Ain Shams Engineering Journal 12 (2021) 3725–3736

Contents lists available at ScienceDirect

Ain Shams Engineering Journal

journal homepage: www.sciencedirect.com

A survey and taxonomy of program analysis for IoT platforms

Alyaa A. Hamza [a,b,*], Islam T. Abdel-Halim [c], Mohamed A. Sobh [b], Ayman M. Bahaa-Eldin [d]

[a] Computer Engineering & Systems Department, Faculty of Engineering and Technology, Badr University in Cairo, Cairo, Egypt
[b] Computer Engineering & Systems Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt
[c] Misr International University, Cairo, Egypt
[d] Misr International University (on leave from Ain Shams University), Cairo, Egypt

ARTICLE INFO

Article history:
Received 14 December 2020
Revised 18 March 2021
Accepted 31 March 2021
Available online 13 May 2021

Keywords:
IoT Security
Program Analysis
Security Analysis System
Malware Detection

ABSTRACT

Heterogeneity in the Internet of Things (IoT) environment is a critical issue for supporting security and privacy.

IoT environment has become an open invitation to hackers to control and attack connected IoT devices. So, it has become essential to face security issues & challenges in IoT due to IoT applications' rapid development. Program Analysis (PA) is a method that focuses on defending against attacks on the systems implemented to detect malware applications. It is also responsible for adequately analyzing applications' behavior to provide security and privacy. This survey has been introduced carefully based on the systematic literature reviews (SLR) guidelines to provide a survey and taxonomy of the PA with its related topics: the sensitivity of analysis and characteristics of analysis. It presents a new classification of PA techniques. This classification has been created by examining the implemented security analysis systems (SAS) that detect various malware applications. More importantly, this survey presents the three types of SAS that used PA methods for the first time. Also, the related surveys, the performance metrics of PA and IoT Security Issues and Challenges have been discussed. Finally, future directions of the PA have been discussed.

1. Introduction

The IoT is not a new technology in which wireless sensors' network is the main backbone. It is composed of four significant components, including applications, services, sensing, and processing information. There are additional components, such as privacy and security [1]. IoT architecture is defined as a framework for determining hardware networks, functional organization, configuration, operation principles, procedures, and data formats.

IoT architecture Includes a range of physical objects, sensors, cloud services, developers, Drives, communications layers, users, layers of business, and IoT protocols. The IoT value chain contains the essential elements of the IoT, it explains how to create the value, and it also shows how to value users interact with each other to provide value. The value chain concept is considered as physical, digital, and virtual data. It is presented as a part of the IoT different ecosystems and the architectural layers of the IoT.

Focusing on building security and protection requirements for devices leads to increased security and privacy when using IoT technology [2]. It is essential to carefully assess the security risks that may occur to devices that are part of the IoT, using program analysis in the sense of identifying all the threats that must be faced, the assets to be protected, and the required level of security.

The PA concept is not a new term [44]. It is critical to achieving safety and security within the IoT environment where program data sources can be studied and analyzed at several points. It can identify and discover malware programs that attack devices to determine the type of this information, integrability, timeliness, originality, type of source, and trustworthiness [3]. Therefore, the PA became very important due to the difficulty of determining the data source and any malware risk. So, it became necessary to study and develop techniques for analyzing programs, but proportionately with the IoT's nature. Indeed, recent studies have begun

* Corresponding author.
E-mail addresses: alyaashams12@yahoo.com (A. A. Hamza), islamhalim@yahoo.com (I.T. Abdel-Halim), mohamed.sobh@eng.asu.edu.eg (M.A. Sobh), ayman.bahaa@eng.asu.edu.eg (A.M. Bahaa-Eldin).

Peer review under responsibility of Ain Shams University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.asej.2021.03.026
2090-4479/

to focus on essential issues related to security and privacy through how to apply effective PA techniques to deal with any attacks that occur against IoT systems and devices [4].

The number of various domains is increased In IoT platforms such as smart homes, healthcare, industrial, etc. These domains have a massive number of different applications with different Security Analysis systems (SAS). These systems have been implemented based on various program analysis types, techniques & methods. In [61], using the latest data acquisition techniques, 5G network slicing, and data interoperability, a built framework is developed to effectively processing healthcare data. This platform can accurately classify a device of unknown type and extract its data, as well as its network parameters, to merge it with other Internet of Medical Things (IoMT) devices into network slices.

PA techniques play a critical role in the platforms; they identify a malicious app. Unfortunately, A few surveys related to IoT program analysis techniques. There are shortcomings in some surveys [4,55,60] to cover all PA techniques, the sensitivity of analysis, and analysis characteristics that applied to SAS. In this context, this survey presents a comprehensive review of different systems. The main contribution of this survey is to achieve the following objectives:

- Present the role of the program analysis (PA) on the IoT value chain platform. It is one of IoT architecture, an overview of the security issues & challenges in IoT.
- Discuss PA and its related topics are analysis techniques, the sensitivity of analysis, and analysis characteristics.
- Present A new taxonomy of PA techniques.
- List the various types of applications based on Security Analysis Systems (SAS).
- Discuss the current security issues and open research areas for IoT security.

The rest of the survey topics is organized as follows: ***Section II*** gives the background of IoT value chain and security issues in IoT. ***Section III*** introduces the methodology and research questions that have been observed in this survey. ***Section IV*** introduces program analysis techniques for malware detection, and ***section V*** presents the sensitivity of the program analysis techniques. Characteristics of the program analysis techniques introduced in ***section VI***. ***Section VII*** introduces types of applications based on SAS. ***Section VIII*** introduces the algorithm of SAS. ***Section IX*** presents a discussion from different points of related surveys. Finally, a conclusion and future trends in IoT security have been presented in ***Section X***.

2. Background

There is a lot of research in IoT security due to its future importance. This section introduces more details about the PA role in the IoT value chain platform as one IoT architecture, an overview of IoT's security issues & challenges.

2.1. Program analysis role in the IoT value chain platform

IoT value chain is one of the platforms that explain the different layers of IoT and the connection between these layers, as shown in Fig. 1. The physical layer is the first layer in the IoT value chain: It is the layer of things that consists of sensors and actuators. The second layer is the IoT Gateway, responsible for connecting things, obtaining data from them, and then transferring them to the network and vice versa. The third layer is Data Acceptance which is responsible for filtering data before it is processed. Concurrent with this stage, there is a critical alarm clock as the time element determines. For example, when the data was obtained from any source. It gets the same data from different sources and delayed access to data; this leads to the suspicion that an error or attack has occurred.

Next comes the fourth layer: automation, which, in turn, relates to the collection of data as part of some processes. The cloud infrastructure (platform) is the most critical IoT value chain phase, including processing, analytics, and storage. Because of scalability, it is more appropriate for IoT than classical backends. The PA plays a critical role here as it checks the data received from various sources to make sure that this data is free from malware.

2.2. Overview of the security issues & challenges in IoT.

There are viewpoints on IoT environments' security specifications, such as the Middleware / IoT gateway, IoT systems, IoT users, communication channels, and cloud applications. IoT security issues and challenges are still at an initial research stage [62] due to the limited processing capabilities and the IoT environments' heterogeneous nature. Traditional security countermeasures cannot be applied directly to IoT applications. There are also various specifications and communication frameworks involved, and the high number of interconnected devices increases scalability challenges [5]. Many efforts have been made to examine security issues about IoT programming frameworks, cloud-based and hub-based systems [6,7]. In [64], Integrate the Industrial Internet of Things (IIoT) platform and blockchain into industrial processes in real-time to identify security issues like Data Security, Complicated processes, Data connectivity, and Hardware compatibility issues.

IoT devices have limited resources (memory, time, and computation), so they need an integrated malware analyzer to detect and prevent malware attacks. The problem is that most of the existing SAS have very high computing costs, and there have been several studies on systems that suggest solutions to this issue within the IoT framework. However, they still need to be improved either in terms of accuracy or efficiency [8]. IoT security requirements, such as transparency, availability, confidentiality, authentication, and access control, make IoT devices unique and challenging, particularly for developers, to build advanced IoT systems that are resistant to IoT-based attacks [9].

Usually, IoT SAS are designed from a variety of resource-constrained tools. It is also difficult to integrate security defense methods, such as vulnerability scanning and malware signature scanning. And while some devices may have adequate resources, implementing these methods on other devices may be challenging because other IoT devices are limited and difficult to upgrade. The use of security firewalls for IoT devices is inefficient and unfeasible. Public key infrastructures are also not appropriate for IoT environments [5].

3. Methodology & research questions

It is essential to clarify all the steps taken to submit the survey. These steps have carefully followed the general guidance of systematic literature review (SLR) proposed by kitchenham [10]. SLR composed of three phases which are planning, conducting, and reporting. Firstly, a set of research questions has been formulated to achieve the survey objective to classify PA techniques based on SAS that are already used in IoT applications in the different platforms.

