers .
Waterdrop attacks : This attack exploits the principle of fragmentation of the IP protocol. The victim does not reassemble packets using the incompatible packet offset values they contain. This attack has the effect of crashing the systems.
“Ping of Death” is an attack that sends IP packets to the victim with a size exceeding the allowed limit, which prevents their reassembly and disrupts the operation of the targeted system.

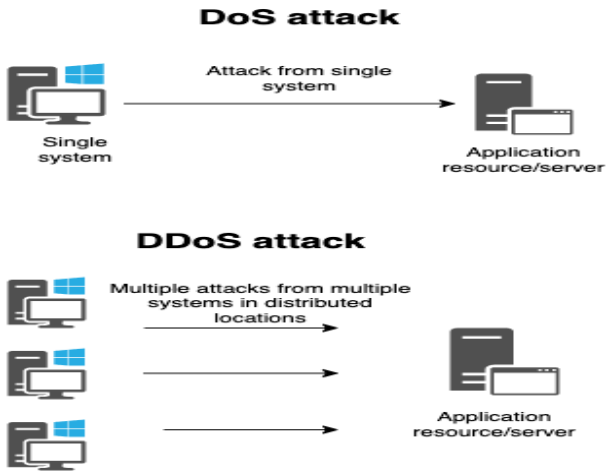


Fig. 2: DoS and DDoS Attacks.

Keylogging consists of recording the actions performed by a user, with the aim of monitoring their activities and recovering confidential information such as identifiers. This process begins with the remote installation of a keylogger, often through a Trojan horse inserted into the connected object. Then, malware is used to capture and transmit the sensitive data. These methods are often used to carry out clandestine surveillance.

2.1.2. Attacks against IoT protocols

IoT systems use lightweight protocols to manage their limited resources. These protocols are subject to attacks that exploit their internal structures to influence the communication channel as well as the data communicated. These attacks can be classified into five groups:

- Communication protocol-based attacks : These attempt to exploit changes that occur during transition phases between nodes during a communication. They comprise:
- Privacy Attacks : This protocol is vulnerable to spoofing, eavesdropping, and MITM attacks due to the lack of data encryption.
- Authentication attacks : Any node, including malicious nodes, can join the network and gain legitimate access because this protocol does not have an authentication system.
- Replay attack : If an intruder retrieves the packets, he can retransmit them as normal traffic but with modified content, as if a valid sender had sent them.
- Sniffing : when an attacker manages to capture all the packets circulating in the network (user identity and passwords).

Table 1: Table of Some Attacks in the IoT

Attack Name	Purpose and result of the attack	Threat	Active or inactive
Back	Saturate a server or block traffic Make a service unavailable	Integrity Availability confidentiality	Active
Man in the middle	Intercept communications between two parties and control the conversation	Integrity confidentiality	Active
Identity theft	Identity theft ; Carry out fraudulent actions; Deliberately assuming the identity of another living person	Integrity confidentiality	Active
Flooding	Exhaust the memory and energy of the nodes, Saturate the network	Integrity Availability	Active

2.2. Some intrusion detection work

Having reviewed the main IoT-related attacks, we now explore the work being done to mitigate these threats to protect existing IoT systems and networks. Traditional IT security solutions, originally designed for servers, networks and cloud storage, can also be adapted to secure IoT systems. These defense mechanisms can be used individually or combined depending on specific threats. In this section, we will describe traditional mechanisms that can be applied to strengthen the security of IoT systems.

