

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/359684597>

Forensic Analysis on Internet of Things (IoT) Device Using Machine-to-Machine (M2M) Framework

Article in Electronics · April 2022

DOI: 10.3390/electronics11071126

CITATIONS

24

READS

576

8 authors, including:



Muhammad Shoaib Mazhar

University of Engineering and Technology, Lahore

3 PUBLICATIONS 24 CITATIONS

[SEE PROFILE](#)



Dr. Yasir Saleem

University of Engineering and Technology, Lahore

72 PUBLICATIONS 658 CITATIONS

[SEE PROFILE](#)



Ahmad S. Almogren

King Saud University

212 PUBLICATIONS 6,632 CITATIONS

[SEE PROFILE](#)



Jehangir Arshad

COMSATS University Islamabad

67 PUBLICATIONS 601 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:









Game Based Learning [View project](#)



ENVIRONMENTAL AND ECOSYSTEM POLLUTION [View project](#)

Article

Forensic Analysis on Internet of Things (IoT) Device Using Machine-to-Machine (M2M) Framework

Muhammad Shoaib Mazhar ¹, Yasir Saleem ¹, Ahmad Almogren ², Jehangir Arshad ^{3,*},
Mujtaba Hussain Jaffery ³, Ateeq Ur Rehman ⁴, Muhammad Shafiq ^{5,*} and Habib Hamam ^{6,7,8,9}

- ¹ Department of Computer Engineering, KICS, University of Engineering and Technology (UET) Lahore, Lahore 54000, Pakistan; shoaib.mazhar@kics.edu.pk (M.S.M.); yasir@uet.edu.pk (Y.S.)
- ² Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia; aalmogren@ksu.edu.sa
- ³ Department of Electrical & Computer Engineering, Lahore Campus, COMSATS University Islamabad, Lahore 54000, Pakistan; m.jaffery@cuilahore.edu.pk
- ⁴ Department of Electrical Engineering, Government College University, Lahore 54000, Pakistan; ateeq.rehman@gcu.edu.pk
- ⁵ Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Korea
- ⁶ Faculty of Engineering, Université de Moncton, Moncton, NB E1A 3E9, Canada; habib.hamam@umoncton.ca
- ⁷ International Institute of Technology and Management, Libreville BP1989, Gabon
- ⁸ Spectrum of Knowledge Production & Skills Development, Sfax 3027, Tunisia
- ⁹ Department of Electrical and Electronic Engineering Science, School of Electrical Engineering, University of Johannesburg, Johannesburg 2006, South Africa
- * Correspondence: jehangirarshad@cuilahore.edu.pk (J.A.); shafiq@ynu.ac.kr (M.S.); Tel.: +92-321-4141912 (J.A.)

Abstract: The versatility of IoT devices increases the probability of continuous attacks on them. The low processing power and low memory of IoT devices have made it difficult for security analysts to keep records of various attacks performed on these devices during forensic analysis. The forensic analysis estimates how much damage has been done to the devices due to various attacks. In this paper, we have proposed an intelligent forensic analysis mechanism that automatically detects the attack performed on IoT devices using a machine-to-machine (M2M) framework. Further, the M2M framework has been developed using different forensic analysis tools and machine learning to detect the type of attacks. Additionally, the problem of an evidence acquisition (attack on IoT devices) has been resolved by introducing a third-party logging server. Forensic analysis is also performed on logs using forensic server (security onion) to determine the effect and nature of the attacks. The proposed framework incorporates different machine learning (ML) algorithms for the automatic detection of attacks. The performance of these models is measured in terms of accuracy, precision, recall, and F1 score. The results indicate that the decision tree algorithm shows the optimum performance as compared to the other algorithms. Moreover, comprehensive performance analysis and results presented validate the proposed model.

Keywords: cyber security; machine learning; internet of things (IoT); forensic analysis; machine-to-machine (M2M); attack detection



Citation: Mazhar, M.S.; Saleem, Y.; Almogren, A.; Arshad, J.; Jaffery, M.H.; Rehman, A.U.; Shafiq, M.; Hamam, H. Forensic Analysis on Internet of Things (IoT) Device Using Machine-to-Machine (M2M) Framework. *Electronics* **2022**, *11*, 1126. <https://doi.org/10.3390/electronics11071126>

Academic Editor: Paolo Visconti

Received: 6 February 2022

Accepted: 23 March 2022

Published: 2 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

IoT is a system of interconnected devices that share resources and data securely when a connection with the Internet is established. In terms of less direct interaction with humans, a wider perspective, and expandable characteristics, the IoT supersedes all other traditional networks [1]. The pervasive utilization of IoT has facilitated the development of numerous inventions, including home automation, wearable technology, smart firefighting, smart metering, advanced manufacturing, and intelligent buildings [2]. IoT has made life easier for humans [3].

Despite the fact that IoT device applications are constantly evolving, IoT device security remains a limitation [4]. The manufacturers of IoT devices are more concerned with

bringing new attractive features/functionalities and simplifying the design to make the devices smarter and more cost-effective than with making them secure [5]. In fact, numerous cyberattacks on IoT devices have occurred in recent years as a result of inadequate security measures [6]. The count of cyberattacks and IoT devices are both gradually increasing [7].

One of the most prevalent attacks on the IoT network is denial of service (DoS). According to Cisco's annual Internet study, DDoS cyberattacks are predicted to rise from 2018 to 2023. Figure 1 compares the number of possible DDoS attacks for each year [8]. It is also observed that DoS attacks on IoT are continuously increasing every year. It is reported by the Palo Alto Networks Unit 42 research team, the traffic of 98% of IoT devices is not encrypted which exposes the confidential data on network traffic and causes multi-level attacks [9] on the network and systems. When these vulnerable IoT devices are connected with the network, it increases the threat space for attackers. According to Kaspersky's research, 1.5 billion attacks on IoT devices have been reported in the first 6 months of 2021 [10].