RQ1: What are PA and their related topics?
RQ2: What is the classification of PA techniques for IoT?
RQ3: How to create SAS?

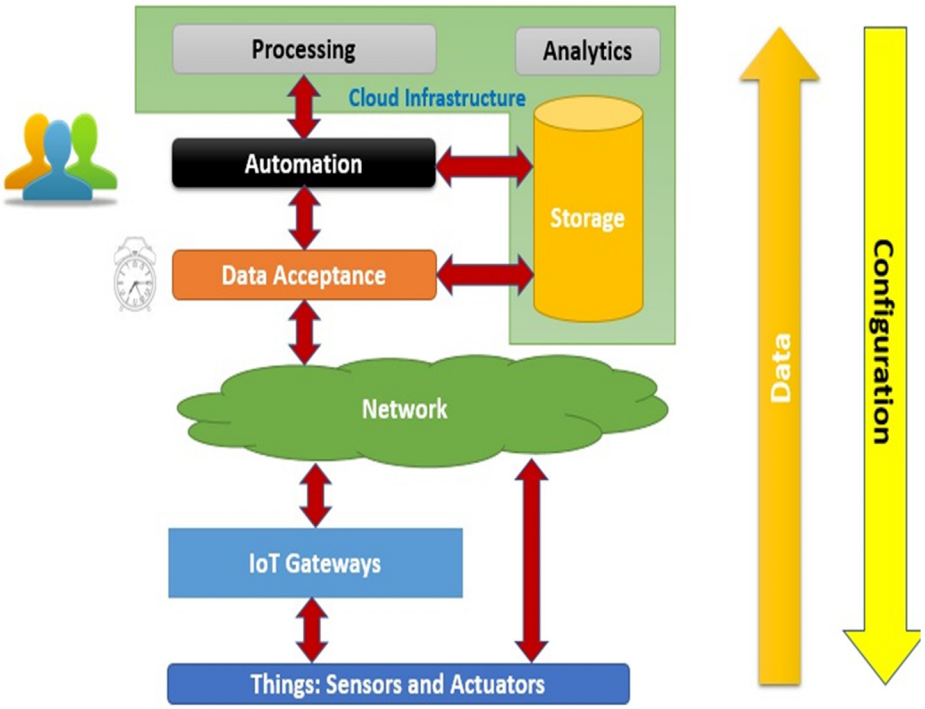

Fig. 1. IOT Value Chain Platform.

RQ4: What is the performance metrics that evaluate program analysis in various Platforms?

RQ5: What is the current security issues and challenges in the program analysis and open research areas?

Through answer to these questions was obtained through the following steps: firstly, is to define the search method that includes the criteria based on the search engines that used in this survey which is (Elsevier, Springer, IEEE, Science Direct & Other publishers) as shown in Fig. 2. Secondly, it is to define the initial selection of the keywords concerning this survey shown in table 1. Finally, it is determining the selection criteria for candidate papers divided into two categories (Inclusion & Exclusion).

The inclusion category contains all selected papers in this survey. These papers meet the selection criteria proposed starting from the keywords in the research domain, number of paper citations with the year of publication, written in English, and published in a peer-reviewed journal or conference. The exclusion category contains all excluded papers for several reasons. I) papers not related to the scope of this survey. II) papers not published in a peer-reviewed journal or conference. III) papers not focused on analysis techniques. IV) papers with shortcomings (such as the clarity of research objective & results).

Table 1
Search Keywords.

Domain of Research	Keywords
Program Analysis (PA)	Malware Analysis, Static Analysis, Dynamic Analysis, Hybrid Analysis, Analysis Techniques, Analysis Sensitivity
Security Issues	Security Challenges, Vulnerability, Malware Detection, Malware Classification
Platform & Applications	IoT Platform, IoT Domains, IoT Applications, Android
Systems	Security Systems, Security Tools, Security Solutions

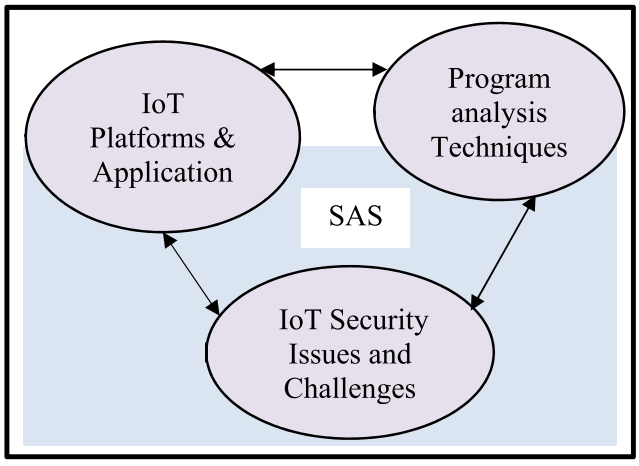

Fig. 2. The Objective of this Survey.

Many SAS has been introduced to analyze IoT malware using several techniques. After the category of selection is applied. This survey focuses on the paper published from 2014 till now, as shown in Fig. 3.

4. Program analysis (PA) techniques

There are various techniques used in PA, and it is the fundamental impact of making an efficient security system. There is not a clear classification of these techniques till now. In this section, a new classification of the PA techniques used for security systems to detect either IoT malware in various applications or malware in IoT applications. As shown in Fig. 1, the IoT value chain platform can be sent or received through the network and devices. Therefore, the process of PA is a crucial phase, and it can be applied at every stage of data collection to ensure that there is no harmful program affecting its behavior. For PA to keep up with the various malware, electronic defense mechanisms (security systems), espe-

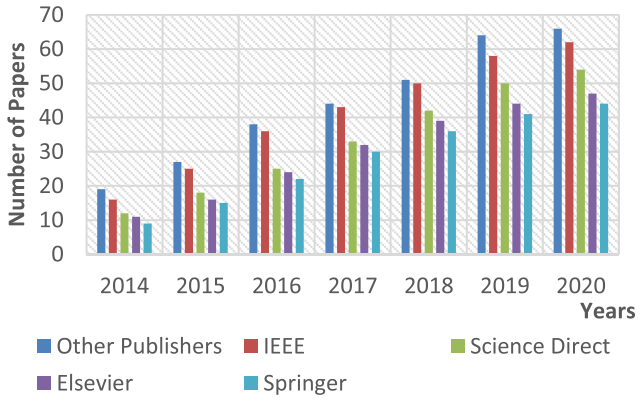

Fig. 3. The numbers of publications of SAS from 2014 to 2020.

cially protecting network terminals, had to be improved like URL filtering, firewalls, email protection, anti-spam, and sandboxing [11].

PA has some problems related to detecting viruses, detecting unpacking execution, matching malware with the stored samples, and detecting trigger-based behavior. These problems detect hidden viruses, an automatic dump for malware detection, do not rule out the general virus detection problem, semantic malware detection, identify trigger-based behavior automatically, and NP-complete problems. These problems must take int to account when design a security system [12]. From Studying the various research about security systems, PA has related issues to be done perfectly: analysis types (static, dynamic, or hybrid), analysis techniques, the sensitivity of analysis, and analysis characteristics.

After reviewing all relevant PA studies, the PA techniques are classified into four categories in this survey as shown in Fig. 4. **A.** Visualization Based Analysis Technique (VBAT) is the technique-based on describing the features of malware in the form of visual signs or images. **B.** Features Based Analysis Technique (FBAT) is the technique-based on analyzing the features of the source code, or file to detect malware. **C**. Traditional Based Analysis Technique (TBAT) is the technique-based on various efficient traditional techniques to detect malware. **D.** Supplementary Based Analysis Technique (SBAT) is the technique-based on machine learning and data mining techniques to detect malware.

4.1. Visualization based analysis technique (VBAT)

VBAT is a technique that is used to visualize any malicious software. There are two steps in the malware visualization technique: Converting the generic executable into a standard byte plot picture (executable binaries are transformed into 2-D & 3-D images). then, extracting information from those images using image processing techniques [13]. The use of visualization and highly immersive visual analytics tools will directly facilitate the time-consuming cycle of malware analysis in the investigation, comparison, and summary of malware samples. By using this technology, it is easy to discover any harmful program. Indeed, this technology has been used in many systems of security analysis [14]. This technology includes many mechanisms that depend on malware detection through images. As follows, some of these mechanisms are utilized.

- Honeypots: A honeypots technique is a type of security installations that has been created deliberately to investigate the attack and penetration. It is used to protect production system detection and unauthorized access and distortion. It also helps to study attackers' behavior and unknown attacks behavior via collect information about them [15]. Many security systems used honeypots' technique. IoTPOT is such a system used to honey pot attacks that focus on IoT devices [16].

- Visualizing Graphs: This technique is based on converting any text form (codes) to visual form (set of nodes and edges) by using visual techniques [17]. There are ways to analyze malware by using visualizing graphs techniques such as force-directed layout, a group in a box, radial, arc, semantic substrate, Planar Graphs, Tree Layout, Space-Filling, and matrix diagram. Many security systems use visualization graphs.
- Standard 2D/3D Displays: It is a technique to represent documents as several factors on either a 2-D or 3-D plane as the space between each pair of factors indicates how similar the two files [18].