2.2.1. The work of kasinathan et al. [kasinathan , p.]

They examined DoS detection for 6 LoWPAN as part of the EU-funded ebbits project (FP7). The architecture shown in Figure 3 is based on the Suricata IDS . Although IDS probes are distributed, the architecture is considered centralized. Indeed, these probes, as external modules, monitor the network in promiscuous mode and transmit data to the main NIDS (based on Suricata) via a wired connection. When the latter detects traffic corresponding to an attack signature, an alert is sent to the denial of service attack protection manager. This handler analyzes the attempt using additional data collected from other ebbits handlers , and reduces the rate of false alarms (incorrect detection of genuine data: false positives and false negatives). This approach overcomes the limitations of SVELTE because the IDS mechanism is not tied to the network architecture and therefore cannot be affected by DoS attacks against the IoT . The proposed (demo) framework is scalable and can be applied in real time to most IoT systems.
This work was evaluated using a penetration testing system (PenTest) called Scapy . The IDS adapts existing open source technologies. It starts from Suricata , an open-source IDS, and modifies it with IEEE 802.15.4 and 6LoWPAN decoders. Additionally, an additional detection module, FAM, consists of a frequency agility manager that analyzes channel occupancy states in real time to enable the network to become aware of the interference level and monitor attacks on Prelude , which is a security incident and event management (SIEM) system for monitoring attack events or alerts. The Suricata engine triggers alerts based on programmed rules; this solution could therefore detect different attacks depending on the rules developed.

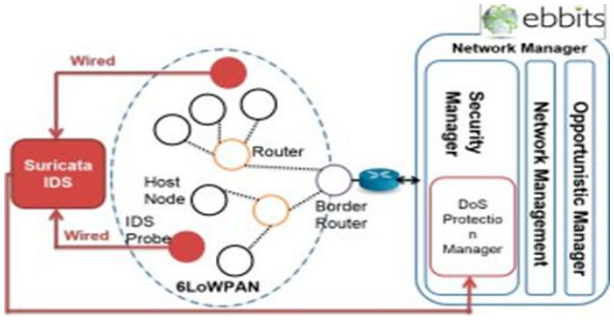


Fig. 3: DEMO Architecture.

2.2.2. The work of jun and chi [jun, c. and al]

Jun and Chi developed an IDS for IoT systems using CEP (Complex Event Processing) technology, known for its ability to filter and process events in real time. This approach is suitable for large volumes of messages with low latency, thus meeting the requirements of IoT. They favored online rather than offline performance evaluation. Their solution architecture is shown in Figure 4. The process starts with collecting data (network traffic and usage events) from IoT devices, followed by extracting events from the detected data. Then, security event detection is performed using the EPR event processing repository and the CEP engine. Finally, the actions are executed by the action engine. Jun and Chin implemented their IDS architecture using Esper , a CEP engine for complex event processing and event series analysis. Their approach, although intensive in terms of CPU, consumes less memory and performs better in real time. For example, for 800,000 data, CEP-based IDS uses 62% of the CPU, 730 MB of memory and 422 milliseconds of processing time, while a traditional IDS uses 57% of the CPU, 1,064 MB of memory and 8,688 ms to process the same amount of data. However, it is important to note that their architecture has not been evaluated for attack detection.

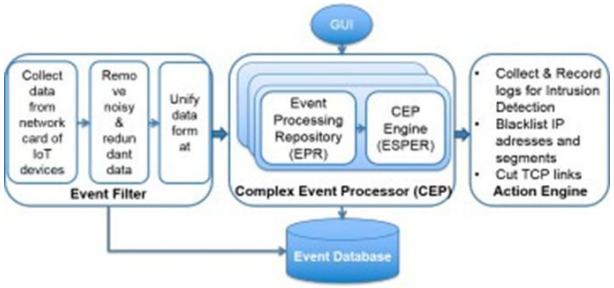


Fig. 4: CEP-Based IDS Architecture for IoT.

2.2.3. The work of surendar and umamakeswari [surendar , m and al]

Surendar and Umamakeswari developed an intrusion detection and response system (InDReS) for IoT networks using 6LowPAN. This system uses constraint-based specification techniques to detect sink attacks in RPL networks. In InDReS , sensor nodes are organized into clusters under the supervision of an observer node, which uses Dempster Shafer theory to detect malicious nodes by counting packets dropped by their adjacent nodes. Once detected, these malicious nodes are isolated with the cooperation of all nodes, and the network rebuilds itself. The authors demonstrated that this strategy improves the efficiency of some critical QoS metrics compared to the existing " INTI " scheme, particularly regarding energy consumption and packet drop rate. This proposal was simulated on the NS2 simulator.

2.3. Other intrusion detection architectures based on machine learning

In this section, we will explore several machine learning (ML)-based architectures in the context of network intrusion detection systems (NIDS). This in-depth analysis will allow us to better understand how ML techniques can be applied to strengthen the security of IoT networks against current and emerging threats .