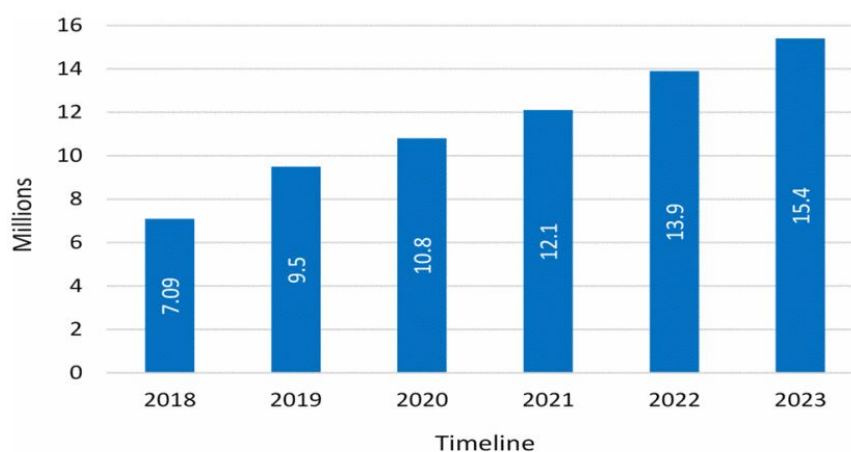


Figure 1. Yearly comparison of DDoS attacks in 2018–2023 [8].

Similarly, smart home IoT devices such as smart TVs, smart cameras, and other IoT devices share 25% of the compromised devices in a botnet attack [11]. By exploiting default credentials, the Mirai attack acquired control of thousands of IoT devices and launched a distributed DoS (DDoS) attack on major systems [12]. HP also reported that 70% of IoT devices are vulnerable [13]. As a result, IoT device vulnerabilities must be protected. IoT devices are not designed to be fully protected against counter vulnerability attacks [14]. A vulnerability is a loophole in a system that provides a fertile ground for cyberattacks. Improper product testing, a haste to market, and a lack of effective legislation are the main causes of IoT vulnerabilities [15]. Because IoT devices have limited memory and processing power [16], it is very hard to identify the shreds of evidence related to attacks. A framework is needed which can detect the attacks on IoT devices and stores evidence of these attacks. By applying the forensic analysis, these vulnerabilities of IoT devices can be mitigated. In addition, trace of attack and attacker can be easily found.

IoT devices have limited memory and processing power, and they can only process a limited number of predetermined instructions sets [17]. As a result, they are unable to capture, monitor, and analyze communication between IoT devices. This nature of IoT devices has made it difficult for security analysts to keep record of various attacks performed on them using forensic analysis. Due to these limitations, evidence acquisition is a major problem for forensic analysis [18]. Special tools and approaches are required to make the IoT environment network more robust and secure. For the study and investigation of IoT devices, more effective forensic techniques must be created and deployed [19].

The above-discussed issues can be easily averted with the help of the forensic analysis technique. We proposed a forensic analysis framework for IoT devices which performs

automated detection of attacks and generates the logs and alerts for these attacks. Forensic analysis refers to the deep investigation of the crime after it happened to explore the reasons behind the crime such as an offender, reasons, and ramifications of a security incident [20]. It is closely related to Security Incident Management (SIM) [21] in which the identification of the security events is performed on the computer network and then reasonable steps are performed to meet the limitations of the security principles that are compromised. Forensics analysis also differs from network auditing [22] because auditing is the pre-analysis of the vulnerabilities in a network and forensic analysis is the post-analysis of the security breaches that document how and when something occurred.

In this framework, data acquisition limitation is resolved by a third-party logging server. IoT devices traffic is redirected to a logging server, where logs and alerts of malicious attack traffic are generated and stored to be used in forensic analysis. These stored logs are regenerated and analyzed in a forensic server to gather information about attacks and attackers. This attack detection is automated through a machine learning approach using a dataset generated from these logs.

Generic forensic analysis process has four steps: data acquisition, examination, analysis, documenting and reporting [23]. In data acquisition [24], data related to a specific attack are collected. Because of the limitation of low processing power of IoT devices, data acquisition was the major problem. No evidence was found in the case of attacks. This problem is resolved by our proposed system by introducing the logging server which used snort for the detection of attacks and stores the logs of malicious traffic and also generate alerts. In the examination, collected data are processed to extract relevant pieces of information. An IP table is used for redirection of the traffic in IoT device configuration. Snort logging server uses logs written for the specific attacks and incorporates them into the detection engine. When traffic is redirected to a server, only the relevant information is extracted, and the rest of the packets are discarded. The extracted data are subjected to analysis in order to obtain more helpful information. Forensic server security onion provides the details of inspection when logs captured by the logging server are regenerated. Security onion provides attackers details, attack type, source port, destination port, and a good deal of other information. Snort also provides the alerts of these attacks. This inspection provides all necessary information to find out the attacker and damage occurred by these attacks. In documenting and reporting, results of the analysis are generated. Future forensic operations should take into account the lessons learned during the forensic procedure [25].

The proposed framework is incorporating both the machine learning model and forensic tools, which will be helpful in IoT forensic investigation. Security onion, which is a forensic server, also generates a new alert for any malicious traffic resides in the network by sniffing mode and helps to write rules for the attack detection. These attack rules can then also be incorporated in our proposed system. Machine learning provides the automated attack detection. After these, multiple reports are generated which describe the details about the type of attack and the number of times an attack was launched and the recommended action that could be taken. This forensic detail will help to draw the complete attack scenario and make the tracing of attackers possible.

The scope of this research is to detect cyberattacks on IoT devices using a forensic analysis of the logs of these devices. For the detection of an attack, we used different forensics tools and machine learning techniques. The major contributions of this research are listed below:

1. Extending existing methods for extracting and examining traces from IoT devices.
2. The major benefit of this system is its capability to efficiently examine and analyze a large amount of data without degrading the performance of IoT devices.
3. The proposed system has the capability to acquire and store the data related to attacks on low powered and low memory IoT devices using a logging server.

4. The proposed framework provides a method to generate an IoT attacks dataset for any IoT enabled organizations, which help to monitor and analyze malicious activities performed.
5. By integration of the proposed system, one will be able to generate the evidence of the cyber crime which can be provided to a court of law by the professionals.
6. It will also help the security analysts to create a significant system to detect IoT attacks efficiently with forensic tools as well as the machine learning.