- Geometrically-Transformed Displays: This technique utilizes transformation concepts of its attribute through reorganization and projection to visualize data [19].
- Iconic Displays: This technique can describe any multidimensional data as icons (its properties correspond to data attribute values). It allows visual analysis speedy by comparing many other icons to recognize similarities and differences, such as Chernoff faces [20].
- Pixel Displays: This technique is like the icons technique. Both methods contain small groups in the spatial arrangement to represent these groups' multidimensional values, and the organization of larger arrangements may represent a dimensional spatial/temporal data set.
- Stacked Displays: This technique is based on representations for hierarchical data, including such hierarchical stacking, treemaps, and neighborhood treemaps (Nmaps) [21] and hierarchical structures for multidimensional data including such dimensional stacking [22].

4.2. Features based analysis technique (FBAT)

FBAT relies on extracting features from programs to detect any malware. Features analysis is used by dynamic, static, and hybrid analysis. These features contain several hexadecimal sequences of bytes. There are several studies about how to extract features from the program. For example, the heldroid application detects any ransom Android program, uses NLP-based file encryption activities such as text classifiers, lock detection, and tracking, and uses features extracted from software applications Harmful massage alarm, call function, etc. [23].

- Anomaly-Based: It is one of the detection security systems that often deal with a huge of data. So, utilizing feature selection is an important method to remove redundant data complexity, which is usually applied for anomaly detection. This malware detection method involves establishing a standard process or software profile and searching for any deflection from that profile. In [24] provides an anomaly-based technology where dynamic analysis is used to detect any injected code created and obfuscated.
- Image-Based: This technique focuses on processing images by extracting features (binary data or malware behavior logs). In [25], Present a new approach to developing a malware classification by applying a deep network to images converted from binary samples. Create a novel hybrid transformation method for transforming binaries into color images that transmit binary semantics.

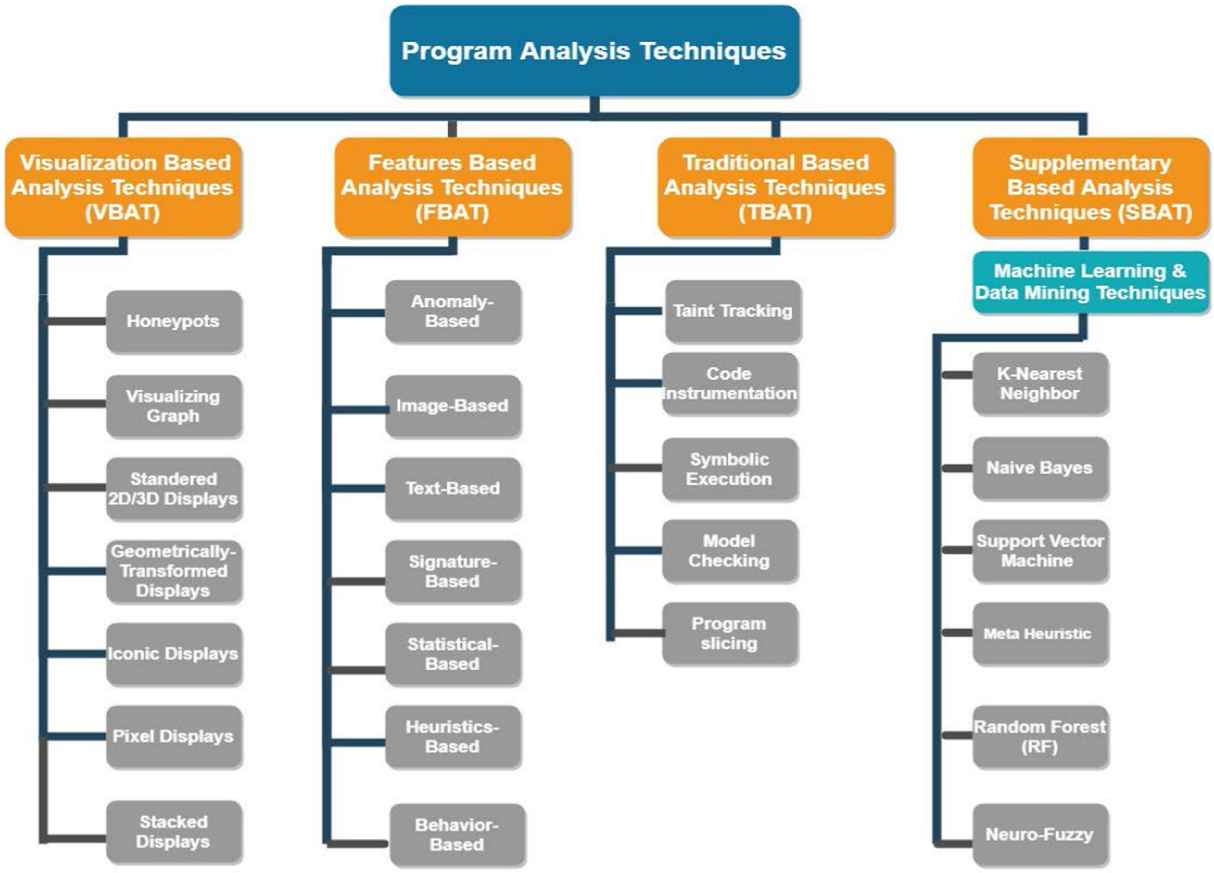

Fig. 4. Classification of Program Analysis Techniques.

- Text-Based: This technique relies on extracting features from source code as text to detect any malware. Cybersecurity is a study that applied text mining and analysis, such as malware detection and classification [26].
- Signature-Based: This technique extracts a specific signature from the recorded malware file and utilizes this signature to detect similar malware. A signature is a file hash or a series of bytes applied to recognize a specific malware [27].
- Statistical-Based: This technique relies on the statistical concepts for detecting any malware obtained from program features such as Hidden Markov Models (HMMs) used to classify metamorphic malware [28].
- Heuristic-Based: This approach can detect vulnerabilities by monitoring the movements of the software file. In [27] Present a detection system that analyzes the malware during the training run (learning) level. If the malware were detected, the file would be labeled malware or, during the testing (monitoring) process, based on a pattern extracted during the training test.
- Behavior-Based: Malware is discovered using behavioral technology to address the shortcomings that result from using the signature-based method. This method distinguishes others from finding new malware and quickly detecting any malicious programs done by polymorphic technologies [29].

4.3. Traditional based analysis technique (TBAT)

TBAT includes five different malware detection mechanisms. These techniques are always used to analyze programs.

- Taint Tracking: It utilizes this technology to follow Taint as a general software security method as it has a high and accurate ability to recognize malware. This technology begins detecting sensitive information obtained from a taint source with a label showing the type of information. After that, this technology starts tracking the source of Taint from a stealth source and publishes that defect to all references to be careful and then delete it when removing any traces from the tainted data. The tainted data could be a variety loaded with a constant code, such as [30].
- Code Instrumentation: This technique distinct from others as it adds special code to the program's origin code to get the program run-time behavior. This additional code during the instrumentation phase is often named instrumented code. This code executes as part of the program's normal behavior. Still, it is different from the original code that gets data necessary for some analysis such as context description, attack discovery and attack rehabilitation [31].
- Symbolic Execution: This technology is a popular method that simultaneously analyses the property facing multiple execution paths. Symbolic execution can concurrently manage many paths that can be provided under different data [32].
- Model Checking: This technique examines the programs' behavior is on a true path or not. The model checking begins with representing applications as finite state machines. The application's execution is examined the listed defined specifications expressed in temporal logic formulas like a Computational Tree Logic and Linear Temporal Logic by a general model checker [33].
- Program Slicing: It is collected on the program's special sections that have a more side impact on some states to determine the program's slices. It is helpful in the debugging phase as it can capture some program basics and control to restrict trusted data from interacting with infected data. [34].

4.4. Supplementary based analysis technique (SBAT)

SBAT includes machine Learning & data mining techniques. This technique is used to detect malware to other techniques VBAT, FBAT, and TBAT, as it is used to detect the same features and delete the redundant features. Some techniques have appeared mainly with the SAS.

- K-Nearest Neighbor: Recent research has proposed a new K-nearest neighboring (K-NN) algorithm to insure versus one of the Label manipulation attacks [35]. K-NN also can be used as a classifier like in [36] supports a novel classification approach by using the combination of requirements that the user requires from each application using the Basic Component Analysis and the closest neighborhood methods probability methods.
- Naïve Bays: It is one of the machine learning algorithms that considered a classifier of probability. It is based on Bayesian theory with strong assumptions of independence. [37].
- Support Vector Machine (SVM): It is a supervised-based (machine learning algorithms) used for classification a supervised-based (machine learning algorithms) used for classification purposes. It works as a non-probabilistic binary linear classifier or a non-linear classifier using implicit mapping [38].

