ISSN: 2251-9246
EISSN: 2345-6221

pp. 99:111

A Survey on Botnet Detection Methods in IoT

Taha Shojarazavi, Hamid Barati*, Ali Barati

Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran, hbarati@iaud.ac.ir

Abstract

Today, the Internet of Things (IoT) is expanding due to a wide range of applications and services. The variety of devices connected to the Internet has made discussing security in these networks a challenging issue. Security includes various aspects such as botnets. Botnets are a collection of devices such as smartphones, computers, and other devices infected by a program. This program, which is a herder bot, performs many harmful operations and leads to various anomalies in the network. Identifying botnets is one of the main challenges in IoT security due to their unique complexity. In this article, we have reviewed the botnet detection methods in IoT. Since there are different botnet detection methods in IoT, we need to do detailed research on different botnet detection methods and their strengths and weaknesses. In a way that shows the evolution of these malwares. Concepts such as life cycle, command and control models, communication protocols, botnet protocols, and botnet detection methods are described in this research. In the following, the advantages and disadvantages of botnet detection methods are discussed and these methods are compared.

Keywords: Internet of Things, Botnet detection, True positive, Security, Feature extraction
Article history: Received 2023-04-01; Revised 2023-04-20; Accepted 2023-05-13.

https://doi.org/10.30495/ijsee.2023.1982835.1261

1. Introduction

Internet of Things networks includes different types of devices that connect to the Internet and use its services [1]. This technology has created many opportunities in different fields [2]. One of the applications of this category of networks is in the smartening of homes, where household appliances are connected to the Internet and provide remote control capabilities [3]. In addition to smart homes, this technology can be used in health systems [4]. In this way, wearable devices are connected to sensors that collect body information and send it to the central server for further processing [5]. Also, Internet of Things networks are used in other fields such as industry [6], automobile [7], smart cities [8], creating decision support systems [9], etc. In the Internet of Things environments, there are different types of devices with different applications and very heterogeneous physical equipment [10]. These heterogeneities have turned the security issue in Internet of Things networks into a fundamental challenge [11]. Security is one of the different aspects of this category of networks, one of which is bot networks [12]. Bots are internet robots, which are called bots for short [13]. A bot is a software program that automatically performs some tasks, including executing various scripts [14]. A bot network is also a collection of computers that run a bot or several instances [15-16]. Botnets can execute Distributed Denial of Services (DDOS) attacks, steal data, send spam and allow hackers to access devices and their connections [17-19]. Botnet owner or hackers send their commands to bots using commands and control [20]. There are different architectures for bot networks, the simplest of which is for a client computer causes it to fail by sending various requests to the server [21-23]. The solution to deal with this category of anomalies is simple, and this category of bots can be stopped by blocking the requests received from the client's computer [24]. In more complex cases, bots are used in a distributed manner. In general, there are two common models in this field, which are [25-26]:

– Client-server model: In this model of botnets, users' systems are infected when they use chat channels or visit some sites. A server monitors this group of botnets, and its owner sends its commands to the server. Then the server sends the desired commands to its subordinates to perform their duties.

– Peer-to-peer model: The client-server model is easily identifiable, and the bot network is controlled by controlling the server. Therefore, hackers and botnet owners have created a peer-to-peer model. In this category of networks, bot network owners' control and manage it in a distributed manner.

Identifying peer-to-peer networks is one of the main challenges in the security of Internet of Things networks due to their specific complexities [27]. The rest of this paper is organized as follows. In section 2, bot network is described, and comparison of advantages and disadvantages of botnet detection methods is presented. In section 3 introduces botnet construction mechanisms. In section 4, previous works for botnet detection in IoT are reviewed. Section 5 presents the evaluation and comparison of methods and describes the advantages and disadvantages of each method, and finally, in section 6, the conclusion is presented.

2. Bot networks

The word bot is derived from the word robot, and a bot network is several devices connected to the Internet, each of which runs one or more bot programs. Also, the word "botnet" is a combination of two words ", Robot" and "Network". A controller usually uses this network to take advantage of any of the following [28-29]:

– Distributed Denial of Service Attacks: Some disrupters of service providers cause the service provider to fail to serve other customers by sending repeated requests; this type of attack is called a denial of service attack. To deal with this undesirable situation, usually, when a customer sends repeated requests, it is temporarily blocked. However, disruptors continue to track their target by using bot networks and sending requests through different devices.
– Data theft: Bot network devices collect data by checking data traffic from different networks and providing them to the controlling person or system. These systems also extract other information from the collected data. Inferential databases are examples of these.
– Sending spam: In addition to the inconvenience they cause to the recipient, spam also consumes network bandwidth and increases traffic. Attackers sometimes disrupt network activities by sending many spam messages.
– Loss of privacy: Bot-infected devices expose device connections and data to the person or system controlling the bot network. In this way, the connections and data of the device are easily available to the desired person or system, and the confidentiality of the data is lost.

In table 1, a comparison has been made on the techniques of recently presented works, and the advantages and disadvantages of the works have been stated.

Table.1.
Comparison of Advantages and Disadvantages of botnet detection methods

Method category	*Advantages*	*Disadvantages*
Automatic deep auto encoder	No need for data pre-processing Lack of impact of outlier data on the model High sensitivity rate	Low accuracy Very high computational load High false negative rate
Auto encoder neural network	Ability to load on small hardware memories Ability to load on all kinds of IoT devices Fast execution time	Dependence on libraries Dependence on computing servers High false positive rate in distinguishing bot from non-bot traffic Dependence of detection power on the number of received packets
Neural network	Ability to detect anomalies related to hosts and network Low computing overhead on devices Minimum load imposed on the network High accuracy Can be used in unsupervised mode	Based on simulated data Strong dependence on input error High sensitivity to network and device changes Dependence on data transferred from another environment
Deep learning	Network based and high detection capability Feature extraction Very high accuracy rate (at best 99.8%) Low false positive rate	Very high duration of deep neural network training Abandoning some features in the pre-processing stage

3. Background

Based on understanding botnets' working mechanism and behavioral characteristics, this section introduces botnet construction mechanisms in terms of botnet architecture, lifecycle, and Command and Control (C&C) channel.

A) Architecture

Bot networks usually follow two architectures, which are [30-31]:

Client-server architecture: In this architecture, which is shown in Figure 1, the bot header, which is the person controlling the bot network, is responsible for loading and infecting devices connected to the Internet. Once this person is identified, infecting more systems and pursuing destructive goals will no longer be possible.

