IoT Crawler with Behavior Analyzer at Fog layer for Detecting Malicious Nodes

Layla Albdour[1], Saher Manaseer[1] and Ahmad Sharieh[1]

[1]Computer science department, The University of Jordan, Jordan

***Abstract*:** The limitations in terms of power and processing in IoT (Internet of Things) nodes make nodes an easy prey for malicious attacks, thus threatening business and industry. Detecting malicious nodes before they trigger an attack is highly recommended. The paper introduces a special purpose IoT crawler that works as an inspector to catch malicious nodes. This crawler is deployed in the Fog layer to inherit its capabilities, and to be an intermediate connection between the things and the cloud computing nodes. The crawler collects data streams from IoT nodes, upon a priority criterion. A behavior analyzer, with a machine learning core, detects malicious nodes according to the extracted node behavior from the crawler collected data streams. The performance of the behavior analyzer was investigated using three machine learning algorithms: Adaboost, Random forest and Extra tree. The behavior analyzer produces better testing accuracy, for the tested data, when using Extra tree compared to Adaboost and Random forest; it achieved 98.3% testing accuracy with Extra tree.

***Keywords*:** Cloud Computing, Crawler, Fog Computing, Internet of Things (IoT), Machine learning, Malicious node, Security.

1. Introduction

Detecting malicious nodes in internet of things (IoT) before they trigger an attack on the nodes is highly recommended. A special purpose IoT crawler that works as an inspector to catch malicious nodes helps in solving such security problem. This paper presents the IoT crawler which is deployed in the Fog layer to inherit its capabilities, and to be an intermediate connection between the things and the cloud computing nodes. Upon a priority criterion, this crawler collects data streams from the IoT nodes. It has a behavior analyzer with a machine learning core that detects malicious nodes according to the extracted node behavior from the collected data streams.

In the early development of IoT, authors in [1] presented the definition of IoT: “A world where physical objects are seamlessly integrated into the information network and where the physical objects can become active participants in business processes. Services are available to interact with these 'smart objects' over the Internet, query their state and any information associated with them, considering security and privacy issues.” Security and privacy must be well-thought-out in designing IoT systems.

IoT is a well-known technology that is used to connect a wide range of different devices such as sensors, actuators and smart nodes with the ability to communicate with each other and make decisions with no need for human involvement [2]There are many IoT applications; such as: smart home, wearable’s, smart grid, connected car, connected health and many others [3]. According to Lee [4] IoT applications are categorized into three main types: monitoring and controlling, big data and business analytics, and information sharing and collaborating. In these categories, many challenges are faced due to characteristics of the things in the IoT such as resource constrained devices.

According to McKinsey bottom-up application analysis, IoT is gaining 11.1 trillion Dollar by the year 2025, which is 11 percent of the world economy. Moreover, they showed that one trillion IoT devices will be used by the year 2025[5]. IoT is not standing alone, it needs a huge backup bone which is Cloud Computing (CC). CC is significant and powerful processing and storing power that works hand by hand with the IoT to achieve its goals. The advantage of CC is the virtualized resources, quality of service (QoS) provisioning, security, and many other services. CC provides a set of services such as: data storage, CPU cycle, computer resources, and even security on the demand of a client/user [6][7]. The user only asks for a service without any direct management or involvement. These services are provided by many enterprises such as amazon and google. The CC services are categorized into three terms: Software as a Service (SaaS), Platform as a service (PaaS) and Infrastructure as a service (IaaS) [8].

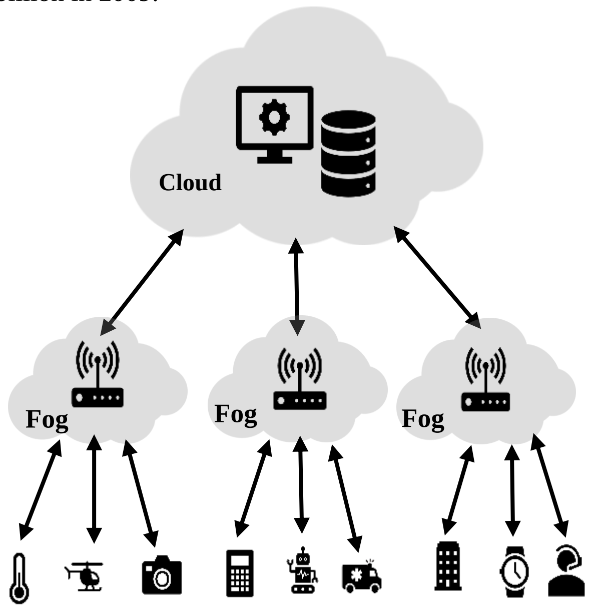
IoT devices are constrained not only due to its power limitations, memory, processing capabilities, and low power, but also radio limits the network interface too[9]. IoT data streams that are generated from widespread sensors are sent to a (CC) for further analysis and decision making. CC response time is not adequate for real time application. The blessing of the CC became a curse for applications that requires real time response. Messages passing and longtime trip from the ground to the CC consume more resources from the things [10].

To overcome the problem of the constrained devices and long messages delay, CISCO in 2012 was the first proposing Fog Computing (FC)[11]. FC is a new platform that extends the CC services to the edge of the network, i.e. a closer CC to the ground is called a fog computing. FC, like the CC, is a virtualized platform which provides several services such as: computation, storage, and connecting the end users with the CC traditional servers.

FC gained new characteristics over CC[12] that make it an attractive platform. These include a) Low latency, b) Geographical distribution over a wide area, c) Location awareness, d) Huge number of nodes, e) Wireless access is overriding, f) Supporting real time applications, g) Robust video streaming, and h) Heterogeneity. Those benefits of the FC make it favorable to deploy many applications that require geo distribution, low latency, and high mobility.

Figure 1 illustrates the relation between fog and cloud computing in serving the IoT. As you can notice, a fog in FC is an intermediate between the cloud in CC and the end users. Thus, it eliminates a disadvantage that happens due to the distant of CC.

IoT is growing rapidly; according to [13], number of IoT devices will reach 26 billion by 2020, growing from 0.9 billion in 2009.



IoT devises/Sensors

Figure 1. Fog computing in serving IoT

Many challenges arise due to the huge diversity and volume for the IoT. One of the major challenges is maintaining security and trusted operations to satisfy users. Network security should protect the networked things against attackers who aim to modify or endanger operation conducted in the things. Authors in [14] stated that the major security challenges face the IoT is: a) Authentication, b) Authorization and access control, and c) Privacy. Authentication means identifying legitimate users, who have the permission to access protected programs and resources. Authorization is the specification for access right for various resources and access control techniques must force a predefined restriction on resources. IoT data privacy is much more different from other data privacy, because IoT is collecting a very sensitive data about people and sometimes without even they notice.

The IoT constrained devices properties make it vulnerable for attacks. The lack in processing power and memory in IoT makes installing security methods and procedures very hard. A security breach would affect the IoT system, steal critical information, alter or steal data streams; or an attack threats the ecosystem and human lives. IoT security protection should overlay different IoT system components and deal with the heterogeneity of the system. Moreover, it should be scalable. Scalability means as more IoT devices added, the system should operate correctly [15].

Authors in [16] demonstrated a layered architecture for IoT, which consists of four layers (perception layer, network layer, support layer, and application layer). They recommend that security solutions should be deployed for the various levels to gain inclusiveness security for an IoT system. At the perceptual layer, different kinds of sensors gather data about unique objects from the real world. The responsibility of the network layer is to send gathered data streams from the perceptual layer to any system responsible for processing those streams, through the existing network protocols (wired or wireless). Support layer is responsible to take automated action depending on the result of the processed data and provide storage capabilities for the collated data. At the application layer, a smart application, upon users' needs, can be provided such as: smart home, smart transportation, smart farm etc.

The perceptual layer consists of devices and sensors. The fog layer is the network layer. The cloud layer is top two which are the support and the application layers. The fog layer (which refers to the network layer) is the backbone that connects different IoT devices in the lowest level and connects them to the highest level, which is the CC[17]. Security challenges in an IoT system can be seen on the different layers, because of the characteristics of each layer different threats occur [18].