- Meta Heuristic: It has various malware detection algorithms such as clonal selection, genetic algorithm, harmony search, and Negative selection algorithms. The *meta*-heuristic algorithms are ideal for counteracting any attacks as they can easily detect the harmful object to control its signature [39].
- Forest random(RF): It is a supervised machine learning algorithm utilized for malware detection. This classifier consists of tree-structured. Therefore, it gives a more trust result all the time without parameter tuning [40], in [41] Utilized random forest to classify malware after transforming a malware binary into an image.
- Neuro-Fuzzy: This technique is a hybrid of fuzzy systems and artificial intelligence. It has been applied in more security systems for malware analysis. In [42], an Adaptive Neuro-Fuzzy System for Malware Detection (ANFSMD) was introduced to detect any malicious code. ANFSMD uses operation codes and Application Programming Interface calls to examine Portable Executable files' performance.

5. The sensitivity of the program analysis

The sensitivity of analysis is one of the essential factors that significantly impact the program's analysis than analysis techniques. Sensitivity analysis describes that the data can be split into uncertainty in any system's output by different methods to the several sources of uncertainty in its input [63]. It relies on recognizing any difference in the execution of the program perfectly. The analysis's sensitivity differs for the various algorithms used by the static analysis methodology, resulting in a trade-off between precision and scalability of the analysis. This parameter, therefore, categorizes static methods focused on their sensitivity to the following properties [43]. From security systems studies, the sensitivity of the program analysis can be divided into six elements.

- Flow Sensitivity: Considering the executive order's flow sensitivity in the program's analysis, it measures the responsive analysis of the flow of variables that alter their content during the program's implementation. On the other hand, in a flow-insensitive analysis, the variable requires one valid abstracting of the values that the variable receives during the program's entire execution [43,44].
- Context Sensitivity: It analyzes cover several procedures, considering the target function block within the code's context that calls it. Specifically, when calling-site contexts are used, only execution paths possible by matching calls and returns are identified during analysis [43,45].
- Path Sensitivity: In program analysis, the predicate needs to be inserted into the conditional branch's sensitivity path. Analyzes consider the execution path and clearly distinguish the knowledge derived from the various paths [44].
- Field Sensitivity Field-insensitive analysis views all fields of an object as equal [46].
- Object Sensitivity: Sensitivity analysis of objects is a known analysis of context-points. This sensitivity analysis is parametrized by limiting the names of symbolic things associated with each location selected.
- Provenance Sensitivity: It focuses on the data sources which need to know whether the data was written by the programmer or hacker like encryption data [4].

6. Characteristics of program analysis

Characteristics of PA have a high effect on malware detection. It helps the compiler represent the source code better and easily verifiable, so this section classified these characteristics into six various groups from different perspectives.

- Control Flow Graph (CFG): It is based on a directed graph that describes the program statements by its nodes and the flow of control between the statements by the edges of the graph [43]. CFG is used in many security software analyses and has been studied for several years [47,48].
- Call Graph (CG): It is a directed graph in which each node perfectly represents a methodology, and the edge indicates the call (or return) of the method [43].
- Inter-Procedural Call Flow Graph(ICFG): ICFG is a perfect combination of CFG and CG that links separate CFG using call and return edges [43].
- N-Gram: It is a sub-strings of a string greater length. For example, the string can be split into "MALWARE" to several of three grams: "MAL" and "ALW" and "ARE" and so on [49]. The features of n-gram correspond to the different mixtures of these n bytes. Each feature reflects how many times a certain mixture of n bytes occurs in a binary. PA prefers the size of the n-grams used in sequences of not more than 3. The number of features to be regarded increases exponentially with n [31,50].
- API Calls: API call sequences are one of the most enticing important ways to represent the actions of a piece of code like malware. APIs and systems call to make all possible to evaluate samples' behavior, but at a higher level. They can be extracted either statically or dynamically by evaluating the disassembly code (to get a list of all calls that can theoretically be executed) or by running tracks (for a list of calls invoked) [51].
- Opcode: It is an abbreviation for the Operational Code, a subset of the machine language instruction that specifies the operation to be performed. Specifically, the program is specified as a series of ordered assembly instructions [52]. Opcodes define machine-level operations performed by Portable Executables (PEs) and can be retrieved by static analysis by testing the assembly code [53,54].

7. Types of applications based on security analysis system (SAS)

SAS is essential for protecting against any attack that might occur. Especially, the malware that appears due to the connections between IoT objects. In the latest studies, this malware is called

"IoT malware" .various SAS is used to detect IoT malware. SAS is used in various applications that are more infected with IoT malware. In this section, SAS has three types according to the applications: A) IoT applications. B) Android applications. C) other applications).

7.1. IoT applications based on SAS

There are many IoT applications such as smart cities, health care, connected cars, smart retail, etc. IoT malware exploits gaps in various objects, such as applications, Network interfaces, firmware, components, software, and sensors. It isn't easy to apply a few SAS to all applications to detect malware due to heterogeneity in the IoT environment. There is a strong connection between applications and their platforms—also, the connection between IoT devices and the considered platform. In [65], IoT applications using the cloud are more affected by new malware. This malware usually needs a secure SAS, which prevents attackers access to the network through data to send and receive from cloud platforms and IoT devices, as discussed. Therefore, many SAS is used to work efficiently on their application. A few of them are working on devices.

The author in [67] develops SAS used in the application like google fit to detect the permission by extracting the requested Permission scopes and the already data used from them. After that, the google fit application will be able to determine the necessary requests from dummy requests. SOTERIA in [66] Creates the model state from the source code of the IoT application to check that the application or multi-device system complies with confidentiality, protection, and functionality. Also, [68] present a SAS called IOTGUARD, which works in three different stages: implementation of a code tool that adds extra logic to an app's source code to collect application data during runtime, storage application data in a dynamic model that reflects the runtime operation actions of applications, and finally, identifying IoT safety and security rules, and applying related policy goals on the dynamic model of individual or sets of applications.

7.2. Android applications based on SAS

Malware from the IoT has infiltrated several million mobile devices worldwide over the past few years. Mobile malware detection methods contrasted based on many assessment requirements and metrics but primarily focused on Android OS.

There are some limitations in android devices that cause increasing malware infection through a connection via IoT, such as Anti-malware systems run as standard software without any additional rights. As a result, other device memories and private data are not scanned during the mobile device's scanning. Android malware detection faces two main critical challenges: 1) how to create an effective malware model; 2) how to reduce false alarms and distinguish between malicious and malicious software from them. Different types of Android malware differ in attack targets, attack methods, and applications [69].

There are several SAS used in android devices, some of which are described in Table 3. Each SAS developed through different methods of analysis, techniques, and analytical characteristics. The author in [70] generates a SASA called Amandroid Transforms an application's Dalvik byte code to an intermediate representation (IR) amiable to static analysis. It creates an intermediate interface design that simulates interactions between the Android device and the user to restrict the scalability analysis range. Also, and Amandroid constructs an inter-component data flow graph (IDFG) of the entire application.

SWORD is a dynamic SAS that Captures the runtime performance of the system-calls software. Constructed a sequential call graph (SSG) structure using the Markov chain. It shows the actions of the program after collecting many standard paths [71]. CANDYMAN is a hybrid SAS that used machine learning techniques to categorize Android malware groups by integrating dynamic analysis with Markov chains.

A dynamic analysis method allows the extraction of descriptive data from a malware sample in a series of states. In contrast, the Markov chain enables the simulation of transition probabilities between sequence states, used as properties in the classification process.

7.3. Other applications based on SAS

The third category of applications that used SAS to detect IoT malware is a collection of various applications such as other operating systems (IOS, Linux, . . .), compilers, websites, memory, and so on. These applications need SAS to be protected from malware. AMA [72] is a static SAS designed to detect malware in websites; it is a framework able to detect malware via the web page's static code analysis.

Towards this purpose, the system conducts a possible plaintext attack using strings that are likely to be found in malicious web pages. Another SAS for the webpage is Jsdc [73], which is a hybrid SAS. It is used text-based techniques with applying one of machine learning as a supplementary technique. It used the API method calls as the analysis characteristics and used the object method to define the analysis's sensitivity. It is used to identify and classify JavaScript malware effectively and efficiently, clarifying the attack model and discovering new malware variants and vulnerabilities.

It begins with machine learning techniques for detecting JavaScript malware using textual data's predictive properties, program structures.

The TaintPipe [74] is a hybrid SAS designed for detecting malware in the operating system. It utilizes the Taint technique, the control flow graph in the characteristics of analysis, and it applies the path method as the sensitivity of the analysis. It executes software runtime tracking to generate compact control flow profiles and executes multiple threads as different phases of a pipeline to perform a symbolic taint analysis in parallel. It has some drawbacks because it detects malware after the attack occurred due to its design depending on the pipeline principle. Also, TaintPipe can perform a symbolic taint analysis when the taint states are not found. Andro-dumpsys [75] is a hybrid SAS that relies on detecting malware from volatile memory; it extracts byte code by dynamic analysis and uses static analysis to detect malware actions.