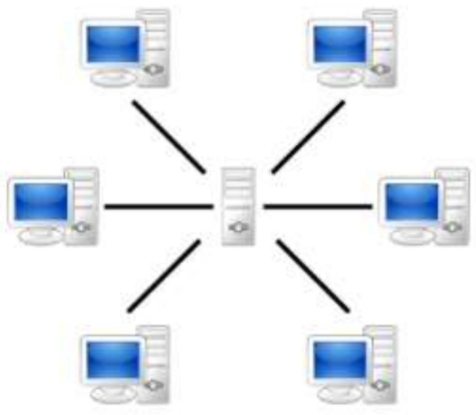

Fig. 1. Client-Server Architecture.

Peer-to-peer architecture: This type of architecture was presented after the previous architecture could easily identify and deal with the polluting person. In this architecture, all the infected devices are also responsible for the distribution and collection of data. In this case, dealing with these networks is very challenging, and their identification requires advanced techniques. An example of this category of networks can be seen in Figure 2.

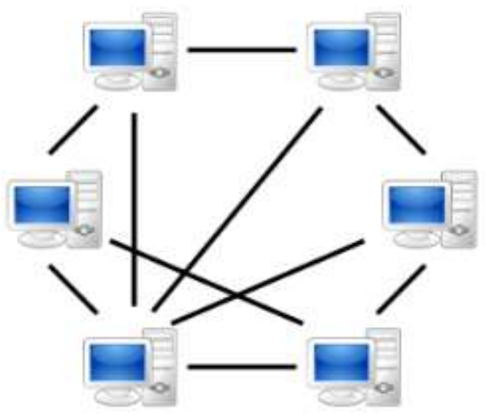

Fig. 2. An example of a peer-to-peer network.

B) Life Cycle

The life cycle of a botnet mainly includes propagation, rally, interaction and malicious activities [32].

Release: It can be used as a stand-alone program to expand bots. The main spreading methods include shared media spread, vulnerability exploitation spread, social engineering spread and password guessing.

Rally: It refers to robots' behavior in locating and controlling the server and its resources. Implementation methods are mainly divided into two categories: static and dynamic. Static addressing means that the C&C resources that bots try to access are fixed and unchanging. These resources are usually hard-coded into the bot's body or stored in a hidden path on the infected machine, such as the registry. Dynamic addressing means that the access address is not fixed but must be generated dynamically based on a specific algorithm.

Interaction: When the zombie host successfully discovers an existing command control server or resource, it communicates with the controller and starts interacting. This process is called the command control stage, which mainly includes four activities: registration, file download, order distribution, and result feedback.

Malicious attacks: An attacker's primary goal in building and controlling a botnet is to control many victim hosts to launch various attacks. Common attack activities include DDoS, spam, malware distribution, data leakage, click fraud, phishing attacks, data gathering, virtual currency mining, and encrypted ransom. (DDoS, spam, spreading malware, information leakage, click fraud, phishing attacks, information collection, virtual currency mining, and encrypted blackmail.)

C) The effect of the bot network on the Internet of Things

The emergence of the Internet of Things networks, in which various types of devices are connected to the Internet, has provided more space for bot network controllers. The review of the work done shows that the following damages can exist in Internet of Things networks [33]:

An unsafe intermediary: a weak intermediary causes information to be disseminated and placed in the hands of unauthorized persons.

Weak authentication: Since there is no necessary infrastructure for authentication and validation of requests, these systems have become vulnerable.

Insecure software: Insecure software, such as inappropriate antivirus, makes the bot software easily infect the device and network with bots and lowers security.

Insecure physical devices: Studies show that secondary and peripheral devices connected to computer systems load bot software and control the system.

4. Review of previous works

Bot networks are a collection of computers infected with hackers' software, which are unintentionally used for malicious purposes. Since there are all kinds of devices in the Internet of Things networks, identifying bot networks is considered one of their fundamental challenges. Various methods have been provided to deal with bot networks. However, a comprehensive comparison of them has yet to be presented. For this reason, in one of the conducted works, the researchers have compared the works presented to deal with bot networks. By comparing the done works, in addition to the application of each, their limitations and future directions are also determined. In comparing and reviewing the works done, a good data set has yet to be used to evaluate the methods [34]. Also, a particular method for identifying bots is not provided in the applications based on the framework's presentation. Which method and for which type of data can be suitable is one of the challenges related to these works. Another challenge raised in the modelling and identification of bot networks is data normalization and pre-processing. Knowing what method should be used in normalizing data or their pre-processing has always been a challenge. The last point is that to compare different tasks, different botnet scenarios, such as spam, distributed denial of service attacks, and peer-to-peer botnets, should be considered [35]. One of the challenges related to these works is which method suits which type of data. Another challenge raised in the modelling and identification of bot networks is data normalization and pre-processing. Knowing what method should be used in normalizing data or their pre-processing has always been a challenge. The last point is that to compare different tasks, different botnet scenarios, such as spam, distributed denial of service attacks, and peer-to-peer botnets, should be considered [36].

In the networks related to bots, the attackers try to achieve their malicious actions using different methods. For this purpose, botnets have a very high priority regarding security level in Internet networks, especially Internet of Things networks. In the following, some of the methods presented to detect botnets are described.

Conventional detection methods are no longer suitable for detecting new botnets. Botnet detection technology classification standards are different, and there are also multi-dimensional classification methods.

Yahyazadeh and Abadi [37] presented a machine-learning model to identify the bot network. The method presented by the researchers is based on the history of botnets and their types of activities, which can be seen in Figure 3. The way this method works is that first, the input stream is filtered. In the filtering part of the incoming stream, bots that have already been reported and are actually present in the database are sent to the malicious activity detection engine. Then they are sent to the bot detection engine. This engine performs report generation. If the entered data flow is not already available in the database, clustering is done on them. In clustering, the data are placed in two categories, suspicious and healthy. After receiving the reports, the bot detection engine produces the final report for the users. Although this method is effective for identifying botnets in the early stages, its main drawback is that they did not consider the noises in the construction of the model.

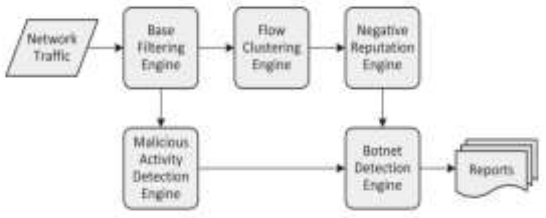

Fig. 3. Flowchart of botnet detection method.

Chen and Lin [38] stated that since botnets do not have a specific organization, their identification is considered a main challenge and one of the most critical security issues. One of the methods that botnets operate on is Internet Relay Chat (IRC). In this case, the bots use the channels created for chatting to achieve collective goals. A method whose general schematic can be seen in Figure 6 is presented to deal with IRC botnets. In this method, IRC traffic is received first, and then features such as time, number of requests, source IP, packet length, etc., are extracted from the relevant traffic. In order to detect bots, two groups of data are created; the first group is related to bot network data, and the second group is related to non-bot data. If the data is identified as part of a group related to bot networks, that group itself is divided into several categories and scored based on their importance. A method whose general schematic can be seen in Figure 4 is presented to deal with IRC botnets. In this method, IRC traffic is received first, and then features such as time, number of requests, source IP, packet length, etc., are extracted from the relevant traffic. In order to detect bots, two groups of data are created; the first group is related to bot network data, and the second group is related to non-bot data. If the data is identified as part of a group related to bot networks, that group itself is divided into several categories and scored based on their importance.