The rest of the paper is organized as follows. Section 2 contains a literature survey which highlights the forensic analysis techniques, attacks detection methods using machine learning, IoT forensics, and network forensics. Section 3 consists of the methodology of our proposed framework. Section 4 describes results and the discussion with snort-based attack detection of the attacks, security onion-based forensics analysis on acquired logs, and machine models evaluation based on various parameters. The evaluation of machine learning models along with previous work and real-time deployment of the proposed model is also presented. Section 5 elaborates the conclusion and future recommendations of our proposed framework and gives some points to re-imagine our proposed work.

2. Literature Survey

The Internet of Things is divided into several categories. Cloud computing, virtualization, sensor technologies, and A.I technologies are examples of some technologies. When forensic analysis on an IoT device is performed, then this forensic procedure automatically involves these areas as well. Many IoT devices are dispersed in diverse areas outside of user control, creating a challenge to the forensic investigative process. Furthermore, the IoT forensics process faces additional obstacles, such as the short time to capture evidence before it vanishes and the limited storage capacity of IoT devices [26].

There has been research in the field of forensic analysis on IoT devices, which involves many built-in mechanisms and machine learning models. Various frameworks and tools have been designed for attack detection in the IoT domain. Table 1 elaborates the forensic techniques of IoT devices. Kebande et al.'s [27] proposed framework elaborates the various steps involved for a digital investigation procedure, but it lacks a feedback and evaluation plan and is unconcerned about privacy and integrity.

Babun et al. [28] presented an IoT forensic framework for smart environments. The main objective of the proposed framework is to provide an effective IoT forensics framework that is both compatible with IoT devices and capable of investigating crimes committed in the IoT infrastructure. This framework is dependent on some data privacy principles, which is not suitable for IoT devices which have limited resources. Rios et al. [29] proposed a framework for sharing information related to cyberattacks with investigators. This framework is based on the PROFIT technique. It is basically a digital witness scheme for IoT attacks. The author proposes a new approach to researching IoT environments that considers a variety of privacy and security concerns.

The goal of Zia et al.'s [30] research paper is to provide an application-specific digital forensic investigation model. This model can be used in the IoT-related forensic investigation framework. In this research, a forensically significant artefact has been discovered in three common IoT applications such as home automation, smart cities, and smart wearables. Nieto et al. [31] presented a new approach to IoT forensics that prioritizes privacy. The goal is to put ISO/IEC 29100:2011's various principles into practice. All the key stages of any forensic process model are included in the proposed framework, as well as new stages such as review, initialization, and feedback. In contrast, the proposed framework is a privacy-conscious framework that considers a set of privacy principles that can increase data privacy, but it is ineffective for IoT devices with low resources.

Koroniotis et al. [32] proposed a framework called particle deep framework (PDF). PDF describes a forensic analysis of encrypted traffic in IoT networks. This framework completes the process in three stages. First, to cope with encrypted IoT network traffic, it identifies data flows and validates their integrity. Second, PDF uses the particle swarm optimization

(PSO) algorithm to automate the adoption of deep learning parameters. Finally, a deep neural network (DNN) is designed. DNN is based on a PSO for identification and tracing of the cyberattacks in the IoT network. This framework is also evaluated on smart home use case. Overall, the PDF achieves 98% accuracy. It is trained and tested for various datasets to evaluate the performance of the proposed framework.

Patil et al. [33] proposed a method for IoT device rapid digital forensics. Attackers are using vulnerabilities of IoT devices to target digital and physical infrastructures of systems. The forensic analysis of billions of networked devices is difficult due to the small amount of evidence they provide. As a result, computer forensics necessitates the creation of smart and fast digital research methodologies. As a result, digital forensics investigation frameworks are jam-packed with a wide range of tool-kits and apps to meet the demands of any criminal investigation. To determine which tools are required, specific objects are examined and evaluated using the microscope provided by the Digital Forensics Process. The proposed study raises knowledge of digital forensics issues, difficulties, and possibilities across a wide range of disciplines, including networks, IoT, cloud computing, and other areas.

Meffert et al. [34] proposed the FSAIoT (Forensic State Acquisition in the IoT). This technology records the state (open or closed) of IoT devices, allowing for a more accurate depiction of events. The researcher proposed the centralized forensic state acquisition controller that is utilized in three distinct configurations: controller to controller, controller to cloud, and controller to IoT device. This paper uses openHAB (a Linux-based open source IoT device controller) and a script to explain the notion of FSAC. Access to deleted and previous data is prohibited under the suggested plan. Second, physical access is necessary. Third, connecting various IoT devices is a significant barrier.

Oreški et al. [35] proposed a network forensics mechanism based on a genetic algorithm and a neural network. A genetic algorithm is used to optimize the network parameters instead of the time-consuming trial and error method. For testing, the BoT-IoT dataset is used, which contains traffic from IoT network assaults. This technology advances forensics by allowing accessible data to be used to predict infiltration and generating valuable prediction models using a neural network approach supported by a genetic algorithm.

Aslan et al. [36] reviewed different malware detection approaches along with their pros and cons. Signature-based detection involves the creation of a database of signatures that are unique values for malicious files. When malware is detected, its signatures are compared with existing signatures in the database. Behavior-based detection involves the use of tools to find the behavior of benign or malware programs. Specific features are extracted from the dataset and are classified using ML techniques. Heuristic-based detection involves the experience of humans and different ML techniques and is best at zero-day malware detection but cannot detect modern complicated malware. Model-checking-based detection involves checking behavior and grouping files with the same behavior which can be classified as malware or benign. Deep learning-based detection involves the use of artificial intelligence for malware detection. Steps involve feature extraction, identifying layers of the neural network, and the final output is analyzed to detect malware. Cloud-based detection involves uploading files to cloud storage, and malware is detected based on behavior and signatures stored on the database. Mobile device-based detection involves malware detection specifically for android devices that uses different features such as APIs, system calls, etc., that are fed to ML algorithms. ML-based detection involves malware detection using feature selection, extraction, and classification using different techniques. IoT devices are more vulnerable to malware because of the lack of security as the processing power of current devices is more than before, which results in security compromise in these devices.

Alrashdi et al. [37] addressed the IoT of security threats in a smart city framework, which requires new techniques for mitigation of these threats. They propose an anomaly detection IoT (AD-IoT) system that uses a random forest machine learning method to detect anomalies. The proposed solution is a cable for detecting IoT devices under attack

at the fog nodes. They used a recent dataset to demonstrate the model's correctness when evaluating the proposed model. The AD-IoT can successfully reach the maximum classification accuracy of 99.34% with the lowest false positive rate, according to the result.