2.3.1. Architecture proposed by hodo [12]

Hodo and colleagues employed a multi-layer perceptron (MLP), a form of supervised ANN, in an offline IoT IDS. This architecture includes three layers, with a sigmoid transfer function for neurons in the hidden and output layers. Their study, based on Internet packet traces, aims to detect DoS and DDoS attacks in IoT networks. The NIDS was evaluated on a simulation comprising four client nodes and one server relay node. DOS/ DDoS attacks were carried out on the server node, with 10 million UDP packets sent by a single host for the DoS attack and with three hosts at wire speed for the DDoS attack . The training data set had 2,313 samples, of which 496 were used for validation and the same number for testing. The overall attack detection accuracy was 99.4% with a false positive (FP) rate of 0.6%. These results ensure early detection of attacks, thus guaranteeing network stability.

2.3.2. Architecture proposed by nobakht [hodo, e and al]-[nobakht , m and al]

Nobakht et al presented an IoT-IDM host-based IDS framework for user-selected smart devices in smart homes. IoT-IDM monitors device traffic to detect threats, using Software Defined Networking (SDN) architecture and machine learning (ML) techniques . It identifies compromised hosts and mitigates attacks by taking actions such as blocking the intruder or redirecting malicious traffic to the underlying routers/switches. SDN technology enables remote security management, providing Security as a Service (SaaS) to IoT IDM users . The solution of Nobakht et al. stands out for its modular design, composed of five distinct modules: Device Manager, Sensor Element , Feature Extractor , Detection Unit and Mitigation Unit . Therefore, there is flexibility in choosing an ML algorithm from a given set of techniques. ML algorithms use signature patterns learned from known attacks to train the model. Besides the wide range of attacks detected, one of the disadvantages of IoT-IDM is that technically it cannot monitor all home IoT devices due to the high volume of network traffic, with the detail that the sensor elements are positioned above the SDN controller. Therefore, the IoT-IDM can only inspect chosen IoT devices that do not overload the SDN controller. Nobakht and al. tested IoT-IDM on a real IoT device (the connected lamp (Hue lights)), and compared the results produced by logistic regression and SVM (support vector machines). In unauthorized detection, the first gives a precision rate (TP) of 94.25% and a recall rate (TR) of 85.05% compared to 98.53% and 95.94% for SVM.

2.3.3. Comparison of some architectures

As presented earlier, many researchers are paying attention to IoT-powered NIDS using machine learning algorithms. A comparison between the previously presented proposals is detailed in Table 2 , where the main focus is on the IDS deployment, the detection methodology, the data sets used, the threats addressed and the ML algorithms employed.

Table 2: Summary of Some NIDS For IoTt Based on Learning Techniques

Reference	IDS deployment	Deployment methodology	Dataset used	Threats addressed	ML algorithms
Hodo and al	Network IOT	Base on anomalies	From simulation	DoS / DDoS	Multilayer perceptron (MLP)
Nobakht and al.	Network (SDN)	Based on host anomaly	From devices IoT real	Unauthorized access	Regression logistics vs. SVM
Hosseinpour and al.	Distributed	Base on anomalies	KDD99 and SSH Brute Force from ISCX	botnet attack	System immune artificial (SIA)

3. Problematic

After this literature review highlighting the different intrusion detection architectures to our knowledge, it should be noted that an excellent work has been carried out. This work ranges from architectures based on IDS, NIDS and even the possibilities offered by machine learning tools. However, in this new world characterized by the increased use of IOT tools associated with the exponential growth of social networks, producing and storing all kinds of data; data that will need to be processed or used at high speed; It turns out that these proposed architectures obviously have limits in terms of security. So, the question that arises is:
What architectures are proposed for security or intrusion detection in an IOT environment producing data?
This is the question that our contribution will try to address.

4. Contribution

To make our contribution to the question raised by our problem, we propose an architecture called IOT For BIG DATA. This architecture is made up of four elements:

- IoT tools for data capture
- The dataset
- Pretreatment
- The hybrid machine learning algorithm associated to give intelligence to this architecture

4.1. Architecture components and their roles

The main components of our architecture are: the IoT tools (data sensors), the dataset used, preprocessing and the Machine Learning algorithm.