8. Security Analysis Systems (SAS) Algorithm

This section focuses on the six phases to construct and design SAS after reviewing various types of SAS. The steps are shown in Fig. 5 have a high impact on SAS actions' success and efficiency.

The first phase defines which applications the SAS will be applied to and defines the platform domain that interacts with this application. Also, the malware type should be predicted, which is related to the application domain. The second phase is to decide the analysis type (it plays a successful role in the SAS success). Although static analysis has more advantages, often, the application form does not consider other types of analysis. The third phase is to select a suitable technique from the VBAT, FBAT &TBAT. It depends on SAS's target (ex. Excusable file, codes, byte . . .) and the analysis method. It is recommended in the fourth phase to use one of machine learning or data mining techniques as a support technique with the core technique of analysis to remove any redundant features and sometimes to classify malware easily into

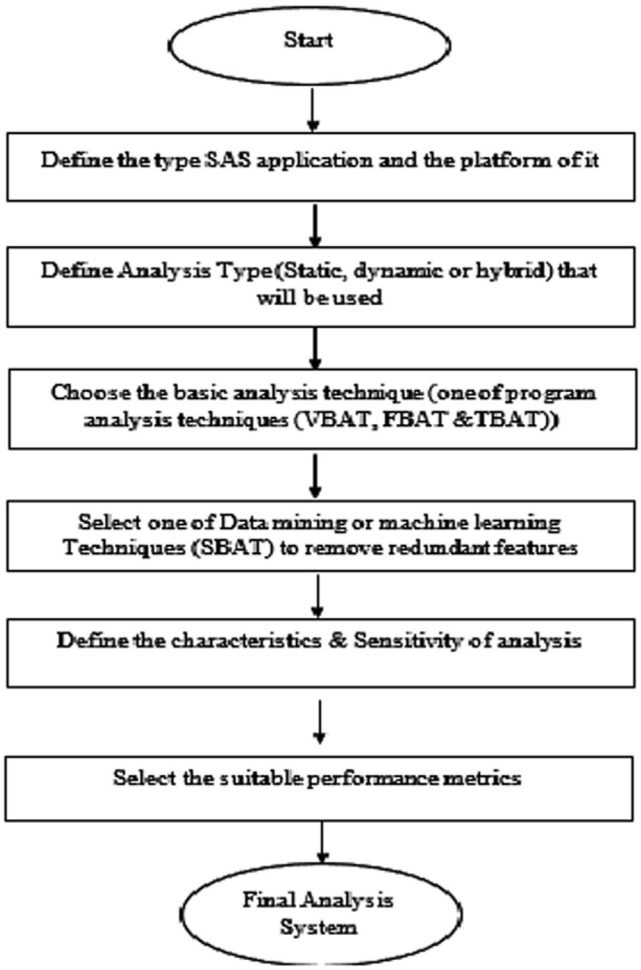

Fig. 5. SAS Algorithm.

classes. The fifth phase defines the characteristics and sensitivity of analysis due to the urgent roles to change SAS's behaviour. The characteristics and the sensitivity method that will be utilized play an essential role in the SAS actions. Finally, it is essential to put the performance metrics suitable with the developed SAS to show the performance and efficiency of it is fulfilled.

9. Discussion

This section examines three significant points in the PA: A) the overview of similar surveys covering PA and its techniques from various perspectives. B) The different performance metrics that have already been used to estimate SAS. C) IoT security issues and challenges that have several different perspectives.

9.1. Related program analysis (PA) surveys and its techniques

There is a lot of research in program analysis to ensure the security of applications, whether such applications are connected to a standard network, including several common threats or applications on the IoT that have an excellent opportunity to expose IoT applications to malware continuously. More dangerous is that the malware from the IoT is improving all the time.

The author in [4] discusses security and privacy issues in IoT that motivate PA techniques based on static and dynamic analysis types. TBAT and the sensitivity of the analysis are discussed. It also explores various platforms such as Amazon's AWS IoT, Samsung's SmartThings, Open-HAB, Apple's HomeKit, and Android Things platforms. The main contributions of [11] are: Offers a comprehensive overview of the PA techniques and characteristics of the analysis based on SBAT for malware detection and classification and discusses the problems and drawbacks of machine learning.

In [55] Provide a taxonomy for android systems and categorize methods of analysis according to some questions: (1) what the main task of SAS is, (2) what the types of features are extracted from Portable Executable Files (PEs), and (3) what machine learning algorithms they are using.

While the survey discusses the features and characteristics of analysis, it does not outline new research trends, especially deep learning and multimodal approaches.

A hybrid analysis is used in most SAS in all different applications as it performs best, unlike using one of the two approaches separately. Some applied SAS does not have any characteristics or sensitivity, so this SAS has more limitations than other SAS that defines them. There are no performance metrics used in most applied SAS, making SAS have more detecting malware restrictions.

As shown in Table 2, this survey covers all categories possible of it into four categories of techniques (A. VBAT. B. FBAT C. TBAT D. SBAT. It covered all analysis types (static, dynamic, and hybrid). The characteristics and the sensitivity of SAS are covered.

After studying various SAS, static analysis is the low rate in usage as it has more disadvantages for detecting malware, as shown in Fig. 6.Program analysis techniques that are mainly used are VBAT &SBAT, as shown in Fig. 7. In Fig. 8, the most application types that applied SAS are the android applications as the IoT applications not commonly used such as the android. but, IoT applications will be the highest usage in the future.

9.2. Performance metrics that evaluate SAS

It is essential to evaluate the SAS that helps developers to know the weak points in their SAS. Various mechanisms have examined these weak points. Many of the tools used in the test process. Such as, data sets, case studies, run time results, mathematical evaluation, model complexity assessment, effectiveness in extracting policies, the significance of safe logic patching, permission request, and so on. But, there is a famous test used in more SAS, it is called "Confusion Matrix." The Confusion Matrix represents the predicted outputs of the classification model (mostly SAS based on SBAT). It is obtained by summarizing the total number of correctly and incorrectly categorized predictions based on each class [76]. There are four main elements (True Positive, True Negative, False Positive & False Negative) to be achieved:

a. True Positive (TP): This reflects the total number of positive cases precisely identified by the SAS.
b. True Negative (TN): This reflects the total number of negative cases just recognized by the SAS.
c. False Positive (FP): This reflects the total number of negative cases mislabelled as positive cases.
d. False Negative (FN): This reflects the total number of positive cases mislabelled as negative cases.

9.3. IoT security issues and challenges

There are other viewpoints on IoT environments' security specifications, such as the Middleware / IoT gateway, IoT systems, IoT users, communication channels, and cloud applications. IoT Security issues and challenges are still at an initial research stage. Due to the limited processing capabilities and the heterogeneous nature of the IoT environment. Traditional security countermeasures cannot be applied directly to IoT applications.

Table 2
A Summary of related surveys on program analysis.

Ref.	Analysis Types			Analysis Techniques Categories(Ctg.)				Characteristics of Analysis	Performance Metrics	Sensitivity of Analysis	Security Systems		
	Static	Dynamic	Hybrid	Ctg.1	Ctg.2	Ctg.3	Ctg.4				Sys.1	Sys.2	Sys.3
[4]	✓	✓				✓				✓	✓		
[11]	✓	✓	✓				✓	✓	✓				
[55]	✓	✓					✓	✓				✓	
[56]					✓			✓					
[27]	✓	✓	✓		✓								
[57]	✓						✓	✓					
[43]	✓	✓	✓				✓	✓		✓		✓	
[14]	✓	✓		✓									
[40]					✓		✓		✓				
[58]	✓	✓	✓		✓								
[59]	✓	✓	✓		✓				✓			✓	
[60]	✓	✓	✓		✓		✓						✓
This Survey	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 3
SAS used in android applications.