Pilli et al. [38] proposed an application-specific forensic investigative model which combines the process of forensics as evidence collection, examination analysis, and application-specific reporting. This study highlighted the importance of this model in the traditional forensic model as well. The proposed model consists of three independent components as application-specific forensic, digital forensic, and forensic process. Information fetched from application-specific and digital forensic goes to the forensic process, which provides final evidence of the cyberattack. This study provides a holistic overview of the development of the proposed model, guidelines, and tools which will be helpful in forensic investigation to deal with evidence in different IoT systems.

Al-Sadi et al. [39] proposed a forensic investigation model for IoT devices. The advancements in technology have provided many open-source tools for investigating digital forensics in IoT networks. In this research, the most cost-effective tools for gathering and examining evidence from IoT devices have been proposed. IoT forensic investigation framework is also proposed which consisted of three layers, the IoT Application Layer which is responsible for monitoring communication between servers and application at cloud level, the Network Layer which is responsible for communication of all layers, and the IoT Device Layer which is responsible for the generating data which are stored at the same level or pushed on cloud servers. Overall, open-source tools along with the proposed framework are highly recommended for any forensic investigation of IoT networks because in any IoT network, multi-vendor tools make investigation complex.

Fagbola et al. [40] proposed a conceptual architecture for smart digital forensic readiness (SIoTDFR). SIoTDFR consists of six steps which include IoT device identification, IoT device monitoring, and capturing and conserving digital potential evidence for forensic readiness for shadow IoT device enterprises. It allows IoT-based environments to better prepare for security threats and criminal behavior. The SIoTDFR model is strong enough to catch even the smallest PDE, making criminal conduct easier to track in the case of an occurrence.

Riadi et al. [41] presented a forensic model to detect and identify the flooding attack on IoT devices. The proposed forensic model has many stages. These stages are linked and play their roles equally in the investigation of the incident. Network forensics lies between cloud forensics and device-level forensics. In this paper, the classification of IoT prominent attacks is also described. The Arduino and Bluetooth scheme is used as the IoT device. After configuration, an IP packet is delivered on target, and the port is attacked. To detect the attack, a log file located in Bluetooth Arduino Uno is analyzed using the Wireshark tool. It resulted that the overload issue was because of three IP addresses which committed illegal actions.

Scheidt et al. [42] proposed a generic framework for identification of IoT devices using DNA. DNA of IoT devices is generated from the buyer information and device identification number. With DNA of each IoT device, the fingerprint of attacks of these IoT devices can easily be traced. They implemented this proposed DNA approach on the Hybrid Forensic IoT Server, where every IoT device is registered and existing forensic investigation can easily apply.

Shrivastava et al. [43] performed classification of different attacks on IoT devices using machine learning algorithms and then performed forensics using Cowrie HoneyPot. The authors have employed J48 decision tree, Naive Bayes, random forest, and Support Vector Machines (SVM) for attack classification and concluded that SVM best classifies these attacks with an accuracy of 97.39%. They analyzed commands and found that malicious commands are homogeneous and follow a pattern, and it identifies how an attacker performs malicious activity. Li et al. [44] used the Contiki-NG operating system to investigate possible DoS attacks on an IoT network using the NSL-KDD dataset, focusing on low-power routing protocols, loss RPL networks, and PAN networks. The resulting

dataset is fed into eleven machine learning algorithms, which are then tested for their ability to classify different threats.

Baig et al. [45] examined an ADE-based DoS attack detection scheme for IoT sensors presented in wireless sensor networks (WSN). WSNs are a part of the current IoT paradigm. Furthermore, DoS attacks that use numerous network packets to target a given sensor node can disrupt normal operations and result in catastrophic losses for rescue services. Modules for data generation, feature ranking and creation, training, and testing are included in the proposed system. Moreover, the framework was put to the test in real-world IoT attack scenarios, and it outperforms current classification methods.

Koraniotis et al. [46], in their study, described network forensics on IoT botnets. A wide range of applications of IoT has been using IoT system targeting for attacks. To investigate activities of these IoT botnets in a small-scaled system, the digital forensic method is used that includes Deep Packet Analysis, Network Flow Analysis, Attack Recognition, Visualization of Network Traffic, Intrusion Detection Systems, and Honeypots. According to this study, deep learning is the best solution for network forensics of these botnets. This research examines the many models suggested in the subject of network forensics, as well as their strengths and weaknesses. Challenges to inhibit in an investigation of these IoT botnets are also described. This paper will help in our model to select the proper network forensic mechanism and machine learning model related to our problem.

Table 1. Literature survey of forensic analysis.

Research Paper	Technique	Outcomes
"Smart Digital Forensic Readiness Model for Shadow IoT Devices" [40]	SIoTDFR framework	This framework captures the tiniest PDE and makes criminal activity monitoring simple when incident occurs.
"Roadmap of Digital Forensics Investigation Process with Discovery of Tools" [33]	Forensic techniques analysis	This research provides holistic review of various tools and techniques for fast digital investigation framework.
"A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework" [32]	Framework depends on PSO and DNN	PDF achieves 98% accuracy by using MLP model. However, our attack scenarios are based on machine learning classifiers.
"A Comprehensive Review on Malware Detection Approaches" [36]	According to a comparison of various malware detection methodologies.	IoT devices are more vulnerable to malware because of a lack of protection.
"DistLog: A distributed logging scheme for IoT forensics" [47]	Modified Information Dispersal Algorithm which ensures log file's ability having a degree of (n-t) that log file have not been modified.	Anti-forensic techniques are being used against IoT log files, so they must be secured.
"Identification of IoT Devices for Forensic Investigation" [42]	DNA-based IoT device identification	With the help of DNA from IoT devices, it is possible to improve device management and examination efficiency during forensic investigations. Only certain types of devices are covered by this framework.