Table 1 summarizes some of the challenges and threats on every layer.

Table 1. IoT security challenges

Layer	Attack	Ref
Perception Layer	Unauthorized Access to the Tags	[19]
	Tag Cloning	[20]
	Eavesdropping	[21]
	Spoofing	[22]
	Radio Frequency (RF) Jamming	[23]
Network Layer	Sybil Attack	[24]
	Sinkhole Attack	[25]
	Sleep Deprivation Attack	[26]
	Denial of Service (DoS) Attack.	[27]
	Malicious code injection.	[28]
	Man-in-the-Middle Attack	[29]
Support layer	Unauthorized Access	[30]
	DoS Attack	[31]
	Malicious Insider	[32]
Application Layer	Malicious Code Injection	[33]
	DoS Attack	[34]
	Sniffing Attack	[35]

Crawlers are computer programs that visit the web pages and download them for the purpose of Internet searching or cashing the web page for search engines [36].

The main contributions of our work are developing a smart fog computing crawler to collect IoT data streams based on a priority, implement a behavior analyzer to detect malicious nodes according to IoT collected data streams, and investigate the performance of the behavior analyzer using three machine learning algorithms, which are: Adaboost, Random forest and Extra tree.

The rest of the paper is organized as follows. Section 2 delineates the related works. Section 3 demonstrates the proposed IoT crawler and its priority model. The behavior analyzer is described in section 4. Experimental results are shown and discussed in section 5 Section. The last section concludes the paper and suggests the future work.

2. Related Works

According to [37] security and privacy requirements for an IoT system are resilience to attack, data authentication, and access control and user privacy. Resilience to attack means that an IoT system should be designed to be against single point of failure, data authentication means checking the integrity of data and its origin. Access control on provided user's data is a must and the client's privacy protection are highly required in every successful IoT system. Authors in [38] stated that security requirements for an IoT system are confidentiality, integrity, authentication, authorization/access control and availability. Table 2 gives a brief description for those requirements. The security requirements for IoT systems are increasing over time.

Table 2. IoT security requirements

Security requirement	Description	Ref
Confidentiality	Only authorized users can access resources	[39]
Integrity	Protecting data from unauthorized modification	[40]
Authentication	An IoT node is checked to be what it claims.	[41]
Authorization and access control	Ensuring that authorized users are using resources as they supposed to and in a proper way.	[42]
Availability	To make sure that IoT resources are always available as users are promised.	[43]
Energy efficiency	No irregular power consumption shall be allowed.	[44]

The main objective of IoT is making decisions depending on data collected by the sensors and analyses this data for actions and future predictions. IoT constrained devices exchange sensitive and critical data. Data privacy and security is a major issue in these multibillion industries. Many solutions for gaining secure IoT and systems are presented in the literature [45][46][47][48][49][50]. IoT security threats are categorized into two classes according to [51]. The first class is the conventional security threats that any network ecosystem is vulnerable to. These threats are Confidentiality, Integrity, and Availability (CIA). However, because of the IoT heterogeneity, diversity, and huge growing enumeration, the security threats are much more complex and severe. The second class of threats arises because of the sensitivity of data that IoT devices collect which are more personal and dynamic. Huge number of widespread devices collect big data which is considered sensitive. For example, this data could be about personal spaces, health status, and geographical location. Attackers may reveal personal privacy by extracting this data. For dealing with new security threats in IoT systems, old solutions are not enough. New solutions should deal with huge variety and volume of IoT devices and should be scalable to span on wide range IoT systems.

Authors in [52] presented an IoT intrusion detection system, they called it SVELTE. The system detects sinkhole and selective forwarding attacks. The overhead of their system is small and suitable for IoT constrained nodes. They implemented and evaluated the system in Contiki OS; the true positive rate they achieved didn't reach 100%.

Authors in [53] proposed a trust management protocol for IoT. The protocol addresses misbehaving nodes that change their behavior dynamically. In their work, every node autonomously executes trust evaluation with a formal treatment for the convergence, accuracy, and resilience properties for the dynamic management protocol. The authors worked according to the assumption that good nodes execute the trust management protocol properly while a malicious node provides false trust recommendations. They used a trust-based service composition application for the protocol evaluation purposes. Their evaluation results show a maximum performance according to a ground truth status. Moreover, the proposed protocol is adaptable to dynamically changing environment and use three trust properties for evaluation which are honesty, cooperativeness, and community-interest.

A Network Intrusion Detection System is proposed in [54] to detect policy violations or malicious nodes in IoT. The proposed approach is based on conditional variation auto-encoder, where they integrated the intrusion labels in the decoder layers. They reported that the proposed method is less complex than unsupervised methods and the results for classification are better than other classifiers. They tested their classifier on a host or host's network, and they concluded that it could recover missed features with a high accuracy.

In [55] they proposed a network-based approach to detect IoT bots' attacks. They made a behavior snapshot of traffic for every IoT device to extract statistical features. These features are used as input for deep learning techniques to detect anomalies. They trained an auto-encoder (one for each device) to learn the normal behavior of the IoT device. The benefit of using auto encoders is ability to learn complex patterns. Their results hardly detect false alarm. In[56], authors proposed a new intrusion detection scheme for Sybil attack in IoT. The scheme is lightweight. They combined Fuzzy C Means Algorithm (FCM) and Principal Component Analysis (PCA) algorithm. Their simulation showed that the new method improved the detection accuracy with a less false positive rate.

In[57], authors presented three mechanisms for intruder detection in IoT, the proposed systems use trust management model. Three attacks against Routing Protocol for Low power and Lossy networks (RPL) are detected which are: Selective-Forwarding attacks, Sinkhole attacks, and Version Number attacks. In [58], they proposed an IoT/Fog attack detection system using distributed deep learning. They designed the system for distributed systems such as smart cities and used performance metrics such as detection rate, and accuracy false alarm rate to show that deep model outperforms are shallow ones. Moreover, they showed that distributed model performs better than centralized one because sharing parameters avoids local minima.

In [59], they proposed a method to detect malicious applications depending on energy consumption patterns. They used unique local fingerprint for non-malicious nodes for detection and distinguishing malicious nodes. In[60] they presented a system to detect compromised IoT nodes, by using self-learning technique to classify devices according to communication patterns. Their proposed system is completely autonomous and can adapt with new devices types and new attacks too.

In [61], authors proposed an unsupervised learning technique to detect malicious nodes that manipulate transmitted packets from source to destination. In their work, they identified suspicious nodes in IoT multi hop forwarding transmission paths. They clustered members in the network according to their behavior into three groups based on their suspicious level. They used two detection modes, hard detection and soft detection. Both [62][63] discussed IoT systems security and privacy and how to deploy fog layer to improve the security solutions for its attractive properties by distributing a certificate revocation for the IoT devices.

In the context of IoT, authors in [64] have improved the Internet web crawler to obtain data from IoT nodes to determine the application and the node version. By knowing the application and node version, IoT vulnerabilities can be managed.

Authors in [65] proposed an IoT crawler that is called Cross Node Driven Search (CND) to extract knowledge from IoT nodes. The crawler differentiates between IoT nodes according to some criteria which makes it superior to classical crawling strategies such as: Breadth-First Search (BFS) [66], crawling online social graphs, and Random Walk (RW) [67].

Many search engines for IoT and Web of Things (WoT) have been proposed in the literature [68][69][70][71]. The existing IoT crawlers are mainly used for knowledge extraction, and the crawlers were not used to investigate the IoT nodes for security reasons. Thus, our proposed crawler works as a security inspector that watches the IoT nodes and collects data streams to analyze the behavior of those nodes.

3. Smart IoT Fog Crawler

IoT crawler is responsible for visiting the IoT nodes in a cyclic manner according to a priority model we call it ThingsRank. IoT nodes are part of a very huge population. These nodes are not in the same level of importance. For instance, if a node is a small sensor collecting weather data along with other sensors, it is not as important as an actuator controlling a facility door entrance. We are presenting the IoT nodes ranking as an importance level. If the node is important, the crawler would visit it more than a less important IoT node. The reason is that important node security breaches affect the IoT ecosystem more than just attacking a small sensor collecting redundant data.