Ref.	Systems	Types of program analysis	analysis Techniques	Characteristics of Analysis	Sensitivity of analysis	Limitations
[70]	Amandroid	Static	Graph Behavior	Control Flow Graph (CFG)	path	1- Amandroid's ability to manage exceptions is limited.2-Amandroid does not currently handle reflections and concurrency. 3- Android is a component-based system and makes extensive use of inter-component communication (ICC).4-The Android runtime consists of a large base of library code that an app depends upon.
[80]	FlowDroid	Static	Taint analysis	Control Flow Graph (CFG)	flow, context, field & object	1-FLOWDROID resolves reflective calls only if their arguments are string constants, which is not always the case.2-Fully incorporating sound support for multi-threading is a big challenge.
[71]	SWORD	Dynamic	Statistical-Based	System calls	—	The detection accuracy of our work remains intact till 30% in case of rare system-call injection. After that, it gradually decreases.
[81]	DroidCIA	Static	Text-Based	Control Flow Graph (CFG)	—	1-Input String Problem.2-Dead Code Problem.3-Other Miscellaneous Problems.
[82]	OpSeq	Static	Machine Learning	1- Opcode-sequence similarity 2-Semantic-based detection	—	1-OpSeq only extracts features from the available classes.2-OpSeq is designed to process only the Dalvik instructions and as such cannot handle native code.
[83]	EspyDroid +	hybrid	Reflection Guided Static Slicing (RGSS)	—	—	1-EspyDroid + cannot handle hybrid apps that include native code in the app itself not from the Android OS, and JavaScript code.2-EspyDroid + will be inefficient in situations where the app contains behavior dependent on specific data inputs which are not removed in RGSS and the desired inputs are not provided by dynamic analysis.

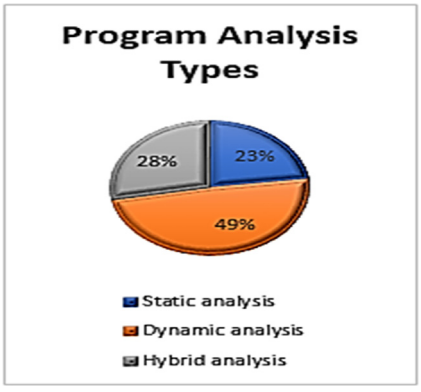

Fig. 6. Program Analysis Types.

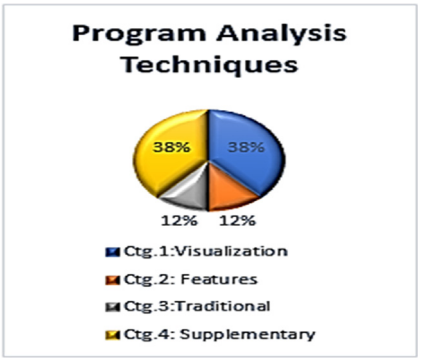

Fig. 7. Program Analysis Techniques.

There are also various specifications and communication frameworks involved, and the high number of interconnected devices increases scalability challenges [5]. So, many efforts have been made to examine IoT programming frameworks' security issues in cloud-based and hub-based systems [6]. The topology of the IoT multi-hop and multi-path domain is entirely different from that of the traditional Networks where service providers path and control traffic, preventing cyberattacks and protecting not just specific computers but also the entire network. Such issues tend to be challenging for IoT environments where there is no central network control. Unavailability of IP for several innovative artifacts

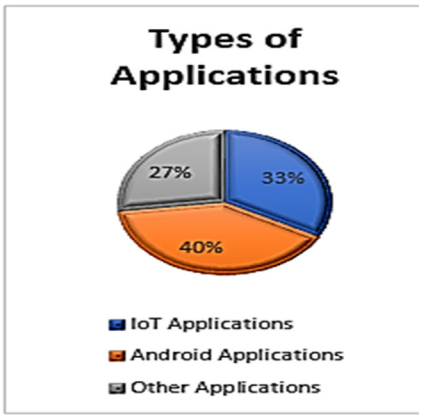

Fig. 8. Types of Application.

raises security issues for routing protocols where spoofing, forwarding, black holes, and sinkhole attacks are started [77].

IoT devices have limited resources (memory, time, and computation), so they need an integrated malware analyzer to detect and prevent malware attacks. The problem is that most of the existing SAS have very high computing costs, and there have been several studies on systems that suggest solutions to this issue within the IoT framework. However, they still need to be improved either in terms of accuracy or efficiency [8]. IoT security requirements, such as transparency, availability, confidentiality, authentication, and access control, make IoT devices unique and challenging, particularly for developers, to build advanced IoT systems that are resistant to IoT-based attacks [78]. Usually, IoT systems are designed from a variety of resource-constrained tools. Therefore, it is hard to integrate can security defense methods, such as vulnerability scanning and malware signature scanning. And while some devices may have adequate resources, the implementation of these methods on other devices may be challenging as these reasons Other IoT devices are limited and difficult to upgrade, the use of security firewalls for IoT devices is inefficient and unfeasible. Public vital infrastructures are also not appropriate for IoT environments [5]. Current and upcoming approaches to IoT security threats were examined, including blockchain, edge computing, fog computing, and machine learning [79]. The utilizing of IoT devices has risen to consideration of security issues and related challenges. IoT apps are vulnerable due to recent increasing attacks, such as the Carna and Mirai botnets. Also, IoT devices generate massive amounts, speeds, and a variety of data. It makes existing solutions less effective and demands new solutions (SAS).

9.4. Conclusion and future directions

Program Analysis (PA) is one of the essential security factors which has more than analysis techniques. These techniques have the successful task of building a perfect SAS that can detect malware. There is a struggle remains between security analysts and malware developers. As it is a battle that does not end quickly, and malware is always complex as fast as discovery grows. Analysis techniques utilize IoT app source code to accomplish various goals, such as recognizing applications' security. This survey presents a new taxonomy survey of the program analysis techniques and their related topics. It explains how to build SAS based on various program analysis techniques. Also. It covers SAS types, which play an essential role in identifying the suitable program analysis techniques to be used. In the future IoT security plan, SAS will be expanded its usage based on a set of different techniques, including a variety of specific tools for developers to test their solutions and analyze complicated interactions between users and IoT system applications for various IoT platforms.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Suo, H., Wan, J., Zou, C., & Liu, J. (2012, March). Security in the internet of things: a review. In 2012 international conference on computer science and electronics engineering (Vol. 3, pp. 648-651). IEEE. https://doi.org/10.1109/iccsee.2012.373
[2] M., T., & Al-Muhtadi, J. (2016). Internet of Things Security based on Devices Architecture. International Journal of Computer Applications, 133(15), 19–23. doi:10.5120/ijca2016908191
[3] Sauerwein C, Pekaric I, Felderer M, Breu R. An analysis and classification of public information security data sources used in research and practice. Computers & Security 2019;82:140–55. doi: https://doi.org/10.1016/j.cose.2018.12.011.
[4] Celik ZB, Fernandes E, Pauley E, Tan G, McDaniel P. Program Analysis of Commodity IoT Applications for Security and Privacy. ACM Comput Surv 2019;52(4):1–30. doi: https://doi.org/10.1145/3333501.
[5] Razouk W, Sgandurra D, Sakurai K. A new security middleware architecture based on fog computing and cloud to support IoT constrained devices. In: Proceedings of the 1st International Conference on Internet of Things and Machine Learning. doi: https://doi.org/10.1145/3109761.3158413.
[6] Fernandes, E., Paupore, J., Rahmati, A., Simionato, D., Conti, M., & Prakash, A. (2016). Flowfence: Practical data protection for emerging iot application frameworks. In 25th {USENIX} Security Symposium ({USENIX} Security 16) (pp. 531-548).
[7] Jia YJ, Chen QA, Wang S, Rahmati A, Fernandes E, Mao ZM, et al. ContexIoT: Towards Providing Contextual Integrity to Appified IoT Platforms. In: Proceedings 2017 Network and Distributed System Security Symposium. doi: https://doi.org/10.14722/ndss.2017.23051.
[8] Soliman SW, Sobh MA, Bahaa-Eldin AM. Taxonomy of malware analysis in the IoT. In: 2017 12th International Conference on Computer Engineering and Systems (ICCES). doi: https://doi.org/10.1109/icces.2017.8275362.
[9] Atlam HF, Wills GB. IoT Security, Privacy, Safety and Ethics. Digital Twin Technologies and Smart Cities 2019;123–149. doi: https://doi.org/10.1007/978-3-030-18732-3_8.
[10] Kitchenham, B. A. (n.d.). Systematic reviews. 10th International Symposium on Software Metrics, 2004. Proceedings. doi:10.1109/metric.2004.1357885
[11] Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. Journal of Network and Computer Applications 2020;153:. doi: https://doi.org/10.1016/j.jnca.2019.102526102526.
[12] Selcuk AA, Orhan F, Batur B. Undecidable problems in malware analysis. In: 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST). doi: https://doi.org/10.23919/icitst.2017.8356458.
[13] Sibi Chakkaravarthy S, Sangeetha D, Vaidehi V. A Survey on malware analysis and mitigation techniques. Computer Science Review 2019;32:1–23. doi: https://doi.org/10.1016/j.cosrev.2019.01.002.
[14] Wagner M, Fischer F, Luh R, Haberson A, Rind A, Keim DA, et al. A survey of visualization systems for malware analysis. In: In Eurographics Conference on Visualization (EuroVis). p. 105–25.
[15] Fan W, Du Z, Fernandez D, Villagra VA. Enabling an Anatomic View to Investigate Honeypot Systems: A Survey. IEEE Syst J 2018;12(4):3906–19. doi: https://doi.org/10.1109/jsyst.2017.2762161.
[16] Pa YMP, Suzuki S, Yoshioka K, Matsumoto T, Kasama T, Rossow C. IoTPOT: A Novel Honeypot for Revealing Current IoT Threats. Journal of Information Processing 2016;24(3):522–33. doi: https://doi.org/10.2197/ipsjjip.24.522.
[17] Tarawaneh, R. A. M., Keller, P., & Ebert, A. (2012). A general introduction to graph visualization techniques. In Visualization of Large and Unstructured Data Sets: Applications in Geospatial Planning, Modeling and Engineering-Proceedings of IRTG 1131 Workshop 2011. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
[18] Diakopoulos N, Elgesem D, Salway A, Zhang A, Hofland K. March). Compare clouds: Visualizing text corpora to compare media frames. In: In Proceedings of IUI Workshop on Visual Text Analytics. p. 193–202.
[19] Grégio, A. R. A., & Santos, R. D. C. (2011). Visualization techniques for malware behavior analysis. Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense X. doi:10.1117/12.883441
[20] Chernoff H. The Use of Faces to Represent Points in k-Dimensional space Graphically. J Am Stat Assoc 1973;68(342):361–8. doi: https://doi.org/10.1080/01621459.1973.10482434.
[21] Duarte FSLG, Sikansi F, Fatore FM, Fadel SG, Paulovich FV. Nmap: A Novel Neighborhood Preservation Space-filling Algorithm. IEEE Trans Visual Comput Graphics 2014;20(12):2063–71. doi: https://doi.org/10.1109/tvcg.2014.2346276.