The above literature shows the relevant research work in the field of forensic analysis in IoT. First, the theoretical study of machine learning techniques is conducted for the exploration of the machine learning field. In addition, different attack detection methods using machine learning have been used by the researchers. Some researchers used previous datasets and applied machine learning algorithms to identify the attacks. However, our research provides a mechanism to generate a dataset for IoT device attack detection which is updated with time to incorporate more attacks. Then, different forensic analysis attack detection techniques using forensic tools are also explored. Our proposed framework resolves the evidence acquisition without interrupting the performance of IoT devices. It

incorporates both machine learning and forensic analysis using various tools for detection of attacks on IoT devices.

3. Methodology

The proposed framework for forensic analysis of IoT devices under attack consists of four modules, as shown in Figure 2. First, traffic generation of attack is responsible for attacks generation from Kali Linux system to IoT devices used under experimentation. Second, traffic redirection logging server and alerts/logs generation is responsible for redirection of traffic from IoT device to logging server, where traffic is analyzed and logs/alerts are generated when traffic is matched with rules written in logging server. Third, forensic analysis using forensic server is responsible for regeneration of logs captured by the logging server. Logs are regenerated and necessary information about the attack and attacker is extracted. Fourth, forensic analysis using machine learning model is responsible for detection of attacks using different machine learning models such as Random Forest Classifier [48], Decision Tree Classifier [49], Naïve Bayes Classifier [50], LDA Classifier [51], MLP Classifier [52], and assemble (Voting Classifier) [53]. The performance of these models is observed based on different evaluation parameters.

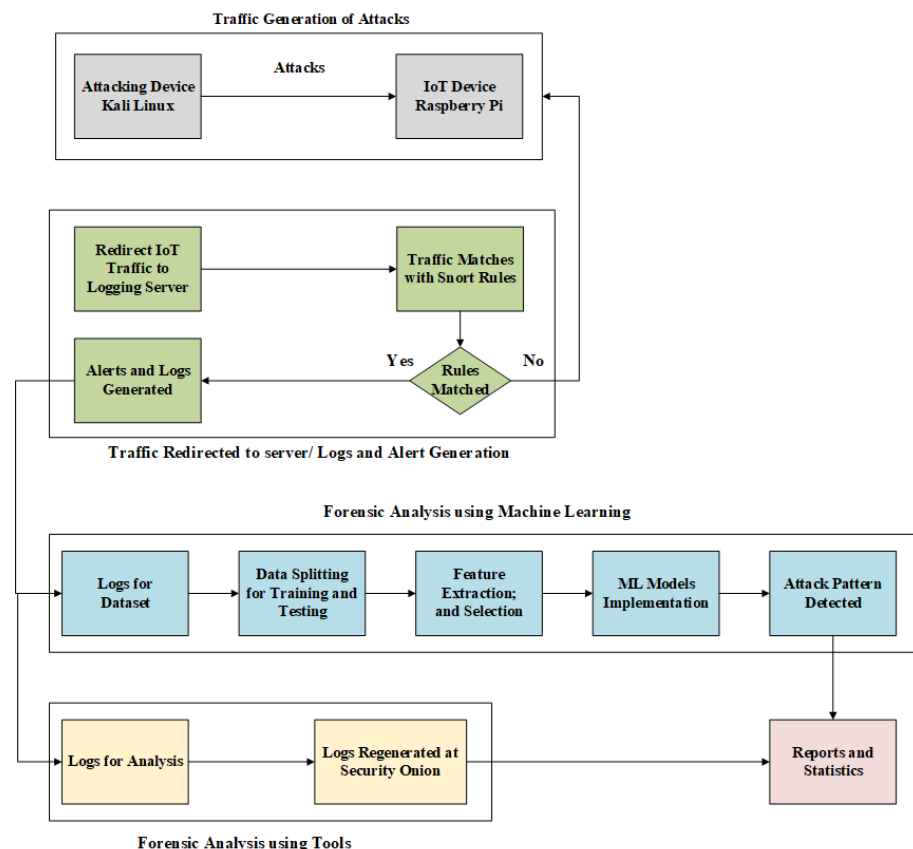
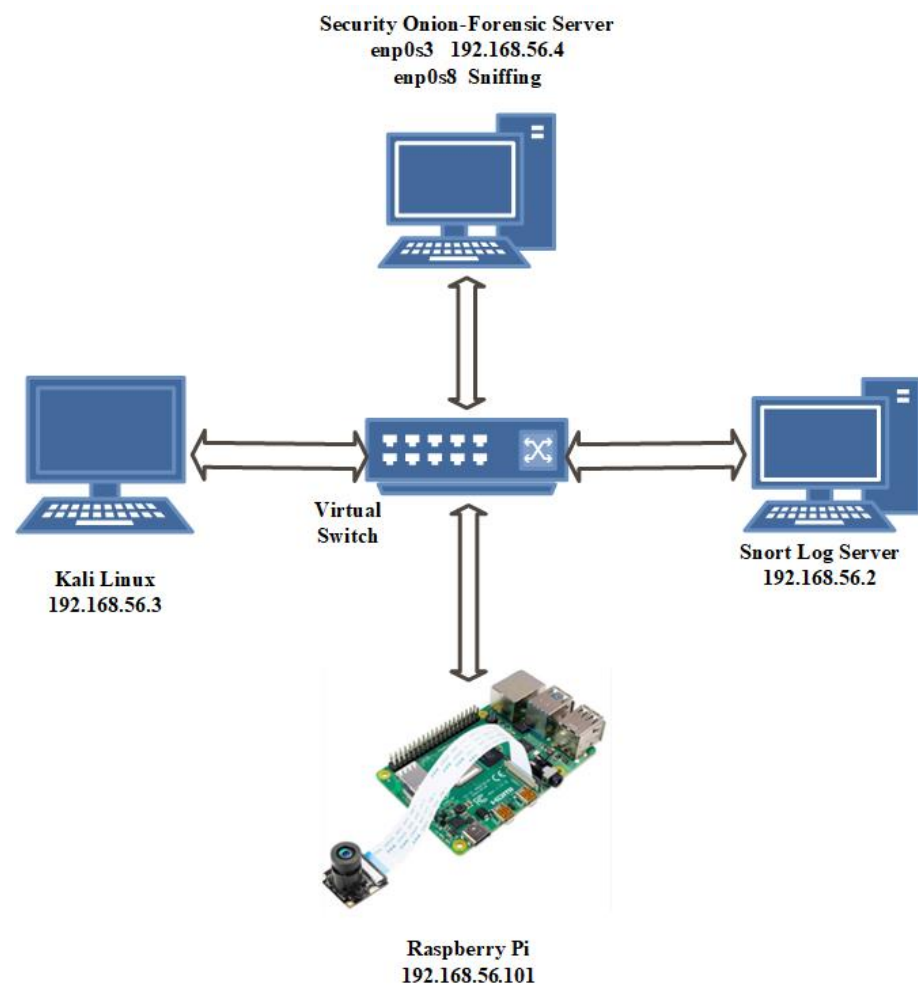


Figure 2. Proposed methodology for forensic analysis of IoT device.