- IoT tools (sensors)

These tools consist of all the sensors or devices allowing all kinds of information to be captured from all media contributing to the operation of the IoT network.
Among these tools we can find cameras, telephones, computers and all other communication equipment

- The dataset

It is the result of the different captures carried out by the sensors (IoT tools), these data can be of various types (text – images – sound – structured or unstructured data). Thus, for the present work we use the Ton- IoT dataset . Indeed, this data set represents a reference in terms of data used for IoT. Also, this game contains two characteristics for the attack:

- " label ", categorized as normal or attack,
- as well as " type ", denoting threat subclasses for multi-class classification tasks, and various attack scenarios; Additionally, it contains IoT telemetry data and a separate dataset per IoT device. Due to its heterogeneity (OS logs, pcap files , sensor data and Bro logs) and the relevance of different types of attacks attempted on more varied IoT devices, Tim M. Booij et al proved that the set of ToN_IoT (derived from a large-scale heterogeneous IoT network) which is collection available for IoT network IDS. Furthermore, they found a good balance in the distribution of features between the test and training sets, all of which illustrates the performance of machine learning applications in artificial intelligence. TON_IoT dataset , i.e. the training dataset (Train _Test_IoT)
- Pretreatment

This component is responsible for cleaning and normalizing data. In fact, data cleaning consists of eliminating aberrant data and/or zero values which may have been found in the sets resulting from the capture.
As for normalization , it consists of scaling the data so that it lies in the interval [0,1]. We use it because we want to preserve the linear character between the original non-numerical data (TON_IoT data being varied) then (categorical and digital at a different scale), and the digital data obtained after normalization.
In our case the normalization is carried out by the min-max method given by the following equation (1):

$$X_{norm} = \frac{X - Xmin}{Xmax - Xmin} \in [0,1] \quad (1)$$

$$\text{With} \begin{cases} Xmin = \text{minimum value of X in the dataset} \\ Xmax = \text{maximum value of X in the dataset} \end{cases}$$

And X_{norm} = the normalized value of information from the data set
Categorical data is encoded with the integer of its index to transform it into numerical values (1,2,3, etc.).

- The Machine Learning algorithm applied

TON_IoT dataset offers two output variables depending on the classification type.
For binary classification, we use the " label " attribute, while for multi-class classification, the attribute used is " type " of the dataset. To carry out this classification, we use the following classic algorithms in this study:
DBSCAN, and K-MEANS then we will combine these methods to improve machine learning results in ensemble learning (EL).
Several EL methods are the most popular:

- Booster: includes the most common algorithms AdaBoost and Gradient Boosting .
- Voting: involves creating standalone algorithms from the training dataset. Then, a voting classifier encapsulates the model by combining the predictions from these already created models.
- Stacking: After training each classifier based on the full training set, the classifiers are combined via a meta-classifier. The latter is adjusted according to the outputs of the meta characteristics of the individual models

4.2. Architecture diagram

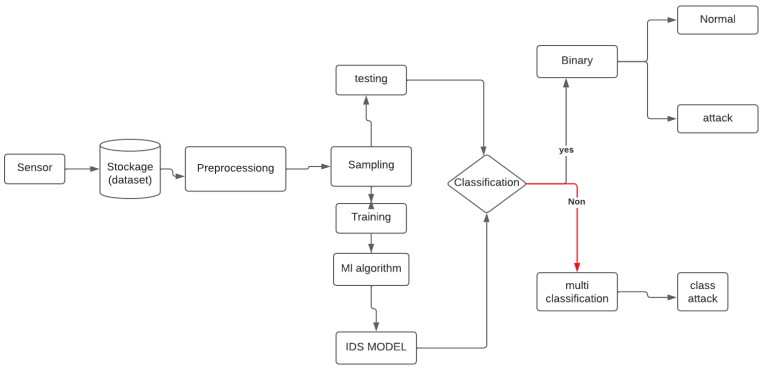


Fig. 5: Diagram of Our Architecture.