An IoT network can be represented as a graph. This will lead to a better visualization how the nodes are connected and what do we mean by the importance level. Things-graph is like the one presented in Figure 2. This graph is undirected; thus, a link exists between two things if and only if they are connected via a communication link. The connected things can be presented as an adjacency matrix too.

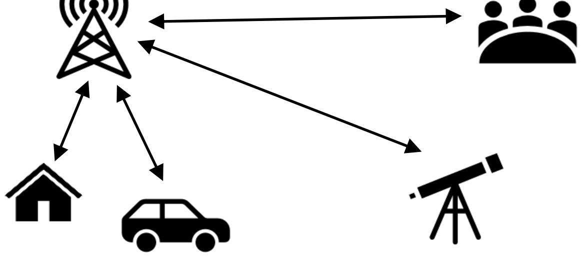


Figure 2. An example of Things-graph

An undirected graph is a pair *G= (V, E),* where *V* is the set of vertices (or nodes) and *E* is the set of edges connecting the vertices in *G*. In an undirected graph of things, the nodes are the IoT nodes and the edges are the connections between the things. A graph that represents the connection between things is called Things-Graph. Things-Graph, as in Figure.3, demonstrates every edge that is connecting two things is an unordered pair *e={u,v}* or *e={v,u}*. This definition implies that the graph does not contain self-loops and it satisfies the connections of the IoT. The undirected graph also represents symmetric relations[72].

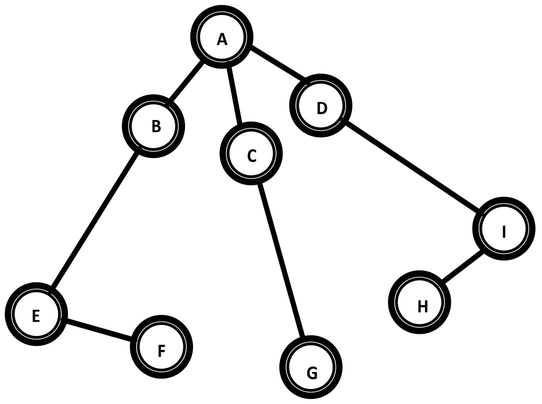


Figure 3. An example of abstract simple Things-Graph, where the nodes represent things and the edges represent connections.

Taking a deeper look into the IoT, all the nodes in the graph must be connected to a master node (i.e. a gateway or any other Fog device) either directly or through other vertices. No node can work alone, this would allow for more grouping the nodes in order to gain a better controlling on them. In other words, every IoT node must join at least one group, where the group head/master is a Fog node.

The ThingsRank for a device in an IoT, mathematically, is expressed by function (1). The ThingsRank of a device *u*, that is *R(u,t)*, depends on the ThingsRank of the devices that are directly connected to it at time *t*, divided by *C(v)*. The *C(v)*denotes the number of devices connected to thing *v*; if the connection is bidirectional; where *Bu* is the set of all devices connected to *u*.

$$R(u,t) = \sum\nolimits_{v \in Bu} \frac{R(v)}{C(v)} \qquad (1)$$

A damping factor is added because a thing that has no connection to other things may not be reached by the crawler. Thus, things with no connection are assumed to be connecting to all other things, and their ranks are divided equally between all other things. The value of the damping factor according to [73] is 0.85 and some Bayesian analysis recommends the optimal value of *d* (damping factor) to be 0.31 [74]. The equation using the damping factor would be as in function (2).

$$R(T_i,t) = \frac{1-d}{N} + d\sum\nolimits_{T_j \in M(T_i)} \frac{R(T_j)}{C(T_j)} \qquad (2)$$

Where $R(T_i,t)$ is the rank of a thing *i*, for *i*= *1, 2, 3, …,N at time t*, *N* is the number of things at time *t*, $M(T_i)$ is set of things that are connected to thing number *i*,, and $C(T_j)$ is the number of connections to thing T_j.

Initially (at *t=0*), all the things (nodes) are gaining the same rank as in function (3).

$$R(T_i;\ 0) = \frac{1}{N}\ ,\ for\ i = 1,2,3,\ldots,N \qquad (3)$$

The crawler is walking as a Markov Chain; the next

movement only depends on the current state of the crawler not on a sequence of previous events. The Markov process is a stochastic model that gives a description of events sequence. In this Markov model, the next state only depends on the current state, thus it is called memory less model[75].
In the World Wide Web (WWW), context web crawler (spider) is software responsible for visiting the web pages and indexing them for the search engines [76]. In our study, the IoT crawler visits the IoT nodes and eavesdrops on the network traffic data to analyze it as a legality behavioral measurement.
As we mentioned earlier visiting the nodes is not trivial. It is done according to an importance level which is called ThingsRank as in function (2). The Web search engine works in three steps: crawling, indexing and ranking. In our approach, we are working according to other three steps, which are: ranking, crawling, and sniffing traffic. You can notice the difference which is the visiting method. In the web crawling, visiting is done according to search algorithms such as breadth first search and depth first search, and after visiting the page and indexing it; a rank is assigned to it.
In the IoT crawler, algorithm visiting is done according to the node importance (i.e. ThingsRank). The most important nodes are tested first if they are exhibiting a normal behavior or not. The group head maintains a list of its member that is updated in a cyclic manner. In this list, each node in IoT is assigned a priority according to ThingsRank algorithm. According to CISCO IoT reference model [77] the connectivity layer is right above the things in IoT and the IoT crawler is deployed in the Fog layer to gain its pros.
Ranking algorithm updates the ThingsRank for a Things-Graph in a cyclic manner since the adjacency matrix is changing dynamically over time. The summation of the IoT nodes rank within the group should be one. Initially, all the things are gaining the same rank as in function (3). The algorithm in the group head calculates the new ThingsRank according to Algorithm1.

```
Algorithm 1: ThingsRank($\breve{A}$)
    Input: $\breve{A}$ representing a set of IoT nodes connected to a Fog device
    Output: An $N$-element array of $R$ which represent the rank for the
            IoT nodes in $\breve{A}$
1  BEGIN
2  Let $X$ be an array of $N$ elements
3  for  $i$=0 to $N$-1 do
4      $X[i] = 1/N$
5      endfor
6  $d$ is a damping factor 0<$d$<1
7  Let $R$ be an $N$-element of array
8  repeat
9      for  $i$=0 to $N$-1 do
10         $R[i] = 1 - d$
11         endfor
12     Let $C_N$ be the number of connected nodes to node $\breve{E}$
13     for  all nodes $\breve{E}$ that are connected to $R[i]$ do
14         $R[i] = R[i] + d * X[\breve{E}]/C_N$
15         endfor
16     for  $i$=0 to $N$-1 do
17         $X[i] = R[i]$
18         endfor
19 until  $t$ finishes;
20 END
```

ThingsRank(Ă) is the algorithm that is responsible for ranking IoT nodes within a group. The input for this algorithm is a set of IoT nodes, and the output is rank for those nodes.

Initially, for a new group, the rank value is partitioned evenly as inline 4 calculations, and then the algorithm modifies the rank value depending on number of connected things to every node in the group periodically, line 14. The new rank values for the set of nodes, for loop in line 16, are used in the next iteration. Algorithm 2 shows the pseudocode for the IoT crawler.

```
Algorithm 2: IoT Crawler($\breve{A}$)
    Input: An $N$-element array of $R$ which represent the rank for the IoT
           nodes in set $\breve{A}$
    Output: An $N$-element array of Records (DST) which contain IoT
            nodes data streams
1  BEGIN
2  Sort ($R[i..N-1]$)
3  for  $i$=0 to $N$-1 do
4      DST[i] = Sniff $\breve{A}$ [R(i)]
5      endfor
6  END
```

In Algorithm 2, the IoT *Crawler* (*Ă*), the ranked nodes are sorted according to their rank values, then the crawler collects the things data stream (Sniff) starting from the most important node to the least important node. An array of records of data streams is the output for this algorithm. These data streams are the input for the behavior analyzer. Calling *ThingsRank(Ă)* and *Crawler* (*Ă*) algorithms are used according to a cyclic manner within a predefined time period.
Detecting a malicious node is done by measuring its behavior; this is so called behavioral Intrusion Detection Systems (IDS)[78]. The output from the crawler is an input for the behavior analyzer that is also deployed in a FC node. As Figure.4 illustrates, the detecting of a malicious node is done by a behavior analyzer. The behavior analyzer should identify the malicious node according to its behavior. Then, the system should take an action regarding the malicious node such as shutting down the system or blocking the node(s).