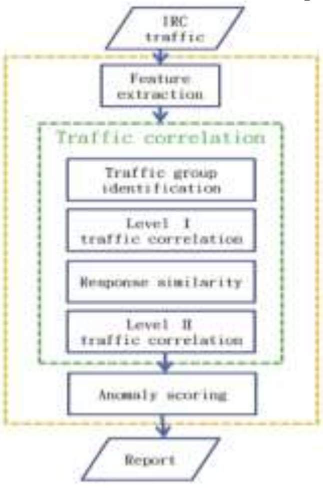

Fig. 4. Identification of botnets based on feature extraction.

Narang et al. [39] have presented a method to identify the bot network based on noisy data. In the presented method, the Fourier transform is first taken from the data; when the Fourier transform is taken from the data, the data is collected around their centre. Noisy data will be very far from the centrality of the data so that they can be discarded with a filter. After removing the noisy data, the corresponding model is built using machine learning techniques. The proposed method has produced acceptable results for noisy, training, and test data. Nevertheless, the model's accuracy has decreased in the methods that use noisy data to create the model.

Kapre and Padmavathi [40] stated that botnets try to disrupt or disable Internet of Things networks using various methods. For this reason, in this research, they have provided a tool to identify bot networks, the general schematic of which can be seen in Figure 5. The performance of the presented tool is such that this tool includes a user-friendly graphical interface and receives the data stream first. Then it analyzes the data stream and returns the result to the GUI. After analyzing the data flow, an analyzer analyzes them and returns the analytical results, and then the GUI reports the results of the analyzer to the user. Machine learning techniques have been used in the analysis section, and feature extraction methods have been used for analysis. After presenting their tool, the researchers evaluated it on different data sets and showed sufficient reliability.

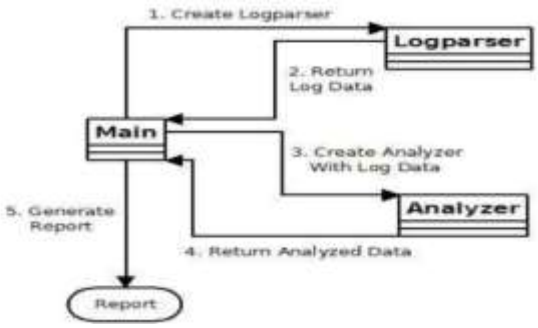

Fig. 5. Detecting botnets using event-based analysis.

Mathur et al. [41] separated botnet detection methods into real-time and non-real-time categories. In real-time methods, the type of network is recognized at a certain time, but in non-real-time methods, the time to distinguish the bot network may be very long. Since non-real-time methods require much time, researchers have developed a real-time method based on machine learning methods to identify bot networks. The general opinion of the researchers was that they first checked the existing features, then created a model based on the extracted features, and then compared their method with other methods. After presenting the proposed method, checking its details, and comparing them, researchers have used criteria such as true positive, false positive, true negative and false negative to compare the methods.

Wang et al. [42] stated that botnets contain a large amount of data, which is very time-consuming to process and build a working model. For this reason, using parallel processing techniques can significantly reduce the time of building the model and the detection of the bot network. Considering the advantages of parallel processing and the limitations of bot networks, they have presented a method based on Hadoop's parallel processing method. How this method works can be seen in figure 6. The working method is that the data stream files are first received and then distributed on HDFS. After distribution and processing, its output is combined based on the completion time. In the next step, the data that are not part of the bot network are discarded, and the data that are recognized as bot networks are sent to the next step, where they are grouped according to the type of threat and importance. The threat is carried out, and finally, based on the scoring done to each of the groups related to the bot network, a list of obtained IPs is provided.

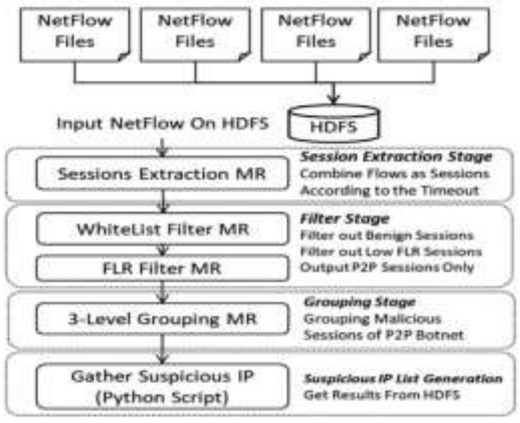

Fig. 6. The method of identifying botnets using clustering techniques.

Khanchi et al. [43] stated that many algorithms had been presented to detect the bot network, and their efficiency has been different. The limitation that most of them have is that when the data is unbalanced, the model's efficiency is also directed towards a specific data group. To deal with this limitation, this article presents a method based on genetic programming, whose general schematic can be seen in figure 7. The operation that is performed to detect the bot network is in such a way that, at first, the data stream related to the network is received, which may or may not be a part of the bot network. After receiving the data flow and in order to deal with unbalanced data, a sampling policy is used. After sampling, the data set is formed. The data set is formed so that almost both categories related to the bot and non-bot network include the same number of samples so that the created model does not tend towards a specific category of features. After forming the data set and based on the genetic algorithm operators, a number of features are selected, and a model is built using them. The relevant operations continue until an acceptable accuracy of the model is obtained.

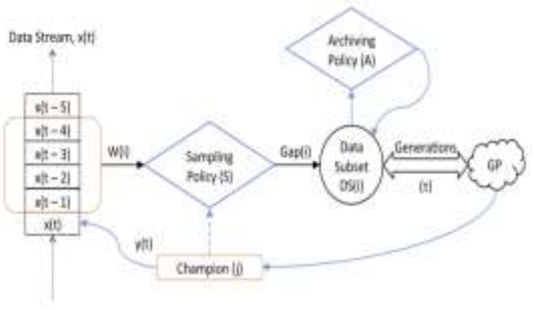

Fig. 7. Detection of botnets using genetic programming.

Cid-Fuentes et al. [44] stated that bot networks constantly change and manifest differently. The limitation of previous works was that they could identify bots that were reported in the past. However, they do not work for new bots. To deal with the raised problem, researchers have presented a decentralized framework whose general schematic can be seen in Figure 8. The way the presented framework works is that, at first, the behavior of the hosts is examined, and features such as the number of received messages, the length of the received messages, their type, their IP and … are extracted. In order to identify behaviors, machine learning techniques are used, and attempts are made to separate similar and dissimilar behaviors from each other. Suspicious behaviors are sent to the network administrator and based on the feedback received from the administrator, the bot network is distinguished from the non-bot. One of the limitations of the work done is that it involves the human user in distinguishing the bot network from the non-bot, which the human user himself may have many errors. This error can significantly affect the efficiency of detecting bots from non-bots.