[22] Inselberg, A., & Dimsdale, B. (n.d.). Parallel coordinates: a tool for visualizing multidimensional geometry. Proceedings of the First IEEE Conference on Visualization: Visualization '90. doi:10.1109/visual.1990.146402
[23] Andronio N, Zanero S, Maggi F. HelDroid: Dissecting and Detecting Mobile Ransomware. Lect Notes Comput Sci 2015;382–404. doi: https://doi.org/10.1007/978-3-319-26362-5_18.
[24] Rabek JC, Khazan RI, Lewandowski SM, Cunningham RK. Detection of injected, dynamically generated, and obfuscated malicious code. In: Proceedings of the 2003 ACM Workshop on Rapid Malcode. p. - WORM'03.. doi: https://doi.org/10.1145/948187.948201.
[25] Vu D-L, Nguyen T-K, Nguyen TV, Nguyen TN, Massacci F, Phung PH. A Convolutional Transformation Network for Malware Classification. In: 2019 6th NAFOSTED Conference on Information and Computer Science (NICS). doi: https://doi.org/10.1109/nics48868.2019.9023876.
[26] Suh-Lee C, Jo Ju-Yeon, Kim Yoohwan. Text mining for security threat detection discovering hidden information in unstructured log messages. In: 2016 IEEE Conference on Communications and Network Security (CNS). doi: https://doi.org/10.1109/cns.2016.7860492.
[27] Damodaran A, Troia FD, Visaggio CA, Austin TH, Stamp M. A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques 2015;13(1):1–12. doi: https://doi.org/10.1007/s11416-015-0261-z.
[28] Wong W, Stamp M. Hunting for metamorphic engines. J Comput Virol 2006;2(3):211–29. doi: https://doi.org/10.1007/s11416-006-0028-7.
[29] Jacob G, Debar H, Filiol E. Behavioral detection of malware: from a survey towards an established taxonomy. J Comput Virol 2008;4(3):251–66. doi: https://doi.org/10.1007/s11416-008-0086-0.
[30] Yuan J, Qiang W, Jin H, Zou D. CloudTaint: an elastic taint tracking framework for malware detection in the cloud. The Journal of Supercomputing 2014;70(3):1433–50. doi: https://doi.org/10.1007/s11227-014-1235-5.
[31] Ahmadi M, Ulyanov D, Semenov S, Trofimov M, Giacinto G. Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification. In: Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy. doi: https://doi.org/10.1145/2857705.2857713.
[32] Baldoni R, Coppa E, D'Elia DC, Demetrescu C. Assisting Malware Analysis with Symbolic Execution: A Case Study. Cyber Security Cryptography and Machine Learning 2017;171–188. doi: https://doi.org/10.1007/978-3-319-60080-2_12.
[33] Jhala R, Majumdar R. Software model checking. ACM Comput Surv 2009;41(4):1–54. doi: https://doi.org/10.1145/1592434.1592438.
[34] Weiser M. Program Slicing. IEEE Trans Software Eng 1984;SE-10(4):352–7. doi: https://doi.org/10.1109/tse.1984.5010248.
[35] Taheri R, Javidan R, Shojafar M, Pooranian Z, Miri A, Conti M. On defending against label flipping attacks on malware detection systems. Neural Comput Appl 2020;32(18):14781–800. doi: https://doi.org/10.1007/s00521-020-04831-9.
[36] Kang S, Yoon JW. Probabilistic K-nearest neighbor classifier for detection of malware in android mobile. Journal of the Korea Institute of Information Security and Cryptology 2015;25(4):817–27. doi: https://doi.org/10.13089/jkiisc.2015.25.4.817.
[37] Pham BT, Prakash I, Khosravi K, Chapi K, Trinh PT, Ngo TQ, et al. A comparison of Support Vector Machines and Bayesian algorithms for landslide susceptibility modelling. Geocarto International 2018;34(13):1385–407. doi: https://doi.org/10.1080/10106049.2018.1489422.
[38] Mantoo BA, Khurana SS. Static, Dynamic and Intrinsic Features Based Android Malware Detection Using Machine Learning. Proceedings of ICRIC 2019;2019:31–45. doi: https://doi.org/10.1007/978-3-030-29407-6_4.
[39] Sun M, Li X, Lui JCS, Ma RTB, Liang Z. Monet: A User-Oriented Behavior-Based Malware Variants Detection System for Android. IEEE Trans Inf Forensics Secur 2017;12(5):1103–12. doi: https://doi.org/10.1109/tifs.2016.2646641.
[40] Karanja EM, Masupe S, Jeffrey MG. Analysis of internet of things malware using image texture features and machine learning techniques. Internet of Things 2020;9:. doi: https://doi.org/10.1016/j.iot.2019.100153100153.
[41] Çayır A, Ünal U, Dağ H. Random CapsNet forest model for imbalanced malware type classification task. Computers & Security 2021;102:. doi: https://doi.org/10.1016/j.cose.2020.102133102133.
[42] Sodiya AS, Falana OJ, Onashoga SA, Badmus BS. Adaptive neuro-fuzzy system for malware detection. Journal of Computer Science and Its Application 2014;21(2):20–31.
[43] Sadeghi A, Bagheri H, Garcia J, Malek S. A Taxonomy and Qualitative Comparison of Program Analysis Techniques for Security Assessment of Android Software. IEEE Trans Software Eng 2017;43(6):492–530. doi: https://doi.org/10.1109/tse.2016.2615307.
[44] Nielson F, Nielson HR, Hankin C. Principles of Program. Analysis. 1999. doi: https://doi.org/10.1007/978-3-662-03811-6.
[45] Reps, T., Horwitz, S., & Sagiv, M. (1995). Precise interprocedural dataflow analysis via graph reachability. Proceedings of the 22nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages - POPL '95. doi:10.1145/199448.199462
[46] Sridharan, M., Chandra, S., Dolby, J., Fink, S. J., & Yahav, E. (2013). Alias Analysis for Object-Oriented Programs. Aliasing in Object-Oriented Programming. Types, Analysis and Verification, 196–232. doi:10.1007/978-3-642-36946-9_8
[47] McCabe TJ. A Complexity Measure. IEEE Trans Software Eng 1976;SE-2(4):308–20. doi: https://doi.org/10.1109/tse.1976.233837.
[48] Gustafsson, J. (2006). The Worst Case Execution Time Tool Challenge 2006. Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (isola 2006). doi:10.1109/isola.2006.72
[49] Abou-Assaleh, T., Cercone, N., Keselj, V., & Sweidan, R. (2004). N-gram-based detection of new malicious code. Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004. doi:10.1109/cmpsac.2004.1342667
[50] Sexton J, Storlie C, Anderson B. Subroutine based detection of APT malware. Journal of Computer Virology and Hacking Techniques 2015;12(4):225–33. doi: https://doi.org/10.1007/s11416-015-0258-7.
[51] Islam R, Tian R, Batten LM, Versteeg S. Classification of malware based on integrated static and dynamic features. Journal of Network and Computer Applications 2013;36(2):646–56. doi: https://doi.org/10.1016/j.jnca.2012.10.004.
[52] Gutmann P. The commercial malware industry. In DEFCON conference, 2007.
[53] Ye Y, Li T, Chen Y, Jiang Q. Automatic malware categorization using cluster ensemble. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD. p. '10.. doi: https://doi.org/10.1145/1835804.1835820.
[54] Anderson B, Storlie C, Lane T. Improving malware classification. In: Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence -. doi: https://doi.org/10.1145/2381896.2381900.
[55] Ucci D, Aniello L, Baldoni R. Survey of machine learning techniques for malware analysis. Computers & Security 2019;81:123–47. doi: https://doi.org/10.1016/j.cose.2018.11.001.
[56] Bazrafshan Z, Hashemi H, Fard SMH, Hamzeh A. A survey on heuristic malware detection techniques. In: The 5th Conference on Information and Knowledge Technology. doi: https://doi.org/10.1109/ikt.2013.6620049.
[57] Shalaginov A, Banin S, Dehghantanha A, Franke K. Machine Learning Aided Static Malware Analysis: A Survey and Tutorial. Cyber Threat Intelligence 2018;7–45. doi: https://doi.org/10.1007/978-3-319-73951-9_2.
[58] Mathur K, Hiranwal S. A survey on techniques in detection and analyzing malware executables. International Journal of Advanced Research in Computer Science and Software Engineering 2013;3(4).
[59] Abawajy J, Huda S, Sharmeen S, Hassan MM, Almogren A. Identifying cyber threats to mobile-IoT applications in edge computing paradigm. Future Generation Computer Systems 2018;89:525–38. doi: https://doi.org/10.1016/j.future.2018.06.053.
[60] Souri A, Hosseini R. A state-of-the-art survey of malware detection approaches using data mining techniques. Human-Centric Computing and Information Sciences 2018;8(1). doi: https://doi.org/10.1186/s13673-018-0125-x.
[61] Mavrogiorgou A, Kiourtis A, Touloupou M, Kapassa E, Kyriazis D. Internet of Medical Things (IoMT): Acquiring and Transforming Data into HL7 FHIR through 5G Network Slicing. Emerging Science Journal 2019;3(2):64. doi: https://doi.org/10.28991/esj-2019-01170.
[62] Aldowah H, Ul Rehman S, Umar I. Security in Internet of Things: Issues, Challenges and Solutions. Recent Trends in Data Science and Soft Computing 2018;396–405. doi: https://doi.org/10.1007/978-3-319-99007-1_38.
[63] Saltelli A. Sensitivity Analysis for Importance Assessment. Risk Anal 2002;22(3):579–90. doi: https://doi.org/10.1111/0272-4332.00040.
[64] Iqbal A, Amir M, Kumar V, Alam A, Umair M. Integration of Next Generation IIoT with Blockchain for the Development of Smart Industries. Emerging Science Journal 2020;4:1–17. doi: https://doi.org/10.28991/esj-2020-sp1-01.
[65] Garg H, Dave M. Securing IoT Devices and SecurelyConnecting the Dots Using REST API and Middleware. In: 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU). doi: https://doi.org/10.1109/iot-siu.2019.8777334.
[66] Celik, Z. B., McDaniel, P., & Tan, G. (2018). Soteria: Automated iot safety and security analysis. In 2018 {USENIX} Annual Technical Conference ({USENIX} {ATC} 18) (pp. 147-158).
[67] Nobakht M, Sui Y, Seneviratne A, Hu W. PGFit: Static permission analysis of health and fitness apps in IoT programming frameworks. Journal of Network and Computer Applications 2020;152:. doi: https://doi.org/10.1016/j.jnca.2019.102509102509.
[68] Celik ZB, Tan G, McDaniel P. IoTGuard: Dynamic Enforcement of Security and Safety Policy in Commodity IoT. In: Proceedings 2019 Network and Distributed System Security Symposium. doi: https://doi.org/10.14722/ndss.2019.23326.
[69] Vignau, B., Khoury, R., & Halle, S. (2019). 10 Years of IoT Malware: A Feature-Based Taxonomy. 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C). doi:10.1109/qrs-c.2019.00088
[70] Wei, F., Roy, S., Ou, X., & Robby. (2018). Amandroid. ACM Transactions on Privacy and Security, 21(3), 1–32. doi:10.1145/3183575
[71] Bhandari S, Panihar R, Naval S, Laxmi V, Zemmari A, Gaur MS. SWORD: Semantic aWare andrOid malwaRe Detector. Journal of Information Security and Applications 2018;42:46–56. doi: https://doi.org/10.1016/j.jisa.2018.07.003.
[72] Seshagiri P, Vazhayil A, Sriram P. AMA: Static Code Analysis of Web Page for the Detection of Malicious Scripts. Procedia Comput Sci 2016;93:768–73. doi: https://doi.org/10.1016/j.procs.2016.07.291.
[73] Wang J, Xue Y, Liu Y, Tan TH. JSDC. In: Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security. doi: https://doi.org/10.1145/2714576.2714620.
[74] Ming, J., Wu, D., Xiao, G., Wang, J., & Liu, P. (2015). TaintPipe: pipelined symbolic taint analysis. In 24th {USENIX} Security Symposium ({USENIX} Security 15) (pp. 65-80).
[75] Jang J, Kang H, Woo J, Mohaisen A, Kim HK. Andro-Dumpsys: Anti-malware system based on the similarity of malware creator and malware centric information. Computers & Security 2016;58:125–38. doi: https://doi.org/10.1016/j.cose.2015.12.005.