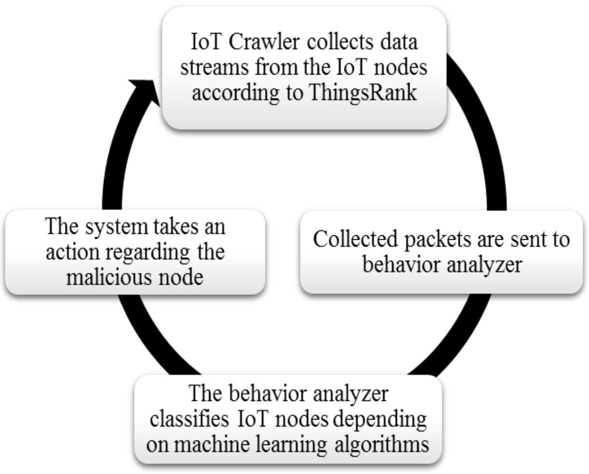

IoT Crawler collects data streams from the IoT nodes according to ThingsRank
Collected packets are sent to behavior analyzer
The behavior analyzer classifies IoT nodes depending on machine learning algorithms
The system takes an action regarding the malicious node

Figure 4. Cycle of malicious node capturing approach

4. Behavior Analyzer

As mentioned earlier, the behavior analyzer is responsible for analyzing captured data streams and differentiates malicious nodes from the legitimate ones. Figure.5

demonstrates the proposed behavior analyzer workflow. The captured data streams are analyzed using a machine learning technique. The behavior analyzer then gives a binary decision 0 if the node is legitimate and 1 if the node is malicious.

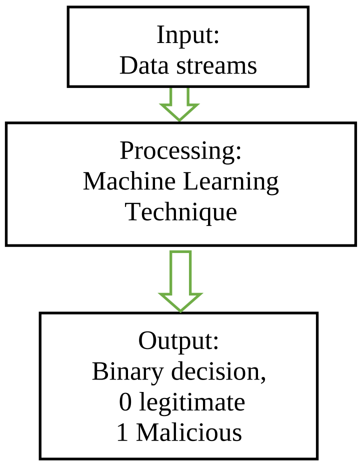


Figure 5. Working flow of the behavior analyzer

In this proposed behavior analyzer, we investigate the deployment of three different machine learning techniques: Random Forest, Extra Tree and AdaBoost. Next, we will give a brief description of each technique.
Machine learning according to [79] means that a computer can learn from prior experience. In machine learning, the system can reach the data and learn from it. For instant, if the system has a prior knowledge about the behavior of the malicious node, then it should be able to detect malicious nodes to maintain the security of an IoT system. Machine learning is divided into supervised and unsupervised learning. In supervised learning, the system is trained using labeled data which are tagged with the ground truth. On the other hand, in unsupervised machine learning, the machine is trained using unlabeled data. For example, finding patterns and similarities are examples of clustering in unsupervised learning.
AdaBoost, short for adaptive boosting, is a supervised machine learning technique, it was first proposed in 1995 by Freund and Shapire [80]. The magic of AdaBoost is its ability to combine a set of weak classifiers to produce a strong one. It is a fast convergence easy implemented algorithm. Adaboost generates a set of weak learners by generating a set of weights over the training data, then after each cycle, the maintained weights are adjusted adaptively. The misclassified training samples weights will be increased, and on the other side, the weights of the correctly classified training samples will be decreased. The main idea of Adaboost is maintaining a distribution of weights over the training data samples. Initially, all the weights are distributed evenly, then, after each iteration, the weights are adjusted according to the classification output.
Random Forest is an ensemble learning method that consists of large number of decision trees [81] with each tree in the forest, a class prediction is made; but the tree with the most votes wins and becomes the model prediction. The concept of the random forest is simple and robust. The uncorrelated models that are presented in the trees are a committee with performance that is superior to any other individual model. The produced trees have a low correlation and each tree is protected from the other trees' errors. In other word, if some trees are wrong, others would still operate in the right direction.
Extra tree is another ensemble learning method based on decision tree [82] it is a low variance random forest. It builds multiple random trees depending on random features. The enhancement that the extreme tree has made on the random forest is that it makes no bootstrapping; this means samples are done without replacement. In extra tree, the splits are random with no best splits; all observations are extremely random splits to minimize over fitting (over learning).

5. Experimental results

5.1 Benchmark Dataset

Variety benchmark datasets are used for intrusion detection evaluation such as: KDD98, KDDCUP99 [83] and NSLKDD [84]. But, unfortunately, they are not recently generated, and those datasets do not reflect low footprint threats and IoT traffic. UNSW-NB15 is a modern benchmark dataset created in 2015; it is hybrid of real and synthetic traffic to mimic attack or malicious behavior [85]. We used UNSW-NB15 dataset to evaluate the behavior analyzer system.
The UNSW-NB15 dataset consists of 2.5 million data point and 47 different features that are categorized into different five groups. Two target labels exist: one is the data point attack and the other is the attack category. The dataset contains nine attack categories with a tenth no attack category. They are listed in Table 3 with their record distribution.

Table 3. Dataset attack categories

Category	No. of records
Normal	2,218,761
Fuzzers	24,246
Analysis	2,677
Backdoors	2,329
DoS	16,353
Exploits	44,525
Generic	215,487
Reconnaissance	13,987
Shellcode	1,511
Worms	174

The UNSW-NB15 benchmark dataset contains all new attacks information according to Common Vulnerabilities and Exposures (CVE)[1]. Nine types of attacks are found in the dataset, they are mentioned in Table 3 with their records distribution. Following are brief descriptions of those attacks.
Fuzzing attack is based on fuzz testing which is a dynamic testing method. The attack involves feeding a random invalid and unexpected data to a node software or hardware until it crashes. The attacker may flood the buffers or overflows an integer in order to cause system crash, memory leakage, or any system fault [86]. In analysis attack, the intruder eavesdrops to the communication lines between two nodes and analyzes it to lunch different types of attack such as: port scan attack, spam and HTML break through [87]. Backdoor attack [88] is a technique of stealing the system security to gain an access to a system or its data. Once the attacker gains access to the system, he/she can steal financial credentials or

[1] https://cve.mitre.org

any other important data. The attacker furthermore would install malware for future attacks, it is considered as the fourth common attack.

In Denial of Service (DoS) [89] attack, the intruder tries in different ways to consume the network resources to block out legitimate users in the network. For instance, flooding the traffic to consume bandwidth or CPU time is an example. Many illegitimate nodes (infected with a Trojan) compromise the system, therefore blocking a single IP address is not adequate to stop the attack. In exploits attack, the attacker discovers a system vulnerability, then takes the advantage of this flaw to attack the system either by using a malicious file or gaining higher privileges to install further malware [90].

Reconnaissance attacks are about gathering information about a system user, a well-known reconnaissance attack is social engineering [91]. Shellcode is a small piece of executable code used to attack a system through vulnerability. The shellcodes are used to remotely download a malware and infect the computer system software/hardware [92]. Worms is a self-replicating malware that attack the system through security holes then it spread itself through the network [93], the worms can damage the files within the system or steal/change the user credentials.