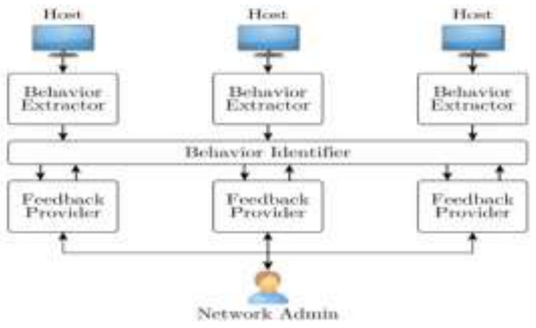

Fig. 8. The framework for identifying botnets.

HaddadPajouh et al. [45] used Recurrent Neural Network (RNN)'s deep learning method to detect Internet of Things malware. The object approach presented in this paper consists of three steps presented in Figure 9. First, they collect IoT malware samples to build a dataset and extract instructions. Then, a feature vector file is created for each sample based on the instructions. In the last step, vector data was used to train and evaluate the deep neural network and finally adjust for optimal results. In the first step, creating the dataset and extracting OpCodes is done. This article focuses on ARM-based IoT applications. In the feature selection stage, text mining was used to obtain the feature vector from the pruned instructions. A dictionary of all special instructions from the dataset is compiled. In the classification step, long short-term memory (LSTM), an RNN structure, is used to build a deep learning structure and identify IoT malware samples based on OpCode sequences. Also, Google Tensor Flow has been used as a support structure for the deep learning approach, and Scikit-learn as a machine learning library to perform model evaluation tasks.

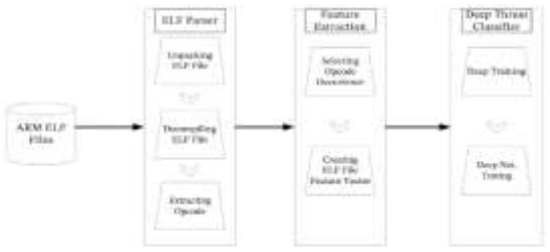

Fig. 9. Identification of botnets.

Azmoodeh et al. [46] presented a deep learning-based method to detect IoT malware through the OpCode sequence. In this method, OpCodes are converted into a vector space, and deep learning of a particular space is used to classify malicious and standard programs. This method creates a dataset of 1078 normal and 128 malware samples for ARM-based IoT applications. This method, as shown in Figure 10, consists of two stages: The opCode-Sequence diagram generation phase and Eigensapce deep learning phase. The feature selection phase is also included in this figure. Control Flow Graph (CFG) is a data structure that shows the order of OpCodes in an executable file. Using this method results in the generation of an adjacency matrix for each sample program in the dataset. In addition, the normalization of the matrix rows converts the values into the probability of occurrence of Vi.

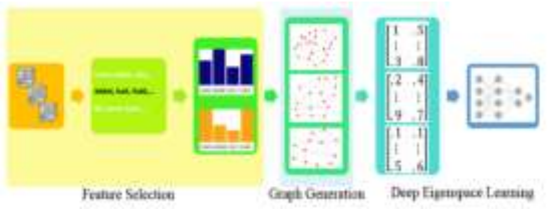

Fig. 10. Botnet identification steps in the reference.

Alhanahnah et al. [47] proposed an efficient signature generation method for IoT malware that generates recognizable signatures based on high-level structural, statistical, and string vectors. The generated signatures for each malware family can be used to develop lightweight malware detection tools for securing IoT devices. This paper attempts to overcome these challenges using a data-driven approach by examining a set of IoT malware examples in the real world. This method aims to identify malware samples from the same family that are similar in both codes and functions. This method takes a lightweight approach by capturing high-level code structure, code statistics, and string properties to develop lightweight signatures for IoT malware. This system integrates malware detection, clustering, and signature generation to accurately and efficiently identify IoT malware. In order to reduce malware detection costs, malware clustering has been done to group malware samples from the same family. Then, each malware group's high-level strings and statistical features are extracted to generate signatures to identify IoT malware on IoT devices. This paper proposes a multi-stage clustering mechanism for clustering IoT malware samples into several families using code statistics, high-level structural similarity, and N-gram string features. Then, an efficient signature generation scheme is designed to generate signatures using strings and reliable and extractable statistical features. The string feature is extracted using N-gram text analysis, while the statistical feature includes code-level statistics. These features are carefully constructed and integrated to minimize inter-cluster similarity and maximize intra-cluster similarity.

Alasmary et al. [48] presented an IoT malware detection mechanism using Control Flow Graphs (CFG) in this work. To motivate the detection mechanism, the underlying characteristics of IoT malware are compared to other types of malware - Android malware, which are also Linux-based - on several features. About 6000 malware and benign samples of the Internet of Things were used to test the model of CFGs, and the detection accuracy is 99.66%. In this method, the data set required to achieve the final goal has been collected. This way, a data set of programs has been created and classified into three categories: Android malware samples, Internet of Things malware samples and healthy samples. In summary, this paper presents a detection model for identifying IoT malware by enhancing the features generated from Control Flow Graphs (CFG). In this regard, an in-depth graph-based analysis of three different datasets, i.e., Android malware, IoT malware, and benign IoT samples, is performed to highlight the similarities and differences between Android and IoT malware, and a detection system is provided for new malware.

Darabian et al. [49] use the sequential pattern extraction technique to identify the most common opcode sequences of malicious Internet of Things programs. Detected maximum frequent patterns (MFP) of opcode sequences can be used to distinguish malicious applications from healthy IoT applications. The suitability of MFPs as a classification feature is then evaluated for K-nearest neighbor (KNN), support vector machines (SVM), multilayer perceptron (MLP), AdaBoost, decision tree, and random forest classifiers. In particular, this method has achieved a 99% accuracy rate in detecting invisible IoT malware. Some opcodes have a high frequency of repetition in the malware dataset. Therefore, the number of repetitions of opcodes is counted in DMalware so that opcodes are ranked based on their repetitions. The more an operation code is repeated, the higher its rank and vice versa. Also, a dictionary is created with operational codes as keys and a ranking of operational codes as values. The recursive nature of these algorithms increases the complexity and execution time. Therefore, these algorithms may only be suitable when a few items are sequenced. Therefore, a number of parallel algorithms, such as Hash-based Parallel Algorithm for Sequential Patterns Mining (HPSPM) and Parallel Sequential Pattern Discovery Using Equivalence (pSPADE), have been proposed to generate maximum patterns in a reasonable time.