4.3. How the architecture works

Our architecture in its operation presents the following stages:
Step1: data capture (managed by the sensor component)
This step consists of positioning the different sensors which will be responsible for collecting the different information circulating on our iot network.
Step 2 data storage (managed by the Storage – dataset component)
big data database therefore supporting nosql
Steps 3 data preprocessing (fig 6) The data preprocessing step is made up of 3 sub-steps which are:

- The normalization sub-step:

It allows scaling of attributes with numeric values, excluding categorical and Boolean type attributes. Thus, we store the attributes which correspond to the numerical data are stored in a list denoted LA , while the other data which do not require preliminary processing are stored in the list denoted LB
The normalized data are obtained from the formula in (1) and come from the LB list.
After normalization, all the initial data is obtained by reconstitution of the sets La and LB such that:

Either:
Ed: starting set

$$\text{Ei a piece of data} \in \text{Edis such that Ei} \in \text{soit à } \begin{cases} \text{LA si Ei est entier} \\ \text{LB sinon} \end{cases} \quad (2)$$

$$\text{Also Ed} = (\cup \text{Ei}) = \text{LA} \cup \text{LB}$$

Formally, the normalization inputs are defined by the following formulas

$$\forall i \in [1, n], \text{Entreenormalisation} = \{\text{Ei} \in \text{LB}\} \quad (3)$$

- The partitioning/grouping sub-step

Here the normalized data is grouped into groups which will define, in part, the different possible states that the IoT system can take. The normalization step having made it possible to associate numerical values with each piece of data $E_{i\text{, in the}}$ partitioning we associate each numerical value (normalized value) with a group.
Let $Gr_{i\text{ be}}$ this group and S_{pi} an exit from the partitioning.
We define Spi such that:

$$\text{Spi} = \{E1, E2, \dots, \text{En}, \text{Gri} \backslash Z\} \quad (4)$$

With Z ={Ei ∈ LA}

$$\text{And GroupeGri} = \{\text{Gri}, \text{BarycentreGri}, \text{DmaxGri}\} \quad (5)$$

Equation 5 defines all data saved and stored for GroupeGri .
D_{maxGri} = the distance between the normalized data and the barycenter of the i th group;
BarycentreGri étant *le barycentre du groupe des Gri*
Thus, processing attributes with different orders of magnitude should not impact the partitioning.
Thus the partitioning output is defined as follows:

- The cutting sub-step

Here each record of the dataset is divided into x sub-parts denoted by Pi and which correspond to subsets of the dataset transformed by the normalization and partitioning steps. For each of these sub-parts, we use the numbering of the data in a time interval that we denote I. This allows us to ignore particular models that could create problems during the detection phase.

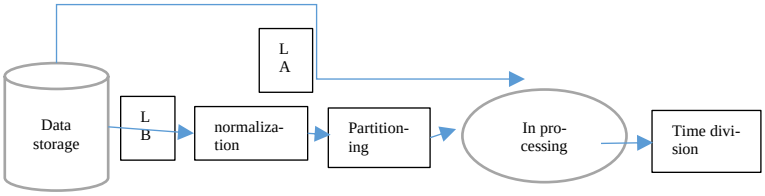


Fig. 6: Processus De Preprocessing.

Step 4 sampling (managed by component sampling)
Sampling consists of choosing a part of a dataset in order to apply machine learning algorithms for this purpose.
In our case we use the "ToN-IoT" dataset to which we apply the partitioning algorithm using DBSCAN density. The main advantages of using DBSCAN are its resistance to noise and its ability to find groups of inappropriate shapes. Here it is a question of source data of intrusion or not.
On the other hand, one of its disadvantages is the difficulty of choosing the most relevant parameters. Indeed, it will be necessary to notify the algorithm of the minimum number of data points that a group must contain before being able to consider it
Step 5 Application of the classification type (binary or multi classes)
Here we choose to apply either:

- Binary classification to know if our data contains attacks or if we have normal data including data from healthy communication
- Multi-class classification which will allow us to identify classes (clusters) of attacks (sets of attacks which have identical centriodes)

4.4. Implementation and testing of algorithms (dbscan – k-means – dbscan+k-means)

Here we proceed to the implementation of the clustering algorithms DBSCAN, K-means and
(dk = dbscan+K-means) highlight the different clusters likely to contain sources of intrusion and attack in our architecture.