The data types for the 47 features of UNSW-NB15 are different such as: Boolean, integer, float, nominal and timestamps. For instance, is-ftp-login is a Boolean feature; it is 1 if the ftp (File Transfer Protocol) session is accessed by the user with a password and 0 otherwise. Source to destination packet count and destination to source packet count are examples of integer features. Most of the timestamp features are floats such as the time between the SYN and the SYN-ACK packets of the TCP. The attack categories and protocol types are examples for nominal data points.

The 47 features of the UNSW-NB15 dataset are divided into five groups: flow features, basic features, content features, time features and additional generated features. There are two additional labeled features, which are attack category and a label for each record that is 1 if it is attack and 0 otherwise. Preprocessing is made to convert the nominal values to numerical values to be properly used in the behavior analyzer.

5.2 Performance metrics

Detecting a malicious node is considered a binary classification problem [94], a node is malicious, or it is legitimate and there is no third option. A binary classifier is used for Network Intrusion Detection System (NIDS) detection. Performance metrics used to measure the performance of NIDS are True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) [95]. Table 4 explains these four-performance metrics along with an example on every metric.

The TP and TN are the wanted results from NIDSs, while the false ones are incorrect results that are not recommended. Confusion metric is a table that represents the four-performance metrics as shown in Table 5. It has two dimensions which are the actual and the predicted ones [96].

From the previous confusion matrix, a set of evaluation metrics can be derived, which are accuracy, precision, sensitivity, specificity, F1 score, Receiver Operator Characteristic (ROC) and Precision-Recall (PR) [97].

Accuracy, measures the effectiveness of the IDS algorithm by showing the probability of the true positive values which gives an overall assessment for the IDS effectiveness. See function (4).

Table 4. Network Intrusion Detection Systems Performance Metrics

Performance metric	Description	Example	Visualization
TP	Classifier correctly labels positive ones.	A malicious node is detected correctly by the system.	😈 👍
TN	Classifier correctly labels negative ones.	A legitimate node is detected correctly.	😇 👍
FP	Classifier incorrectly labels positive ones as negative.	A legitimate node is considered malicious incorrectly.	😇 ▥
FN	Classifier labels positive ones with negative incorrectly.	A malicious node passes the system as legitimate.	😈 ⊘

Table 5. Confusion matrix

	Actual Positive	**Actual Negative**
Predicted Positive	TP	FP
Predicted Negative	FN	TN

$$Acuuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (4)$$

Precision measures the predictive capability of the IDS algorithm; it is also called positive predictive value, as shown in function (5).

$$Precision = \frac{TP}{TP + FP} \qquad (5)$$

Sensitivity or recall is the true positive rate which measures the actual intrusions that are predicted by the algorithm, as in function (6).

$$Recall = \frac{TP}{TP + FN} \qquad (6)$$

Specificity is the true negative rate which measures the actual legitimate nodes that are predicted by the algorithm, see function (7).

$$Specificity = \frac{TN}{FP + TN} \qquad (7)$$

As in function (8), F1 score is a precision/recall function. In other words, it measures the balance between precision and recall.

$$F1 = 2 * \frac{(precision.recall)}{(precision + recall)} \qquad (8)$$

Both sensitivity and specificity measure the effectiveness of the IDS on a single class. The ROC curve demonstrates the relation between the two. ROC curve is a preferable way for visualizing the IDS performance and to choose an operating point or a suitable decision threshold. PR (precision-recall) curve demonstrates the relation between precision and recall; it measures the performance of a system at different thresholds [98].

5.3 Experimental Results

Experiments were made using Python 3.7 and tensorflow 2.0. Class 0 means no attack and class 1 means an attack (malicious node). Three machine learning algorithms, which are Adaboost, Random forest and Extra tree, were employed. The testing accuracy for the used machine learning algorithms are shown in Figure.6, the results imply that when the crawler runs with the Extra Tree testing, Random forest, and AdaBoost, the average accuracies are 98.3%, 97.7%, and 80.2%, respectively. The results were averages of five runs and rounded to three significant digits. The results of the five tests of each algorithm were very close. Extra tree outperforms the Random forest because it uses all the learning samples rather than using replicas. In the Extra tree cut-points are fully random; no best split is made as in random forest.

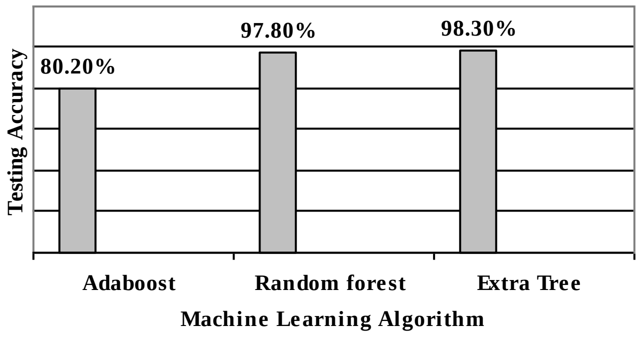


Figure 6. Testing Accuracy using Adaboost, Random forest and Extra tree algorithms

Table 6. Class specific results for attack detection for precision, Recall and F1-Score on UNSW-NB15 dataset using the Adboost, Random, and Extra Tree machine learning algorithms.

Algorithm	Class	Precision	Recall	F1-Socre
AdaBoost	0	1.00	0.72	0.84
	1	0.32	0.98	0.48
Random Forest	0	1.00	0.98	0.99
	1	0.84	0.99	0.91
Extra Tree	0	1.00	0.99	0.98
	1	0.88	1.00	0.93

The results in Table 6 show that the three algorithms have 1.00 precision for class 0 predicting (normal traffic). This implies that the three algorithms can predict no attack class, thus a legitimate node would not be considered malicious. For predicting class 1(malicious traffic), extra tree algorithm has the highest precision over AdaBoost and Random forest with 88%, while AdaBoost precision is the lowest with 32% value. The lowest value for class 0 recall among the three algorithms is 72% for Adaboost algorithm; the highest is for Extra tree with 99%, while random forest recall for random forest is 98%. Class 1 recall values are high among the three algorithms; they are ranging from 98% to 100%. F1 scores for class 0, for the three algorithms, are higher than F1 scores for class 1. Adaboost F1 scores for class 0 and class1 are the lowest among the other two algorithms, with 84% and 48% values respectively. The reason why Extra tree overall results were the best is that the splitting reduces the variance by averaging the deep decision trees and randomly choosing a value for the trees split to boost the performance of the last decision tree.

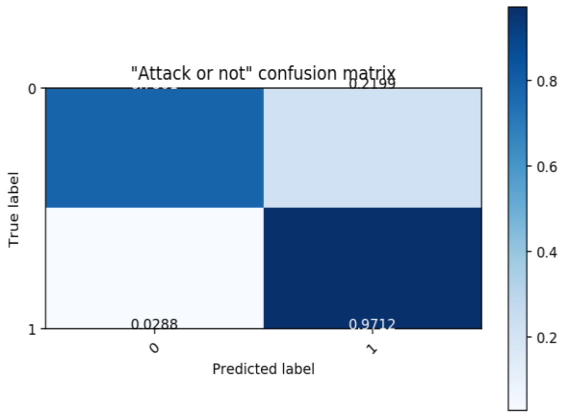
Figure 7 shows the confusion matrix for the three algorithms. Extra tree and random forest results are almost similar for predicting a malicious node (TP) with 99% rate, while Adaboost is the lowest with 97% TP rate.

Figure 8 demonstrates the ROC curves class 1 prediction for Adaboost, Random Forest and Extra tree, respectively. The largest the area the better the result would be. Notice that the AUC (Area Under the Curve) for Extra tree is 99.6%, which is the largest among the three algorithms. This implies that Extra tree outperforms random forest and AdaBoost. In summary, all the conducted results from the simulation (testing accuracy, precision, recall, F1 score, confusion matrix, ROC and AUC) for the Extra tree are always the best. Extra trees are extremely randomized in both the attributes and cut point, while generating the decision trees, using all the learning samples without replicas generates no bios in the extra tree performance.