Takase et al. [50] proposed a malware detection mechanism using values extracted from the processor. The goal is to load the malware detection mechanism into the hardware using processor information and reduce the consumption of hardware resources. In this paper, a prototype of the proposed mechanism is implemented using QEMU, which is a virtual machine. It is shown that the proposed mechanism can classify malware or healthy programs using processor information. The obtained tracking data is returned to the classifier, and the classifier identifies the malware. Therefore, a classifier should be prepared that is trained in advance from trace data of healthy and malware programs. It is taught by labelling Normal or Attack for the obtained trace data. The class clause classifies trace data with a unit of instruction. However, the classification with a single instruction unit has the problem that the classification granularity needs to be considered bigger. Therefore, the concept of window width is introduced in this method, and the classification is done with a window unit. Even if parts of the instructions are classified as an attack, the executed program is not always considered an attack. Therefore, this method aims to improve classification accuracy by adjusting the window width. If the rate in the window is equal to or greater than the threshold, the window is classified as normal.

Nguyen et al. [51] have proposed a lightweight method for identifying IoT botnets based on extracting high-level features from functional graphs called PSI-Graph for each executable file. This feature shows effectiveness when faced with the multi-architecture problem while avoiding the complexity of control flow graph analysis used by most existing methods. The experimental results show that the proposed method reaches an accuracy of 98.7% with a dataset of 11,200 ELF files, which includes 7,199 IoT botnet samples and 4,001 healthy samples. According to the life cycle of the Internet of Things botnet, an Internet of Things botnet is characterized by five stages of its life cycle. This method consists of four steps: function call generation, PSI graph construction, preprocessing, and classification. First, binary programs are extracted with UPX and IDA Pro. Then the function call diagrams and PSI diagrams are made from isolated codes and caller-callee relationships. After that, the PSI graphs are processed and converted into numerical vector data with the graph embedding technique called graph2vec. Finally, a convolutional neural network is used to classify these samples into botnets and non-botnets.

Asadi et al. [52] used the botnet detection method using the combined algorithm of particle swarm optimization (PSO) with the voting system (BD-PSO-V) to improve the challenges of previous studies. This method used the particle swarm optimization algorithm to select prominent and effective features in detecting botnets. Also, the voting system, including a deep neural network algorithm, support vector machine (SVM), and C4.5 decision tree, was used to identify botnets and classify samples. The decision-making strategy of the voting system was based on maximum votes. The most important innovation of this research is the combination of the PSO feature selection algorithm with the voting system using deep learning to identify botnets. Two datasets, ISOT and Bot-IoT, were used to verify the BD-PSO-V system's performance further. The BD-PSO-V simulation improved the accuracy by an average of 0.42% and 0.17% on the ISOT dataset and Bot-IoT dataset, respectively, compared to the other investigated methods. Additionally, the impact of six well-known attacks on both datasets was evaluated. Despite the slight decrease in accuracy, the results of BD-PSO-V showed promising performance against various attacks.

The main approaches for IoT botnet detection are static, dynamic, and hybrid analysis. Static analysis is the process of parsing files without executing them, while dynamic analysis, in contrast, executes them in a controlled and monitored environment to record system's changes for further investigation. Nguyen et al. [53], proposed a novel and advanced method for IoT botnet detection using dynamic analysis to improve graph-based features, which are generated based on static analysis. Specifically, dynamic analysis is used to collect printable string information that appears during the execution of the samples. Then, used the printable string information to traverse the graph, which is obtained based on the static analysis, effectively, and ultimately acquiring graph-based features that can distinguish benign and malicious samples. In order to estimate the efficacy and superiority of the proposed hybrid approach, authores conduct the experiment on a dataset of 8330 executable samples, including 5531 IoT botnet samples and 2799 IoT benign samples. Our approach achieves an accuracy of 98.1% and 91.99% for detecting and classifying

IoT botnet, respectively. These results show that the approach has outperformed other existing contemporary counterpart methods in the aspects of accuracy and complexity. In addition, experiments also demonstrate that hybrid graph-based features for IoT botnet family classification can further improve static or dynamic features' performance individually.

Many researchers have studied the effect of reducing the number of features of datasets on the detection performance of IoT attacks. Selecting several features from a dataset is a data mining technique effectively integrated into botnet detection and identification systems design. Hosseini et al. [54], proposed a new botnet detection system to detect botnets in IoT using the feature selection technique based on the slime mold algorithm (SMA) and slap swarm algorithm (SSA). On the other hand, the number of features and the importance of features selected from the dataset can directly impact the detection error rate. Therefore, this paper presented a practical and efficient multi-objective algorithm for detecting botnets based on feature selection. To maximize the performance of the proposed algorithm, authors have used chaos theory. In addition, they have integrated the mechanism of the Disruption operator with the proposed algorithm. An excellent balance is created in the components of exploration and exploitation. Finally, to check and evaluate the proposed algorithm, the standard datasets available in the UCI source, created based on the real traffic of infected devices on the IoT botnet, have been used. The results obtained from the proposed algorithm indicate a significant and good performance. The results show that it has a high ability to detect botnets in IoT networks and has been able to achieve an excellent low error rate. This paper presents the MOSMASSA algorithm as multi-objective anomaly-based intrusion detection in IoT. Pareto dominance is used in the MOSMASSA design. In the proposed algorithm, search operations can be directed to optimize the problem using leader solutions in each iteration, such as an elected leader. At each step of optimization, the operation of the non-dominated solution found in the repository is stored in memory until later in the leader to improve the solution no longer used.

Alani [55], proposed an efficient packet-based botnet detection system based on explainable machine learning. This approach also focuses on feature selection to produce a data set with only seven features to train a machine learning classifier that achieves very high accuracy. Testing the proposed system demonstrates an accuracy exceeding 99% relying on these seven selected characteristics extracted from the network packets. The proposed model is explained using Shapley additive explanation to provide transparency to the classifier prediction process. In this paper, introduced a method named BotStop, a packet-based botnet detection system that examines incoming and outgoing network traffic in an IoT device to prevent infections from botnets. The proposed system is founded on explainable machine learning (ML) algorithms with features extracted from network packets. Once an attack is detected, the source is blocked. Much of the previous intrusion and botnet detection research was focused on examining network flow instead of operating at the packet level, as in proposed method. The prior approach causes a noticeable delay in detection as the network flow must end or timeout before the attack can be identified. On the other hand, with proposed system, detection is based on a few features extracted from network packets to provide high efficiency in detection capability.

5. Evaluation and compression

In this section, the reviewed methods are evaluated. Table 2 summarizes the methods reviewed.