4.4.1. Process of applying the DBSCAN algorithm in our architecture

This process follows the following steps:

1) Connection to the database and download of the dataset

Here we connect to the database and we download the ToN-IoT dataset, which we divide into n centers (clusters).

2) Initializing the DBSCAN model:

This initialization of the DBSCAN algorithm involves determining the radius of a neighborhood to include data in a cluster and a minimum number of data required to form a cluster.

3) Fitting the model to the data:

Here we adjust the model to the data through normalization.

4) Cluster prediction:

Consists of returning the cluster labels for each data source. Sources with the -1 label are considered elements containing attacks or intrusions (outliers).

5) Displaying the number of clusters found:

The number of clusters is calculated by excluding intrusion sources.

6) Visualization of clusters found:

it allows you to visualize the sources colored by their cluster labels.

7) Display of cluster labels:

The cluster labels are displayed, where -1 corresponds to intrusion sources (containing attacks or intrusions).

The implementation of this process in python language gives the result of figure 7

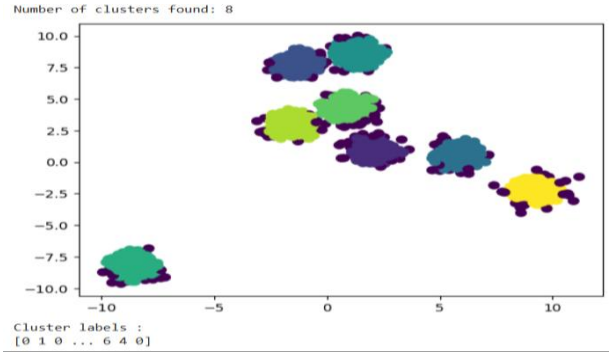


Fig. 6: DBSCAN Result.

4.4.2. Process of applying the K-means algorithm in our architecture

This process follows the following steps:

1) Connection to the database and download of the dataset

Here we connect to the database and we download the ToN-IoT dataset, which we divide into n centers (clusters).

2) Data visualization:

- plt. scatter allows you to visualize data sources.

3) Initialization and adjustment of the K-means model:

- Here we initialize the K-means algorithm with “m” clusters.
- Proceed to fit the model to the data.

4) Cluster prediction:

- Proceed with the prediction of clusters for each data source.

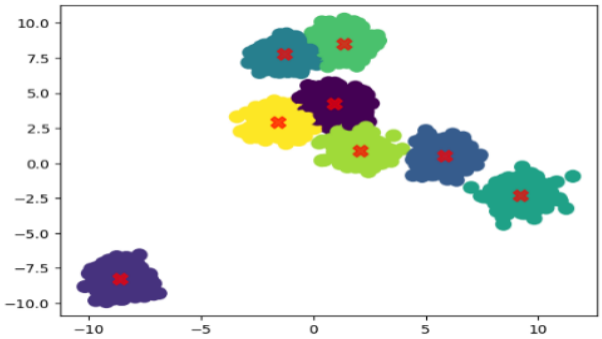
5) Visualization of clusters:

- Proceed to visualize the data colored by their predicted clusters.
- The centers of the clusters are also displayed with an X marker (in red in our case).

6) Display of cluster centers:

- Here we display the coordinates of the cluster centers.

The implementation in Python language gives the results of Figure 8



Cluster centers :
[[0.90773793 4.2655343]
[-8.59839451 -8.24027788]
[5.84515515 0.55113681]
[-1.28284945 7.79833281]
[9.22189738 -2.29719853]
[1.37539911 8.52040521]
[2.08840317 0.89362435]
[-1.58173296 2.94184471]]

Fig. 7: Result Of K-Means

4.4.3. Hybridization process of DBSCAN and k-means (DBSCAN + k-means)

This process follows the following steps:

1) Connection and extraction of the "ToN-IoT" dataset:

Here this step consists of using the reference dataset ("ToN-IoT") in terms of intrusion detection for clustering.