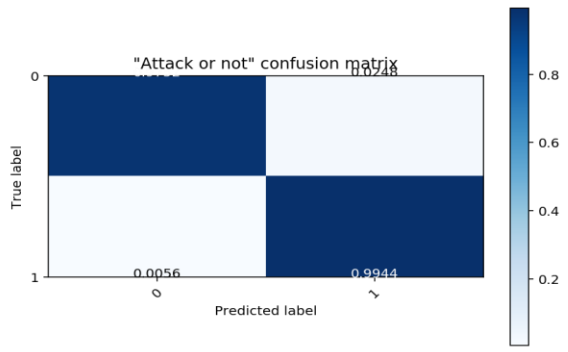
Table 7 shows a comparison between our work and other related work in term of the accuracy. Authors in [99] used different machine learning algorithms on the same dataset, the UNSW-NB15 dataset. Overall, the table shows that the accuracy results for our selected techniques are better than the techniques used in. For instance, our best result was the Extra tree classifier with 98.3% accuracy, while their best result was for the Decision tree with 85.6% accuracy. Our selected classifiers results are better because they are ensemble learning algorithms, which means that they used different learning techniques and select (vote) for the best technique, beside that Random forest and Extra tree can deal with the high complexity of the selected dataset.

Table 7. Comparison between different machine learning techniques on UNSW-NB15 dataset in term of accuracy

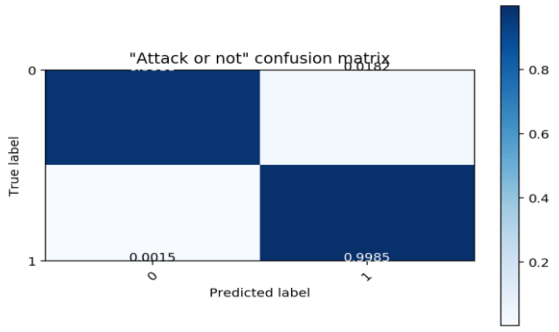
Our Result		Results in [99]	
Technique	Accuracy(%)	Technique	Accuracy(%)
Adaboost	80.2	Naive Based	82.1
Random forest	97.8	Decision tree	85.6
Extra Tree	98.3	Logistic Regression	83.2



(1) AdaBoost

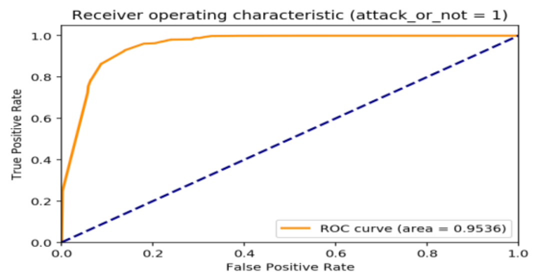


(2) Random forest

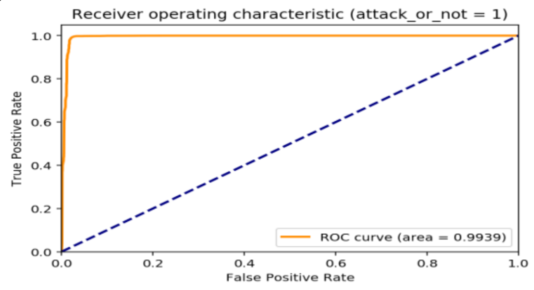


(3) Extra tree

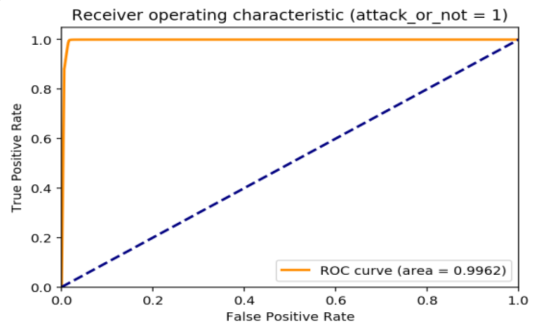
Figure 7. Adaboost, Random forest and Extra tree confusion matrices for attack prediction on UNSW-NB15 dataset



(1) AdaBoost



(2) Random forest



(3) Extra Tree

Figure 8. Adaboost, Random forest and Extra tree ROC curves for attack prediction

6. Conclusion

Securing IoT systems is highly required and published literature lack special dedicated systems for IoT security. We presented an intelligent IoT crawler; and we think it is the first crawler that collects IoT data streams from IoT nodes to analyze it. The IoT crawler walks in a predefined priority criterion. In other words, visiting the IoT nodes is not done trivially. The most influencing and important nodes are visited first to check their legibility. Checking and investigating nodes are done by a behavior analyzer that takes the IoT collected data stream as an input.

We used ensemble machine learning techniques to detect malicious nodes; Extra tree algorithm showed the best results with 98.3% accuracy. Our results for malicious detection, for the selected classifiers, are better than other results in term of accuracy. An action shall be taken regarding a malicious node such as shutting down the system or further investigating in a suspected node. Deploying the IoT crawler and the behavior analyzer is done in a Fog Computing node to avoid the IoT constrained node limitations. In the future, we will increase the number of attacks that the behavior analyzer can detect, and we hope to be able to recognize the zero-day attack.

References

[1] N. Gershenfeld, R. Krikorian, and D. Cohen, “The Internet of Things,” *Sci. Am.*, vol. 291, no. 4, pp. 76–81, Oct. 2004.

[2] M. Ambrosin *et al.*, “On the Feasibility of Attribute-Based Encryption on Internet of Things Devices,” *IEEE Micro*, vol. 36, no. 6, pp. 25–35, Nov. 2016.

[3] S. Misra and S. Sarkar, “Theoretical modelling of fog computing: a green computing paradigm to support IoT applications,” *IET Networks*, vol. 5, no. 2, pp. 23–29, Mar. 2016.

[4] I. Lee and K. Lee, “The Internet of Things (IoT): Applications, investments, and challenges for enterprises,” *Bus. Horiz.*, vol. 58, no. 4, pp. 431–440, Jul. 2015.

[5] J. Bughin, M. Chui, and J. Manyika, “An executive’s guide to the Internet of Things,” *McKinsey Quarterly*. pp. 92–101, 2015.

[6] A. Sharieh and L. Albdour, “A Heuristic Approach for Service Allocation in Cloud Computing,” *Int. J. Cloud Appl. Comput.*, vol. 7, no. 4, pp. 60–74, Oct. 2017.

[7] A. Al-Shaikh, H. Khattab, A. Sharieh, and A. Sleit, “Resource Utilization in Cloud Computing as an Optimization Problem,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 6, 2016.

[8] R. Abu Khurma, H. Al Harahsheh, and A. Sharieh, “Task scheduling algorithm in cloud computing based on modified round robin algorithm,” *J. Theor. Appl. Inf. Technol.*, vol. 96, p. 17, 2018.

[9] A. Sehgal, V. Perelman, S. Kuryla, and J. Schonwalder, “Management of resource constrained devices in the internet of things,” *IEEE Commun. Mag.*, vol. 50, no. 12, pp. 144–149, Dec. 2012.

[10] A. Hudaib and L. Albdour, “Fog Computing to Serve the Internet of Things Applications,” *Int. J. Fog Comput.*, vol. 2, no. 2, pp. 44–56, Jul. 2019.

[11] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the internet of things,” in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC ’12*, 2012, p. 13.

[12] L. M. Vaquero and L. Rodero-Merino, “Finding your Way in the Fog,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014.

[13] Gartner, “Gartner Says the Internet of Things Will

Transform the Data Center," *Gartner Newsroom*, 2014. .

[14] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, "Internet of Things security and forensics: Challenges and opportunities," *Futur. Gener. Comput. Syst.*, vol. 78, pp. 544–546, Jan. 2018.

[15] I. Yaqoob *et al.*, "Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges," *IEEE Wirel. Commun.*, vol. 24, no. 3, pp. 10–16, Jun. 2017.

[16] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the Internet of Things: A Review," in *2012 International Conference on Computer Science and Electronics Engineering*, 2012, pp. 648–651.