Table.2.
A summarizes of the methods reviewed

Ref	*Year*	*Method*	*Dataset*
[37]	2015	A machine learning model for identifying bot networks based on the history of botnets and their activities	A testbed network consisting of some bot-infected hosts
[38]	2015	Botnet control by a bot master through a command and control (C&C) channel.	IRC traffic patterns
[39]	2016	P2P botnet detection using conversation-based mechanism and features based on Fourier transform and information entropy.	ISOT dataset
[40]	2017	Using PSO and SVM to distinguish legitimate user and TCP/HTTP bot	Train dataset
[41]	2018	extracting features and creating a decision tree-based model	CTU-13 and ISOT datasets
[42]	2018	A method based on Hadoop's parallel processing method	NCKU and CCU datasets
[43]	2018	A method based on genetic programming	CTU dataset
[44]	2018	A decentralized framework for extracting features and using machine learning techniques	ISCX dataset
[45]	2018	Identify malicious codes through Opcode sequences	IoT real application dataset

Ref	***Year***	***Method***	***Dataset***
[46]	2018	Using Opcode Graph as a CFG for Malware Detection	A dataset of 1078 benign and 128 malware samples for ARM-based IoT applications
[47]	2018	Using the analysis of CFG charts	Real IoT malware dataset with 5150 malware samples
[48]	2019	Creating signatures for IoT malware classification	Internet of Things malware dataset containing 2962 malware samples from CyberIOC
[49]	2020	Malware detection by Opcode frequency analysis	Collecting different malware samples from online sources and applying multi-pattern techniques on them
[50]	2020	Malware detection using values extracted from the processor	Malware analysis to find out their prevalence and distribution in IoT devices
[51]	2020	Malware detection using PSI diagram	IoTPOT dataset
[52]	2020	Combining PSO with voting system to detect botnet attacks	ISOT and Bot-IoT datasets
[53]	2022	A botnet detection method using dynamic analysis to improve graph-based features	ARM and MIPS datasets
[54]	2022	The required information is collected from the network and then placed in the multi-criteria decision space. Then the algorithm is executed in order to identify the botnet in the Internet of Things.	Vowel, Vehicle, Ionosphere, Sonar, Hill-valley, LSVT, CNAE-9, Yale_64
[55]	2022	Efficient packet-based botnet detection using machine learning	IoT Network intrusion dataset

Table 3 deals with the evaluations presented in each of the articles and false positive and true positive rates are shown. Table 4 shows an overview of the reviewed botnet detection methods along with the advantages and disadvantages of each method.

Table.3.
Evaluated performance metrics of the existing botnet detection protocols

Ref	***FP***	***TP***
[37]	2.3	97
[38]	7	90
[39]	3	90
[40]	9	91
[41]	2	98.6
[42]	2	81
[43]	1.08	97.47
[44]	11	97
[45]	3	98
[46]	2.45	98
[47]	4.6	95
[48]	2	99.66
[49]	2	99
[50]	1	100
[51]	3.2	98
[52]	2.1	99
[53]	3.5	98.1
[54]	2	97
[55]	0.31	99

Table.4.
An overview of the reviewed botnet detection methods

Ref	***Advantage***	***Disadvantage***
[37]	Identifying botnets in the first steps	Failure to consider noise in model construction
[38]	Attack detection in the C&C phase	Not dealing with botnets with encrypted messages
[39]	Avoid the influence of noise	Decreasing detection accuracy, distorting periodic behaviour by inserting random intervals between activities
[40]	Suitable for detecting TCP/HTTP botnets	Detection of some botnet models have not been taken into account in the creation of noise models
[41]	Appropriate performance of different classifiers	Failure to detect botnet with high accuracy and good time efficiency
[42]	The ability to identify hidden and hidden bots with a high degree of reliability	Failure to detect P2P botnets in real-time due to the high waiting time for collecting NetFlow reports
[43]	Improving the performance and efficiency of the algorithm due to the use of the genetic programming algorithm, investigating the effect of increasing the amount of data dimensions and reducing the balance of labels on the performance of the algorithm	This paper does not show that the GP algorithm is the best method to detect botnets in streaming data with limited label budget and class imbalance. To solve the label imbalance problem, it only uses the data beauty method and does not provide other methods.
[44]	Using an unsupervised learning approach to identify new and unknown botnets. Providing an adaptive and adjustable	Involving the human user in detecting the bot network, using different data traffic networks, due to the high volume of data and their complexity,

Ref	*Advantage*	*Disadvantage*
	framework to detect new botnets, using a combination of current and historical data to improve the accuracy of botnet detection	implementing this method on old hardware with limited resources is difficult.
[45]	Using deep recurrent neural networks as an effective approach in detecting malicious software, by using this approach, malicious software can be identified with high accuracy and the amount of false detection can be minimized.	It is only suitable for ARM-based samples, if the malicious software uses newer techniques, the accuracy of the system in identifying them will decrease, and using this method may require high processing.
[46]	The use of recurrent neural networks and eigenvalue space learning as two distinct approaches in identifying malicious software, high accuracy and speed, resistance to intrusion techniques such as encryption, compression and renaming.	The multi-architecture problem is not considered, using recurrent neural networks and eigenvalue space learning requires a lot of training data, requires high processing speed and powerful hardware to perform learning and data processing.
[47]	Providing an efficient and fast method for signature generation to identify malicious software, the ability to apply to devices with different structures, due to the high speed of signature generation, the proposed method can be suitable for the Internet of Things with limited resources.	High complexity and time, due to the simplicity of the method, may be less resistant than intrusion techniques, due to the use of an adaptive approach, until an instance of malware is detected, the signature generation method cannot detect it.
[48]	The use of graph similarity algorithms and dimensionality reduction algorithms in the detection of new malware makes this method effective in detection.	Being time-consuming, the number of malware samples used in this article is very small, and the results of this article may not be effective in real environments due to the lack of data.
[49]	The use of cop code technique and machine learning algorithms in the detection of polymorphic Internet of Things malware, which can be highly accurate in detecting malware.	For ARM-based samples only, the small number of polymorphic malware samples used may reduce the performance of this method in detecting more sophisticated and newer malware, this method only focuses on the parts of the code that are related to the malware that may cause creating false results, increasing the time required for diagnosis