[76] Ting KM. Confusion Matrix. Encyclopedia of Machine Learning and Data Mining 2016;1–1. doi: https://doi.org/10.1007/978-1-4899-7502-7_50-1.
[77] Petroulakis NE, Tragos EZ, Fragkiadakis AG, Spanoudakis G. A lightweight framework for secure life-logging in smart environments. Information Security Technical Report 2013;17(3):58–70. doi: https://doi.org/10.1016/j.istr.2012.10.005.
[78] Khattak HA, Shah MA, Khan S, Ali I, Imran M. Perception layer security in Internet of Things. Future Generation Computer Systems 2019;100:144–64. doi: https://doi.org/10.1016/j.future.2019.04.038.
[79] Hassija V, Chamola V, Saxena V, Jain D, Goyal P, Sikdar B. A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures. IEEE Access 2019;7:82721–43. doi: https://doi.org/10.1109/access.2019.2924045.
[80] Arzt S, Rasthofer S, Fritz C, Bodden E, Bartel A, Klein J, et al. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. Acm Sigplan Notices 2014;49(6). 259–269.https://doi.org/10.1145/2666356.2594299.
[81] Chen, Y. L., Lee, H. M., Jeng, A. B., & Wei, T. E. (2015, August). DroidCIA: A novel detection method of code injection attacks on HTML5-based mobile apps. In 2015 IEEE Trustcom/BigDataSE/ISPA (Vol. 1, pp. 1014-1021). IEEE. https://doi.org/10.1109/trustcom.2015.477
[82] Ali-Gombe A, Ahmed I, Richard III GG, Roussev V. December). Opseq: Android malware fingerprinting. In: In Proceedings of the 5th Program Protection and Reverse Engineering Workshop. p. 1–12. doi: https://doi.org/10.1145/2843859.2843860.
[83] Gajrani J, Agarwal U, Laxmi V, Bezawada B, Gaur MS, Tripathi M, et al. EspyDroid+: Precise reflection analysis of android apps. Computers & Security 2020;90:101688. doi: https://doi.org/10.1016/j.cose.2019.101688.

Alyaa A. Hamza is a PH.D. researcher in Computer Engineering, Faculty of Engineering, Ain Shams university, Cairo. she received her B.Sc. and M.Sc. from Computers and Systems Engineering, Faculty of Engineering, Mansoura University in 2011 and 2017 respectively. Her research interests are (wireless Networks, network security, Programming Language and machine learning).

Islam T. AbdelHalim, Received his B.Sc. and M.Sc., PH. D., degrees in Electrical and Computer Engineering from Computer Engineering & Systems Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt in 2004 ,2011 and 2019 respectively. His current research interests include Mobile Ad Hoc Communication Systems and Wireless Sensor Networks with emphasis on Mobility, Routing and Performance Evaluation in Vehicular Communication Networks. He speaks Arabic, and English.

Mohamed A. Sobh is an assistant Professor of Computer and System Engineering, Ain Shams University. He received his B.Sc., M.Sc. and PhD. in Computer Engineering from Ain Shams University in 1996, 2001, and 2007 respectively. His fields of research are Simulation and Modeling - Computer Programming - Computer Security.

Ayman M. Bahaa-Eldin is a professor, Computer and Systems Eng. Dept. Ain Shams University. Prof.Bahaa-Eldin received his B.Sc., M.Sc. and PhD. in Computer Engineering from Ain Shams University in 1995, 1999, and 2004 respectively. He was a visiting professor in several local and international universities. He was appointed as the managing director of the Egyptian Universities Network (EUN), The Egyptian National Research and Education Network (NREN) from 2010 until 2014. His fields of research are Cryptography, Computer Networks, and Computer and Network Security. He published more than 60 papers in refereed international journals and conferences, managed and participated in several national and international research projects, and participated in the reviewing process of many international journals and conferences