2) Application of DBSCAN :

Apply DBSCAN (α, β) to detect dense clusters and intrusion sources. With α = dense clusters and β the sources of intrusions

3) Identifying intrusion data sources :

- Data identified as sources of intrusion by DBSCAN are extracted for further processing.

4) Application of K-means :

- Here we apply kmeans to the intrusion sources detected by DBSCAN to assign these sources to the closest clusters.

5) Combination of results :

- The final labels combine the results of DBSCAN for data sources from the same clusters and K-means for intrusion sources.

Implementing this process in python gives the results of fig 9 and fig 10

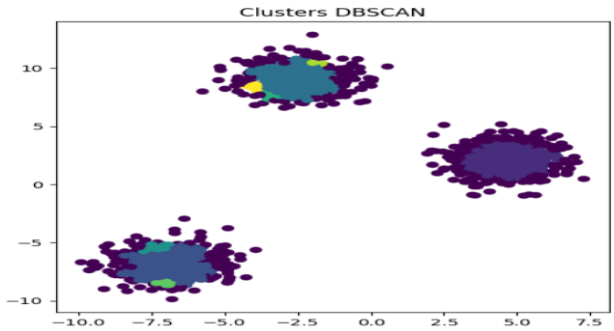


Fig. 8: Initial Clusters Found by DBSCAN.

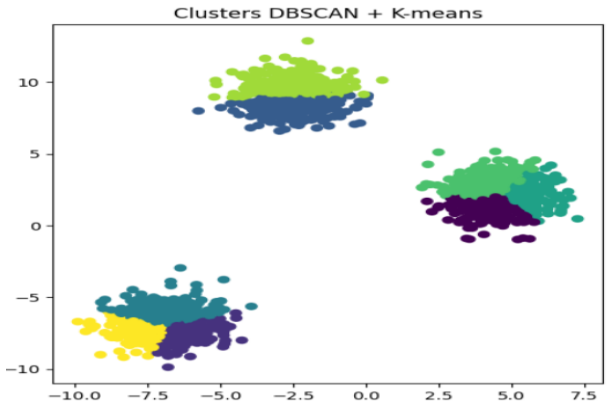


Fig. 9: The Clusters Refined After the Assignment of the Intrusion Sources to the Closest Clusters by K-Means.

5. Discussion

In this work, we have proposed an architecture to detect and predict intrusions in IoT environments based on big data. Concretely, our architecture presents 4 components which are:

- Capture tools
- The dataset
- Pretreatment
- The machine-learning algorithm used.

Adopting our architecture to detect intrusions could produce good results provided that the data used is of good quality (i.e. the normalization, clustering and splitting phase produces output data (equation e and 4) of good quality). Also, adversarial attacks could produce aberrant results (where malicious actors deliberately manipulate the input data to fool the detection system).

Regarding data collection, our architecture does not take into account the management of personal data. Which could be out of step with the regulations in this area. Some attackers could use this personal data to achieve their goals.

As for the maintenance of our architecture, it would largely depend on the machine learning algorithms used because they are complex to remain effective against attacks. To do this, continuous and frequent updating of these algorithms would allow our architecture to produce better results.

6. Conclusion and perspectives

In recent years, the diffusion of IoT devices across the world has progressed rapidly. Connected objects are now deployed in all areas such as health, smart cities, education, etc. To integrate this rapid commercialization flow, special attention has been given to the safety and

security of devices, IoT networks putting users at risk. Anything in turn disrupts the entire ecosystem connected to the Internet including websites, applications, social networks and servers.

The first part of our work laid the foundation, diving into the fundamentals of IoT , Big Data , and intrusion detection. This conceptual exploration established the framework necessary to understand the complex issues that characterize this environment. The second part delved deeper into the research by examining the state of the art of intrusion detection in IoT, identifying key threats and evaluating existing techniques. The correspondence between IoT, mobile applications and Big Data was analyzed, highlighting the importance of their convergence. The third part constituted the heart of our work, with the proposal of an intrusion prediction and detection architecture. The overall design and specific components have been detailed to provide an integrated and robust solution to cyber threats in this complex ecosystem.