In our designed experimentation, different devices are used such as Raspberry Pi as IoT device with Pi camera [54], snort as logging server [55], security onion as the forensic server [56], and Kali Linux to generate attack scenarios. All these devices are on the same network. Figure 3 shows the arrangement of the devices. Table 2 shows the configuration parameters of these devices.

Table 2. Configuration parameters of the experiment.

Items	IP Address	Specifications	Operating System
Kali Linux	192.168.56.3	Virtual Machine (Xeon 2 core, 4 GB RAM, 20 GB storage, 1G Ethernet)	Linux
Snort 2.9.3	192.168.56.2	Virtual Machine (Xeon 2 core, 2 GB RAM, 20 GB storage, 1G Ethernet)	Windows 10
Security Onion	192.168.56.4	Virtual Machine (Xeon 2 core, 4 GB RAM, 20 GB storage, 1G Ethernet)	Linux
Raspberry Pi	192.168.56.101	ARM dual Processor, 1 GB RAM, 20 GB Memory, Wired (1 G Ethernet)	Linux (Debian)

**Figure 3.** Network diagram of the proposed system.

3.1. Traffic Generation of Attacks

In the first step of the proposed framework, Kali Linux, which has an IP address of 192.168.56.3, is used to launch multiple attacks on the IoT device Raspberry Pi with the IP address 192.168.56.101. IoT devices are built on the board, which have options to connect multiple types of devices such as sensors, cameras, etc. The IoT board is designed in an open-source platform. Raspberry Pi is used as an IoT device in this experimentation, with a Pi camera attached to it. We used Kali Linux tools such as NMAP, Metasploit, HPING3, Ettercap, and Wireshark. These are attacks launched on Raspberry Pi.

1. Port Scanning using NMAP;
2. Brute Force Attack using Metasploit;

3. Denial of Service (DoS) Syn Flooding using HPING3;
4. Man in The Middle (MITM) ARP spoofing using Ettercap.

3.2. Traffic Redirection and Logs/Alert Generation for Attacks

In this framework, traffic of the IoT device is redirected to a logging server and logs are generated. To overcome the limitation of low memory and low processing power of IoT devices, traffic of IoT devices is redirected to a logging server with an IP address of 192.168.56.2. All devices are connected in an M2M manner, so they can directly communicate with each other. A logging agent (WAZUH) [57] was used for log storage on the third-party server. However, the ARM architecture of the Raspberry Pi does not support it. Then, IPtables are used to redirect the network traffic to the snort gateway with an IP address of 192.168.56.2. The snort is installed on the logging server. Snort rules are written and included in the configuration file for specific attacks. Snort analyze the traffic coming to IoT device and matches with the snort rules. If the match is successful by the detection engine of the logging server, then a snort alert for attacks is generated and logs are stored at the logging server. If not, then these packets are discarded. Logs of attacks generated by snorts are in pcaps format. CIC flow meter open-source tool [58] is used to convert these pcaps files to CSV files. Then, a CSV file of logs is used for the machine learning models because ML cannot be applied on pcap files.

3.3. Forensic Analysis Using Security Onion

After the detection of attacks by snort, logs are stored for analysis. These network logs contained information such as attack type, source address, destination address, port number, etc. Security onion, which has an IP address of 192.168.56.4, has two network cards. One is used for the management and the other is used for sniffing the network packets to find any malicious traffic in the network. These logs stored at the logging server solve the problem of evidence acquisition. Security onion has different built-in tools such as squil and squert which are used for analysis of these logs. Sguil is basically GUI of snort which is a CLI-based tool. These logs are regenerated to retrieve information about attacks and attackers, as we have captured the log files through the snort IDS.

3.4. Forensic Analysis Using Machine Learning

Attack detection on IoT devices is automated using machine learning algorithms. Snort-based detection is a manual process where each time we have run IDS for different attacks. By ML process, we automated the attack detection by using different types of classifiers, for which we labeled the logs in CSV format. After conducting preprocessing, we divided the dataset into training and testing. When features are extracted, we trained ML models and tested these models with the testing dataset and with the real-time traffic.

3.4.1. Flow Aggregation and Data Labeling

A CIC flow meter is used to convert the PCAP files to CSV format because machine learning models cannot use Pcap files. The traffic's behavior and statistical features were extracted. These characteristics are then fed into a machine learning model to detect threats to IoT devices. The data are then labeled in order to detect normal and abnormal behavior as Table 3 shows.

Table 3. Configuration parameters of the experiment.

Type	Category	Labels
Normal	Normal	0
Anomaly	Dos Syn flooding	1
	Brute force Attack	2
	MITM ARP spoofing	3
	Port Scanning	4

3.4.2. Data Preprocessing

To ensure data consistency, completeness, and soundness, we remove unnecessary fields, remove any characteristics that do not contribute to classification, encode categorical features, and numerical features are also scaled within the range of 0 and 1. Fields such as IP address, port address, and attack type that were previously used for labeling must be removed to avoid data bias and model performance degradation. To remove missing numbers and outliers, various strategies are used.

3.4.3. Splitting the Dataset for Training and Testing

Finally, the dataset is split into two categories: training and testing. The model is trained using the training data, and it is validated using the testing data. The dataset is split into 70% for the training and 30% for testing.

3.4.4. Feature Extraction and Selection

A machine learning algorithm's detection efficiency is degraded by correlated features. We utilized k-best [59], backward elimination [60], and feature significance [61] to choose features. For feature extraction, we chose k-best. The best accuracy results are obtained when $K = 10$, as shown in Table 4.

Table 4. Configuration parameters of the experiment.