[17] H. H. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha, and K.-K. R. Choo, "A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks," *IEEE Trans. Emerg. Top. Comput.*, vol. 7, no. 2, pp. 314–323, Apr. 2019.

[18] M. U.Farooq, M. Waseem, A. Khairi, and S. Mazhar, "A Critical Analysis on the Security Concerns of Internet of Things (IoT)," *Int. J. Comput. Appl.*, vol. 111, no. 7, pp. 1–6, Feb. 2015.

[19] R. Uttarkar and P. R. Kulkarni, "Internet of Things : Architecture and Security," *Int. J. Comput. Appl.*, vol. 3, no. 4, pp. 12–19, 2014.

[20] H. Kamaludin, H. Mahdin, and J. H. Abawajy, "Clone tag detection in distributed RFID systems," *PLoS One*, vol. 13, no. 3, p. e0193951, Mar. 2018.

[21] B. Khoo, "RFID as an Enabler of the Internet of Things: Issues of Security and Privacy," in *2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing*, 2011, pp. 709–712.

[22] A. Mitrokotsa, M. R. Rieback, and A. S. Tanenbaum, "Classifying RFID attacks and defenses," *Inf. Syst. Front.*, vol. 12, no. 5, pp. 491–505, Nov. 2010.

[23] Lan Li, "Study on security architecture in the Internet of Things," in *Proceedings of 2012 International Conference on Measurement, Information and Control*, 2012, pp. 374–377.

[24] M. A. Jan, P. Nanda, X. He, and R. P. Liu, "A Sybil attack detection scheme for a forest wildfire monitoring application," *Futur. Gener. Comput. Syst.*, vol. 80, pp. 613–626, Mar. 2018.

[25] N. Ahmed, S. S. Kanhere, and S. Jha, "The holes problem in wireless sensor networks," *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 9, no. 2, p. 4, Apr. 2005.

[26] T. Bhattasali, R. Chaki, and S. Sanyal, "Sleep Deprivation Attack Detection in Wireless Sensor Network," *Int. J. Comput. Appl.*, vol. 40, no. 15, pp. 19–25, Feb. 2012.

[27] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A Survey on Wireless Security: Technical Challenges, Recent Advances, and Future Trends," *Proc. IEEE*, vol. 104, no. 9, pp. 1727–1765, Sep. 2016.

[28] S. Ozdemir and H. Cam, "Integration of False Data Detection With Data Aggregation and Confidential Transmission in Wireless Sensor Networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 736–749, Jun. 2010.

[29] A. Mallik, A. Ahsan, M. M. Z. Shahadat, and J.-C. Tsou, "Man-in-the-middle-attack: Understanding in simple words," *Int. J. Data Netw. Sci.*, pp. 77–92, 2019.

[30] L. Zhou and H.-C. Chao, "Multimedia traffic security architecture for the internet of things," *IEEE Netw.*, vol. 25, no. 3, pp. 35–40, May 2011.

[31] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks," *IEEE Commun. Surv. Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[32] A. Y. Khan, R. Latif, S. Latif, S. Tahir, G. Batool, and T. Saba, "Malicious Insider Attack Detection in IoTs Using Data Analytics," *IEEE Access*, vol. 8, pp. 11743–11753, 2020.

[33] A. P. Fournaris, G. Keramidas, K. Ispoglou, and N. Voros, "VirISA: Recruiting Virtualization and Reconfigurable Processor ISA for Malicious Code Injection Protection," in *Components and Services for IoT Platforms*, Cham: Springer International Publishing, 2017, pp. 117–130.

[34] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," *Expert Syst. Appl.*, vol. 148, p. 113249, Jun. 2020.

[35] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of Things: Security vulnerabilities and challenges," in *2015 IEEE Symposium on Computers and Communication (ISCC)*, 2015, pp. 180–187.

[36] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan, "Searching the Web," *ACM Trans. Internet Technol.*, vol. 1, no. 1, pp. 2–43, Aug. 2001.

[37] R. H. Weber, "Internet of Things – New security and privacy challenges," *Comput. Law Secur. Rev.*, vol. 26, no. 1, pp. 23–30, Jan. 2010.

[38] S. F. Tan, A. Samsudin, and S. Alias, "Internet of Things: Security Challenges and Its Future Direction," in *Lecture Notes in Electrical Engineering*, 2019, pp. 483–488.

[39] A. Ouaddah, A. Abou Elkalam, and A. Ait Ouahman, "FairAccess: a new Blockchain-based access control framework for the Internet of Things," *Secur. Commun. Networks*, vol. 9, no. 18, pp. 5943–5964, Dec. 2016.

[40] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things," in *2015 IEEE World Congress on Services*, 2015, pp. 21–28.

[41] J. Liu, Y. Xiao, and C. L. P. Chen, "Authentication and Access Control in the Internet of Things," in *2012 32nd International Conference on Distributed Computing Systems Workshops*, 2012, pp. 588–592.

[42] Q. Liu, H. Zhang, J. Wan, and X. Chen, "An Access Control Model for Resource Sharing Based on the Role-Based Access Control Intended for Multi-Domain Manufacturing Internet of Things," *IEEE Access*, vol. 5, pp. 7001–7011, 2017.

[43] A. Boudguiga *et al.*, "Towards Better Availability and Accountability for IoT Updates by Means of a Blockchain," in *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 2017, pp. 50–58.

[44] M. Suárez-Albela, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo, "A Practical Evaluation of a High-Security Energy-Efficient Gateway for IoT Fog Computing Applications," *Sensors*, vol. 17, no. 9, p. 1978, Aug. 2017.

[45] M. Elhoseny, G. Ramirez-Gonzalez, O. M. Abu-Elnasr, S. A. Shawkat, N. Arunkumar, and A. Farouk, "Secure Medical Data Transmission Model for IoT-Based Healthcare Systems," *IEEE Access*, vol. 6, pp. 20596–20608, 2018.

[46] S. Chakrabarty and D. W. Engels, "A secure IoT architecture for Smart Cities," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2016, pp. 812–813.

[47] K. Muhammad, R. Hamza, J. Ahmad, J. Lloret, H. Wang, and S. W. Baik, "Secure Surveillance Framework for IoT Systems Using Probabilistic Image Encryption," *IEEE Trans. Ind. Informatics*, vol. 14, no. 8, pp. 3679–3689, Aug. 2018.

[48] D. Dragomir, L. Gheorghe, S. Costea, and A. Radovici, "A Survey on Secure Communication Protocols for IoT Systems," in *2016 International Workshop on Secure Internet of Things (SIoT)*, 2016, pp. 47–62.

[49] S. Al-Azzam, A. Sharieh, A. Sleit, and N. Al-Azzam, "Securing robot communication using packet encryption distribution," *Netw. Secur.*, vol. 2018, no. 2, pp. 8–14, Feb.

2018.

[50] R. Bremananth and A. Sharieh, "Crytosystem for Computer security using Iris patterns and Hetro correlators," *Int. J. Adv. Comput. Sci. Appl.*, vol. 2, no. 11, 2011.

[51] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT Security: Ongoing Challenges and Research Opportunities," in *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, 2014, pp. 230–234.

[52] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, Nov. 2013.

[53] F. Bao and I.-R. Chen, "Dynamic trust management for internet of things applications," in *Proceedings of the 2012 international workshop on Self-aware internet of things - Self-IoT '12*, 2012, p. 1.

[54] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Conditional Variational Autoencoder for Prediction and Feature Recovery Applied to Intrusion Detection in IoT," *Sensors*, vol. 17, no. 9, p. 1967, Aug. 2017.

[55] Y. Meidan *et al.*, "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul. 2018.

[56] L. Deng, D. Li, X. Yao, D. Cox, and H. Wang, "Mobile network intrusion detection for IoT system based on transfer learning algorithm," *Cluster Comput.*, vol. 22, no. S4, pp. 9889–9904, Jul. 2019.

[57] Z. A. Khan and P. Herrmann, "A Trust Based Distributed Intrusion Detection Mechanism for Internet of Things," in *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, 2017, pp. 1169–1176.