Ref	*Advantage*	*Disadvantage*
[50]	The article has practically examined and evaluated the proposed system, using processor information as one of the inputs of the intrusion detection system can improve the detection of malware.	Internet of Things is not strong for botnet detection, the time-consuming detection by this method reduces the efficiency and accuracy of the proposed method against the variety of technologies.
[51]	Detection of IoT botnets using analysis that can identify botnets more effectively and accurately. Using graph networks to detect botnets is particularly suitable for IoT botnets.	Time consuming to convert graphs into vector data The need for data collected from IoT devices, which may be difficult to collect due to security and privacy restrictions, graph-based diagnosis needs to be implemented and tested in real conditions to understand its efficiency and accuracy.
[52]	Effective detection of botnets using PSO algorithm and voting system is especially suitable for detecting complex patterns and dynamic changes of botnets.	The low accuracy of the approach in both datasets, the proposed method needs to determine the exact number of thresholds and its parameters. So that the optimal determination of each parameter can create challenges for implementation and use in real conditions.
[53]	Using combined graph-based features and improving the performance of static or dynamic features individually	The sandbox used does not support many architectures and does not acquire any dynamic threads during monitoring. Many do not display malicious executables.
[54]	Increasing detection accuracy due to the use of multi-objective hybrid optimization algorithm for botnet detection, the proposed method has a significant improvement over the previous methods.	In the article, the details related to the implementation of the multi-objective hybrid optimization algorithm for botnet detection are not fully stated, a specific dataset was used to solve the problem, and it is not possible to generalize the results to other cases.
[55]	Increasing the accuracy and response time of identifying botnets, which is based on the examination of Internet network packets and the use of machine learning.	Using data from an IoT simulation, therefore, the performance of the proposed method in real conditions should be evaluated, not investigating the security problems that may arise using the proposed method.

6. Conclusion

In conclusion, as the Internet of Things (IoT) continues to grow and expand, the threat of botnets targeting these devices also increases. Therefore, it is essential to have effective botnet detection methods to protect IoT devices from malicious attacks. This examination and comparison of botnet detection methods in the IoT have shown that various approaches, such as machine learning, network behaviour analysis, and signature-based detection, can effectively detect and mitigate botnet attacks. However, each method has its advantages and limitations, and selecting the appropriate method depends on various factors, such as the type of IoT device and the network environment. Thus, further research and development are necessary to improve botnet detection methods and ensure the security and privacy of IoT devices and their users.

References

[1] Shah, S. H., & Yaqoob, I. (2016). A survey: Internet of Things (IOT) technologies, applications and challenges. 2016 IEEE Smart Energy Grid Engineering (SEGE), 381-385.

[2] Lee, S. K., Bae, M., & Kim, H. (2017). Future of IoT networks: A survey. Applied Sciences, 7(10), 1072.

[3] Sovacool, B. K., & Del Rio, D. D. F. (2020). Smart home technologies in Europe: A critical review of concepts, benefits, risks and policies. Renewable and sustainable energy reviews, 120, 109663.

[4] Alshehri, F., & Muhammad, G. (2020). A comprehensive survey of the Internet of Things (IoT) and AI-based smart healthcare. IEEE Access, 9, 3660-3678.

[5] Qadri, Y. A., Nauman, A., Zikria, Y. B., Vasilakos, A. V., & Kim, S. W. (2020). The future of healthcare internet of things: a survey of emerging technologies. IEEE Communications Surveys & Tutorials, 22(2), 1121-1167.

[6] Da Xu, L., He, W., & Li, S. (2014). Internet of things in industries: A survey. IEEE Transactions on industrial informatics, 10(4), 2233-2243.

[7] Chand, H. V., & Karthikeyan, J. (2018). Survey on the role of IoT in intelligent transportation system. Indonesian Journal of Electrical Engineering and Computer Science, 11(3), 936-941.

[8] Arasteh, H., Hosseinnezhad, V., Loia, V., Tommasetti, A., Troisi, O., Shafie-khah, M., & Siano, P. (2016, June). Iot-based smart cities: A survey. In 2016 IEEE 16th international conference on environment and electrical engineering (EEEIC) (pp. 1-6). IEEE.

[9] Li, J., Dai, J., Issakhov, A., Almojil, S. F., & Souri, A. (2021). Towards decision support systems for energy management in the smart industry and Internet of Things. Computers & Industrial Engineering, 161, 107671.

[10] Patel, K. K., Patel, S. M., & Scholar, P. (2016). Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. International journal of engineering science and computing, 6(5).

[11] Sha, K., Wei, W., Yang, T. A., Wang, Z., & Shi, W. (2018). On security challenges and open issues in Internet of Things. Future generation computer systems, 83, 326-337.

[12] Peterson, J. M., Leevy, J. L., & Khoshgoftaar, T. M. (2021, August). A review and analysis of the bot-iot dataset. In 2021 IEEE International Conference on Service-Oriented System Engineering (SOSE) (pp. 20-27). IEEE.

[13] Booij, T. M., Chiscop, I., Meeuwissen, E., Moustafa, N., & den Hartog, F. T. (2021). ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets. IEEE Internet of Things Journal, 9(1), 485-496.

[14] Wazzan, M., Algazzawi, D., Bamasaq, O., Albeshri, A., & Cheng, L. (2021). Internet of Things botnet detection approaches: Analysis and recommendations for future research. Applied Sciences, 11(12), 5713.

[15] Gaonkar, S., Dessai, N. F., Costa, J., Borkar, A., Aswale, S., & Shetgaonkar, P. (2020, February). A survey on botnet detection techniques. In 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE) (pp. 1-6). IEEE.

[16] Ali, I., Ahmed, A. I. A., Almogren, A., Raza, M. A., Shah, S. A., Khan, A., & Gani, A. (2020). Systematic literature review on IoT-based botnet attack. IEEE Access, 8, 212220-212232.

[17] Al-Othman, Z., Alkasassbeh, M., & Baddar, S. A. H. (2020). A state-of-the-art review on IoT botnet attack detection. arXiv preprint arXiv:2010.13852.

[18] Mahboubi, A., Camtepe, S., & Ansari, K. (2020). Stochastic modeling of IoT botnet spread: A short survey on mobile malware spread modeling. IEEE Access, 8, 228818-228830.

[19] Hamid, H., Noor, R. M., Omar, S. N., Ahmedy, I., Anjum, S. S., Shah, S. A. A., ... & Tamil, E. M. (2021). IoT-based botnet attacks systematic mapping study of literature. Scientometrics, 126, 2759-2800.

[20] Xing, Y., Shu, H., Zhao, H., Li, D., & Guo, L. (2021). Survey on botnet detection techniques: classification, methods, and evaluation. Mathematical Problems in Engineering, 2021, 1-24.

[21] Hassan, W. H. (2019). Current research on Internet of Things (IoT) security: A survey. Computer networks, 148, 283-294.

[22] Alaba, F. A., Othman, M., Hashem, I. A. T., & Alotaibi, F. (2017). Internet of Things security: A survey. Journal of Network and Computer Applications, 88, 10-28.

[23] Al-Garadi, M. A., Mohamed, A., Al-Ali, A. K., Du, X., Ali, I., & Guizani, M. (2020). A survey of machine and deep learning methods for internet of things (IoT) security. IEEE Communications Surveys & Tutorials, 22(3), 1646-1685.