Selected	Features
10	Flow_Byts/s, Pkt_Len_Var, Flow_Pkts/s, Fwd_Pkts/s, Bwd_Pkts/s, Bwd_IAT_Max,Src_Port, Bwd_IAT_Mean, Bwd_IAT_Tot, Flow_Duration

3.4.5. Training and Testing of Models

In the training phase, inputs and the labels are fed to machine learning algorithms after the feature extractor extracts the features from inputs. Different machine learning algorithms are applied in search of a best-fitting model. Each model works differently because of the different functions on which data are trained. In the testing phase, inputs are fed to the feature extractor to obtain the features. Then, these are fed to trained classifier models to extract the labels which are the predictions of these models. As previously, we labeled our data. So, the specific attack can be guessed from the label the model predicts. A generic diagram of the training and testing phase is shown in Figure 4.

We used some evaluation parameters to find out the efficiency of models for which we used confusion matrix's accuracy, precision, recall, and F1 score. The testing dataset is used for evaluation of trained models. If we used the training dataset, then accuracy or other parameters will not be a reflection of the true error. Cross-validation is also used to tune the models while training to improve the evaluation parameters. We evaluated our models based on accuracy, precision, F1 score, and recall.

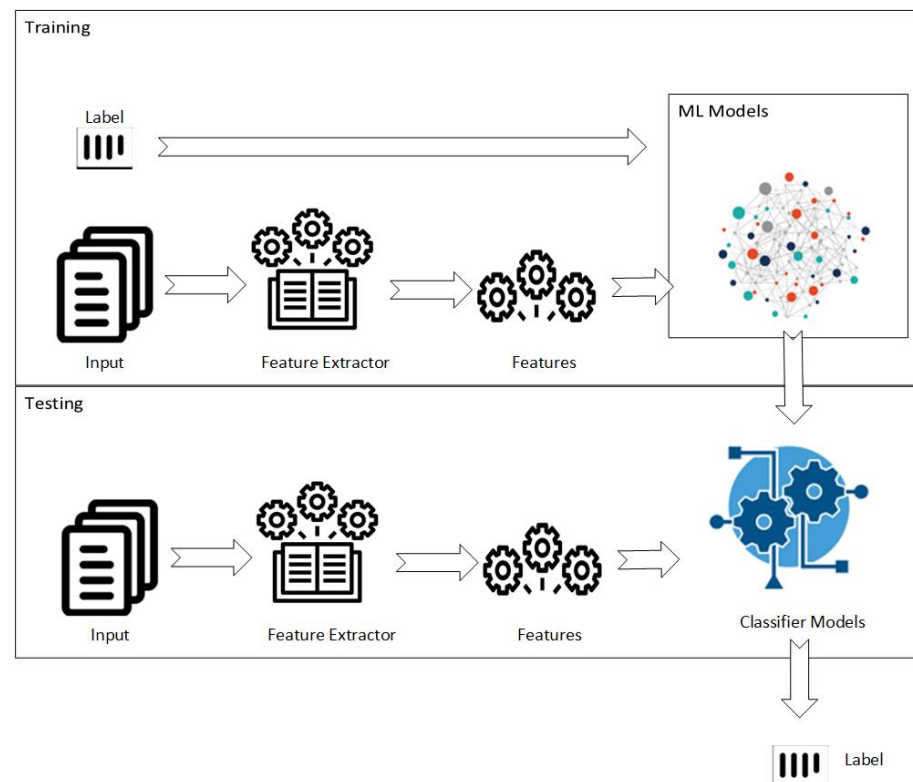


Figure 4. Training and testing of ML models.

3.5. Report and Statistics

The research environment proposed is designed for machine-to-machine (M2M) communication based on IoT devices. Unique IP addresses assigned to IoT devices and other machines are of the private network. The intention of designing the environment is to completely study the forensics on IoT communication using machine-to-machine connectivity. In this environment, multiple attacks are executed on IoT devices. All the network traffic of IoT devices is redirected to the snort logging server. The logs generated by the logging server are transferred to a forensic server security onion, where network packets are regenerated and analyzed. The dataset is used for machine learning models to automate this proposed model. With both machine learning analysis and forensic tools analysis deployed together, detection of cyberattack becomes more efficient and can evolve with time. After these, multiple reports are generated with the details about the type of attack and the number of times an attack is launched and the recommended action that could be taken. This forensic detail helps to draw the complete attack scenario and makes the tracing of attackers possible.

4. Results and Discussion

In the proposed framework, forensic analysis of IoT devices is performed. Low memory and low processing powered IoT devices perform predefined instructions. So, a built-in mechanism for forensics cannot be integrated. It is resolved by logging servers. In this research, cyberattacks are launched on an IoT device and traffic is redirected to the logging server using IPtables of the IoT device. On the logging server, traffic is matched with rules defined for these attacks. On a successful match, alerts and logs are generated and stored for forensic analysis. Forensic analysis is performed using both forensic tools and machine learning models. Our experimentation results are also compared with other proposed systems to evaluate the performance of the proposed framework.

4.1. Traffic Generation of Attacks on IoT Device

In the traffic generation of attacks on IoT device Raspberry Pi, we used Kali Linux as an attacking machine, which has the IP address 192.168.56.3, to launch attacks on victim IoT device with IP address 192.168.56.101. Multiple attacks such as Port scanning using NMAP tool, brute force attack using Metasploit tool, denial of service (DoS) Syn Flooding attack using HPING3, and Man in the Middle (MITM) ARP spoofing using Ettercap were launched on the IoT device.

Port scanning is used to locate the target IoT device's entry ports to conduct a cyberattack. The NMAP software is used to locate the target IoT device's open ports and services for this purpose. After gaining the necessary information about the targeted IoT device, an SSH connection has been established to connect virtually. We used Metasploit framework to launch brute force attacks on an IoT device that opened port 22. Metasploit tries all combinations of usernames and passwords to find the correct one. As Metasploit succeeds in matching the correct username and password, an SSH connection is established between the Kali Linux (attacking device) and Raspberry Pi (IoT device). It is analyzed that when Metasploit opens an SSH session with an IoT device. Then, on the IoT device, there is no footprint found that it has an SSH session with attacking system 192.168.56.3. For testing, a normal SSH session is created from another system having the IP address 192.168.56.4, which is security onion being used as the forensic server. Now, the IoT device shows that it had only one opened remote SSH session with the system having IP address 192.168.56.4 but did not show any information of SSH session from 192.168.56.3 attacking system.