[58] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Futur. Gener. Comput. Syst.*, vol. 82, pp. 761–768, May 2018.

[59] A. Azmoodeh, A. Dehghantanha, M. Conti, and K.-K. R. Choo, "Detecting crypto-ransomware in IoT networks based on energy consumption footprint," *J. Ambient Intell. Humaniz. Comput.*, vol. 9, no. 4, pp. 1141–1152, Aug. 2018.

[60] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A Federated Self-learning Anomaly Detection System for IoT," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 756–767.

[61] X. Liu, M. Abdelhakim, P. Krishnamurthy, and D. Tipper, "Identifying Malicious Nodes in Multihop IoT Networks Using Diversity and Unsupervised Learning," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6.

[62] A. Alrawais, A. Alhothaily, C. Hu, and X. Cheng, "Fog Computing for the Internet of Things: Security and Privacy Issues," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 34–42, Mar. 2017.

[63] S. Ghildiyal, A. Semwal, and S. Kumar, "Enhancing Security of Internet of Things (IoT) using Fog Computing," *SSRN Electron. J.*, 2019.

[64] W. Li, Xuemeng and Wang, Yongyi and Shi, Fan and Jia, "Crawler for Nodes in the Internet of Things," *ZTE Commun.*, vol. 3, p. 009, 2015.

[65] G. Baldassarre, P. Lo Giudice, L. Musarella, and D. Ursino, "The MIoT paradigm: Main features and an 'ad-hoc' crawler," *Futur. Gener. Comput. Syst.*, vol. 92, pp. 29–42, Mar. 2019.

[66] S. Ye, J. Lang, and F. Wu, "Crawling Online Social Graphs," in *2010 12th International Asia-Pacific Web Conference*, 2010, pp. 236–242.

[67] K. Avrachenkov, B. Ribeiro, and D. Towsley, "Improving Random Walk Estimation Accuracy with Uniform Restarts," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010, pp. 98–109.

[68] H. Ma and W. Liu, "A Progressive Search Paradigm for the Internet of Things," *IEEE Multimed.*, vol. 25, no. 1, pp. 76–86, Jan. 2018.

[69] T. Maekawa, Y. Yanagisawa, Y. Sakurai, Y. Kishino, K. Kamei, and T. Okadome, "Context-aware web search in ubiquitous sensor environments," *ACM Trans. Internet Technol.*, vol. 11, no. 3, pp. 1–23, Jan. 2012.

[70] N. K. Tran, Q. Z. Sheng, M. A. Babar, and L. Yao, "Searching the Web of Things," *ACM Comput. Surv.*, vol. 50, no. 4, pp. 1–34, Aug. 2017.

[71] B. Christophe, V. Verdot, and V. Toubiana, "Searching the 'Web of Things,'" in *2011 IEEE Fifth International Conference on Semantic Computing*, 2011, pp. 308–315.

[72] Y. Liu, T. Safavi, A. Dighe, and D. Koutra, "Graph Summarization Methods and Applications: A Survey," *ACM Comput. Surv.*, Dec. 2016.

[73] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *Comput. Networks ISDN Syst.*, vol. 30, no. 1–7, pp. 107–117, Apr. 1998.

[74] R. E. Kass, W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, "Markov Chain Monte Carlo in Practice.," *J. Am. Stat. Assoc.*, vol. 92, no. 440, p. 1645, Dec. 1997.

[75] O. Mezghani and P. M. Abdellaoui, "Efficient clustering protocol based on stochastic matrix & MCL and data routing for mobile Wireless Sensors Network," *Int. J. Commun. Networks Inf. Secur.*, vol. 10, no. 1, pp. 188–198, 2018.

[76] F. Menczer, G. Pant, and P. Srinivasan, "Topical web crawlers," *ACM Trans. Internet Technol.*, vol. 4, no. 4, pp. 378–419, Nov. 2004.

[77] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review," *J. Comput. Commun.*, vol. 03, no. 05, pp. 164–173, 2015.

[78] S.-Y. Ji, B.-K. Jeong, S. Choi, and D. H. Jeong, "A multi-level intrusion detection method for abnormal network behaviors," *J. Netw. Comput. Appl.*, vol. 62, pp. 9–17, Feb. 2016.

[79] E. Alpaydin, *Introduction to Machine Learning 2nd*. 2010.

[80] N. Freund, Yoav and Schapire, Robert and Abe, "A Short Introduction to Boosting," *Journal-Japanese Soc. Artif. Intell.*, vol. 14, no. 771–780, p. 1612, 2009.

[81] A. Liaw and M. Wiener, "Classification and Regression by randomForest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[82] O. Maier, M. Wilms, J. von der Gablentz, U. M. Krämer, T. F. Münte, and H. Handels, "Extra Tree forests for sub-acute ischemic stroke lesion segmentation in MR sequences," *J. Neurosci. Methods*, vol. 240, pp. 89–100, Jan. 2015.

[83] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6.

[84] D. a. M. S. Revathi, "A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection," *Int. J. Eng. Res. Technol.*, 2013.

[85] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.

[86] L. Ouairy, H. Le-Bouder, and J.-L. Lanet, "Protection of Systems Against Fuzzing Attacks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019, pp. 156–172.

[87] M. V. Pawar and J. Anuradha, "Network Security and Types of Attacks in Network," *Procedia Comput. Sci.*, vol.

48, pp. 503–506, 2015.
[88] Y. Ji, X. Zhang, and T. Wang, “Backdoor attacks against learning systems,” in *2017 IEEE Conference on Communications and Network Security (CNS)*, 2017, pp. 1–9.
[89] G. Vasconcelos, R. S. Miani, V. C. Guizilini, and J. R. Souza, “Evaluation of DoS attacks on commercial Wi-Fi-based UAVs,” *Int. J. Commun. Networks Inf. Secur.*, vol. 11, no. 1, pp. 212–223, 2019.
[90] K. Thyagarajan and A. Thangavelu, “An integrated defense approach for distributed denial of service attacks in mobile ad-hoc network,” *Int. J. Appl. Eng. Res.*, vol. 11, no. 7, pp. 4898–4910, 2016.
[91] M. Uma and G. Padmavathi, “A survey on various cyber attacks and their classification,” *Int. J. Netw. Secur.*, vol. 15, no. 5, pp. 390–396, 2013.
[92] T. Okamoto, “SecondDEP: Resilient Computing that Prevents Shellcode Execution in Cyber-Attacks,” *Procedia Comput. Sci.*, vol. 60, pp. 691–699, 2015.
[93] M. A. I. Almarshad, M. M. Z. E. Mohammed, and A. S. K. Pathan, “Detecting zero-day polymorphic worms with jaccard similarity algorithm,” *Int. J. Commun. Networks Inf. Secur.*, vol. 8, no. 3, p. 203, 2016.
[94] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov, “Learning and Classification of Malware Behavior,” in *Detection of Intrusions and Malware, and Vulnerability Assessment*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 108–125.
[95] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan, “An improved method to construct basic probability assignment based on the confusion matrix for classification problem,” *Inf. Sci. (Ny).*, vol. 340–341, pp. 250–261, May 2016.
[96] J. Grau, I. Grosse, and J. Keilwagen, “PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R,” *Bioinformatics*, vol. 31, no. 15, pp. 2595–2597, Aug. 2015.
[97] K. Hajian-Tilaki, “Receiver Operating Characteristic (ROC) Curve Analysis for Medical Diagnostic Test Evaluation.,” *Casp. J. Intern. Med.*, vol. 4, no. 2, pp. 627–35, 2013.
[98] R. G. Lehr and A. Pong, “ROC Curve,” in *Encyclopedia of Biopharmaceutical Statistics*, Informa Healthcare, 2010, pp. 1185–1191.
[99] N. Moustafa and J. Slay, “The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set,” *Inf. Secur. J. A Glob. Perspect.*, vol. 25, no. 1–3, pp. 18–31, Apr. 2016.