The discussion of the results obtained through the study of Machine Learning in the context of IoT provided valuable insights. The performances of the different algorithms have been evaluated, paving the way for informed choices to improve the capabilities of intrusion detection systems. In sum, this work provides a comprehensive and in-depth view of intrusion detection in the dynamic world of IoT. The results, discussions and proposals formulated aim to guide researchers, practitioners and decision-makers in strengthening security within this vital ecosystem.

Future developments of this work could focus on the field of deep learning for intrusion detection, but also a hybridization of dbscan and x-means in order to compare the results of this second approach to those of the present work.

References

[1] Dagorn, N. (2006). Détection et prévention d'intrusion: présentation et limites..

[2] https://inria.hal.science/inria-00084202/

[3] https://inria.hal.science/inria-00084202

[4] Kuwatly, I., Sraj, M., Al Masri, Z., & Artail, H. (2004, July). A dynamic honeypot design for intrusion detection. In The IEEE/ACS International Conference on Pervasive Services, 2004. ICPS 2004. Proceedings. (pp. 95-104). IEEE. https://doi.org/10.1109/PERSER.2004.1356776.

[5] Bhagat , N., & Arora , B. (2018, December). Intrusion detection using honeypots . In 2018 Fifth International Conference on Parallel , Distributed and Grid Computing (PDGC) (pp. 412-417). IEEE. https://doi.org/10.1109/PDGC.2018.8745761.

[6] Depren , O., Topallar , M., Anarim , E., & Ciliz , MK (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. Expert systems with Applications, 29(4), 713-722. https://doi.org/10.1016/j.eswa.2005.05.002.

[7] Guleria , D., & Chavan , M. K. (2012). A study and comparative analysis of conditional random fields for intrusion detection . International Journal of Research in Computer Science, 2(4), 31-38. https://doi.org/10.7815/ijorcs.24.2012.037.

[8] Affinito , A., Zinno , S., Stanco , G., Botta, A., & Ventre, G. (2023). The evolution of Mirai botnet scans over a six- year period period . Journal of Information Security and Applications, 79, 103629. https://doi.org/10.1016/j.jisa.2023.103629.

[9] Tournier, J., Lesueur, F., Le Mouël , F., Guyon, L., & Ben- Hassine , H. (2018, May). Audit of an IoT system by intrusion test. In RESSI 2018-Rendes-Vous de la Recherche et de l'Enseignement de l'Information Systems Security (pp. 1-3).

[10] Bouzoubaa , K., Taher , Y., & Nsiri , B. (2021). Predicting DOS-DDOS attacks : Review and evaluation study of features selection methods based on wrapper process . Int. J.Adv. Comput. Sci . Appl , 12(5), 132-145. https://doi.org/10.14569/IJACSA.2021.0120517.

[11] Kasinathan , P., Pastrone , C., Spirito , MA, & Vinkovits , M. (2013, October). Denial -of-Service detection in 6LoWPAN based Internet of Things . In 2013 IEEE 9th international conference on wireless and mobile computing , networking and communications (WiMob) (pp. 600-607). IEEE. https://doi.org/10.1109/WiMOB.2013.6673419.

[12] Jun, C., & Chi, C. (2014, January). Design of complex event-processing IDS in internet of things . In 2014 sixth international conference on measurement technology and mechatronics automation (pp. 226-229). IEEE. https://doi.org/10.1109/ICMTMA.2014.57.

[13] Surendar , M., & Umamakeswari , A. (2016, March). InDReS : An Intrusion Detection and response system for Internet of Things with 6LoWPAN. In 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET) (pp. 1903-1908). IEEE. https://doi.org/10.1109/WiSPNET.2016.7566473.

[14] Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., & Atkinson, R. (2017). Shallow and deep networks intrusion detection system: A taxonomy and survey. arXiv preprint arXiv:1701.02145.

[15] Nobakht , M. (2019). The internet of things : securing devices and user data (Doctoral dissertation, UNSW Sydney).

[16] Nobakht , M., Sivaraman , V., & Boreli , R. (2016, August). A host- based intrusion detection and mitigation framework for smart home IoT using OpenFlow . In 2016 11th International conference on availability , reliability and security (ARES) (pp. 147-156). IEEE. https://doi.org/10.1109/ARES.2016.64.