[24] Liu, J., Xiao, Y., Ghaboosi, K., Deng, H., & Zhang, J. (2009). Botnet: classification, attacks, detection, tracing, and preventive measures. EURASIP journal on wireless communications and networking, 2009, 1-11.

[25] Sánchez, I., Kuusela, E., Turpeinen, S., Röning, J., & Riekki, J. (2009, November). Botnet-inspired architecture for interactive spaces. In Proceedings of the 8th international Conference on Mobile and Ubiquitous Multimedia (pp. 1-10).

[26] Hachem, N., Mustapha, Y. B., Granadillo, G. G., & Debar, H. (2011, May). Botnets: lifecycle and taxonomy. In 2011 Conference on Network and Information Systems Security (pp. 1-8). IEEE.

[27] Seyfollahi, A., & Ghaffari, A. (2020). Reliable data dissemination for the Internet of Things using Harris hawks optimization. Peer-to-Peer Networking and Applications, 13, 1886-1902.

[28] Howard, P. N., Woolley, S., & Calo, R. (2018). Algorithms, bots, and political communication in the US 2016 election: The challenge of automated political communication for election law and administration. Journal of information technology & politics, 15(2), 81-93.

[29] Koroniotis, N., Moustafa, N., & Sitnikova, E. (2019). Forensics and deep learning mechanisms for botnets in internet of things: A survey of challenges and solutions. IEEE Access, 7, 61764-61785.

[30] Liu, C. Y., Peng, C. H., & Lin, I. C. (2014). A survey of botnet architecture and batnet detection techniques. International Journal of Network Security, 16(2), 81-89.

[31] Anagnostopoulos, M., Kambourakis, G., & Gritzalis, S. (2016). New facets of mobile botnet: architecture and evaluation. International Journal of Information Security, 15, 455-473.

[32] Rodríguez-Gómez, R. A., Maciá-Fernández, G., & García-Teodoro, P. (2013). Survey and taxonomy of botnet research through life-cycle. ACM Computing Surveys (CSUR), 45(4), 1-33.

[33] Moustafa, N., Turnbull, B., & Choo, K. K. R. (2018). An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. IEEE Internet of Things Journal, 6(3), 4815-4830.

[34] Vishwakarma, R., & Jain, A. K. (2020). A survey of DDoS attacking techniques and defence mechanisms in the IoT network. Telecommunication systems, 73(1), 3-25.

[35] Alhajri, R., Zagrouba, R., & Al-Haidari, F. (2019). Survey for anomaly detection of IoT botnets using machine learning auto-encoders. Int. J. Appl. Eng. Res, 14(10), 2417-2421.

[36] Zhao, H., Shu, H., & Xing, Y. (2021, January). A review on IoT botnet. In The 2nd International Conference on Computing and Data Science (pp. 1-7).

[37] Yahyazadeh, M., & Abadi, M. (2015). BotGrab: A negative reputation system for botnet detection. Computers & Electrical Engineering, 41, 68-85.

[38] Chen, C. M., & Lin, H. C. (2015). Detecting botnet by anomalous traffic. journal of information security and applications, 21, 42-51.

[39] Narang, P., Hota, C., & Sencar, H. T. (2016). Noise-resistant mechanisms for the detection of stealthy peer-to-peer botnets. Computer Communications, 96, 29-42.

[40] Kapre, A., & Padmavathi, B. (2017, April). Adaptive behaviour pattern based botnet detection using traffic analysis and flow interavals. In 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA) (Vol. 1, pp. 410-414). IEEE.

[41] Mathur, L., Raheja, M., & Ahlawat, P. (2018). Botnet detection via mining of network traffic flow. Procedia computer science, 132, 1668-1677.

[42] Wang, C. Y., Ou, C. L., Zhang, Y. E., Cho, F. M., Chen, P. H., Chang, J. B., & Shieh, C. K. (2018). BotCluster: a session-based P2P botnet clustering system on NetFlow. Computer Networks, 145, 175-189.

[43] Khanchi, S., Vahdat, A., Heywood, M. I., & Zincir-Heywood, A. N. (2018). On botnet detection with genetic programming under streaming data label budgets and class imbalance. Swarm and evolutionary computation, 39, 123-140.

[44] Cid-Fuentes, J. Á., Szabo, C., & Falkner, K. (2018). An adaptive framework for the detection of novel botnets. Computers & Security, 79, 148-161.

[45] HaddadPajouh, H., Dehghantanha, A., Khayami, R., & Choo, K. K. R. (2018). A deep recurrent neural network based approach for internet of things malware threat hunting. Future Generation Computer Systems, 85, 88-96.

[46] Azmoodeh, A., Dehghantanha, A., & Choo, K. K. R. (2018). Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning. IEEE transactions on sustainable computing, 4(1), 88-95.

[47] Alhanahnah, M., Lin, Q., Yan, Q., Zhang, N., & Chen, Z. (2018, May). Efficient signature generation for classifying cross-architecture IoT malware. In 2018 IEEE Conference on Communications and Network Security (CNS) (pp. 1-9). IEEE.

[48] Alasmary, H., Khormali, A., Anwar, A., Park, J., Choi, J., Abusnaina, A., ... & Mohaisen, A. (2019). Analyzing and detecting emerging Internet of Things malware: A graph-based approach. IEEE Internet of Things Journal, 6(5), 8977-8988.

[49] Darabian, H., Dehghantanha, A., Hashemi, S., Homayoun, S., & Choo, K. K. R. (2020). An opcode-based technique for polymorphic Internet of Things malware detection. Concurrency and Computation: Practice and Experience, 32(6), e5173.

[50] Takase, H., Kobayashi, R., Kato, M., & Ohmura, R. (2020). A prototype implementation and evaluation of the malware detection mechanism for IoT devices using the processor information. International Journal of Information Security, 19(1), 71-81.

[51] Nguyen, H. T., Ngo, Q. D., & Le, V. H. (2020). A novel graph-based approach for IoT botnet detection. International Journal of Information Security

[52] Asadi, M., Jamali, M. A. J., Parsa, S., & Majidnezhad, V. (2020). Detecting botnet by using particle swarm optimization algorithm based on voting system. Future Generation Computer Systems, 107, 95-111.

[53] Nguyen, T. N., Ngo, Q. D., Nguyen, H. T., & Nguyen, G. L. (2022). An advanced computing approach for IoT-botnet detection in industrial Internet of Things. IEEE Transactions on Industrial Informatics, 18(11), 8298-8306.

[54] Hosseini, F., Gharehchopogh, F. S., & Masdari, M. (2022). A Botnet Detection in IoT Using a Hybrid Multi-objective Optimization Algorithm. New Generation Computing, 40(3), 809-843.

[55] Alani, M. M. (2022). BotStop: Packet-based efficient and explainable IoT botnet detection using machine learning. Computer Communications, 193, 53-62.