Denial of service (DoS) attack is launched through the HPING3 and Metasploit software using the Kali Linux platform. Multiple instances of DoS attacks were started to make the attack scenario more severe. After some time, the DoS attacks occupy all the resources of the IoT device and, as a result, the IoT device stops communication with the other devices. After the DoS attack completion, logs are analyzed to find out the cause and destruction of the attack. The log files generated by the logging server are gathered for forensic purposes. Log files of the DoS attack are monitored through the Wireshark and squert. Wireshark shows the detail of captured packets and highlights that source address 192.168.5.3 attack on port 80 of destination address 192.168.56.101 using TCP-Out-of-Order packet sequence number. We observed the traffic coming to the IoT device using Wireshark which shows plenty of traffic coming from the attacker but Raspberry Pi failing to respond to this traffic and halts its services.

Man-in-Middle attack on IoT devices using Ettercap is launched. ARP spoof changes the ARP entries of target systems with the MAC address of the attacking system. All the communication between the victim machines is intercepted by the attacker because the attacking system resides between them. To verify that the attack is successful, the ARP tables of both source and destination change the destination MAC addresses of each other with the attacking system. For further analysis, the log files generated by the logging server are analyzed through Wireshark. Wireshark also points out the duplication of addresses in the network packets. The target IP addresses are of both the attacking device and victim machine and are set as 192.168.56.3 and 192.168.56.101, respectively.

Ettercap provides many attack options, we selected ARP poisoning. Remotely sniffing is selected. Ettercap starts listening at the eth0 of the IoT device, which has the IP address of 192.168.56.101. While an IoT device tries to make communicate with any other device, then it can be observed via Ettercap. We opened the Raspbian browser, then it can be observed.

4.2. Traffic Redirected to Logging Server

Traffic is redirected to the logging server to store the logs on a third-party server, where forensics on these logs is applied. Because of the limitation of low memory and low processing power of IoT devices, it is impossible to store network logs on IoT devices. Therefore, network traffic is redirected to the logging server to store the network logs. For the redirection of packets, we used IPtables. To perform this action, we enabled the forwarding on the device. After enabling the forwarding, which is also called routing, we

access our IPtables to configure. So, the required packets can be redirected to the logging server which is installed on Windows 10 as Snort IDS with IP address 192.168.56.2. A packet filter framework named Netfilter is generally used in the Linux kernel. This framework allows a Linux machine with a sufficient number of network adapters to function as a NAT router. We used the command utility “IPtables” to establish complicated rules for packet modification and filtering. The most important NAT rules are pre-routing, output, and post-routing. Packets that have just arrived at the network interface are routed using pre-routing. Depending on the routing decision, the traffic will either be interpreted locally or transmitted to another machine on a different network interface. The routing decision is taken after the traffic has passed through pre-routing. If the receiver is a local machine, the traffic will be sent to the appropriate process, and we will no longer have to worry about NAT. If the receiver is in a separate network with a different local network, the packet will be forwarded to that interface if the machine is set to do so.

```
# iptables -t nat -A PREROUTING -I eth0 -j redirect-gateway 192.168.56.2
```

4.3. Logs and Alert Generation by Logging Server Snort

When the traffic from Raspberry Pi is redirected to logging server snort, then the logs and alerts generation process starts. Redirected traffic is received on the Ethernet port of a logging server having the IP address of 192.168.56.2. Snort matches traffic with rules written for attacks. If the match is successful, then the log file is generated. The log files are generated with a different name every time the command is started. Moreover, new log files are generated after the log file reached its defined maximum capacity. The generation of log files with a new name arises a problem while implementing the agent-based log monitoring. So, individual log files of respective attacks are generated manually. These files are stored on the logging server with different names. Snort is configured to capture the malicious traffic by writing the rules in the file local.rule, which is presented in the directory of snort. When snorts starts, it scans for different interfaces to track the malicious traffic. After setting up the rules for all attacks, the configuration file is updated and tested; these rules are incorporated in the snort rules. After running snort IDS mode, it monitors the network traffic which was redirected from the Raspberry Pi. This traffic is compared with rules. When a DoS attack is launched, then incoming traffic matches with the rule and snort starts to generate alerts and stores the logs of this traffic. These logs are further analyzed with forensic servers and machine learning models for the detection of attacks. Figure 5 shows that log directory. The rules for denial of service (DoS) attack is described below. Similarly, rules for other attacks are also incorporated into the rules directory.

```
alert tcp any any - $HOME_NET 80 (flags: S; msg:" possible DoS attack Type: SYN flood"; flow: stateless; sid:1000003; detection filter: track by_dst; count: 100; seconds 5;)
```

Name	Date modified	Type	Size
alert.ids	1/18/2021 1:24 AM	IDS File	7 KB
snort.log.1610961216	1/18/2021 1:14 AM	1610961216 File	1 KB
snort.log.1610961512	1/18/2021 1:24 AM	1610961512 File	5 KB
snort.log.1610963194	1/18/2021 1:46 AM	1610963194 File	1 KB
snort.log.1610965658	1/18/2021 2:27 AM	1610965658 File	1 KB
snort.log.1611036679	1/18/2021 10:11 PM	1611036679 File	1 KB
snort.log.1611037357	1/18/2021 10:22 PM	1611037357 File	1 KB
snort.log.1611038302	1/18/2021 10:38 PM	1611038302 File	1 KB
snort.log.1611039686	1/18/2021 11:02 PM	1611039686 File	6 KB
snort.log.1611044269	1/19/2021 12:48 AM	1611044269 File	5,115 KB
snort.log.1611126393	1/19/2021 11:10 PM	1611126393 File	1,203 KB
snort.log.1611127955	1/20/2021 12:05 AM	1611127955 File	1,016 KB
snort.log.1611130978	1/20/2021 12:23 AM	1611130978 File	1 KB

Figure 5. Logs and alerts generated by the snort.

4.4. Forensic Analysis Using Security Onion

Forensic analysis using security onion is applied on logs captured by the logging server snort. As Figure 6 shows, logs based on rules are regenerated in the security onion. We used security onion as the forensic server, which is a free and open-source Linux distribution with security monitoring, intrusion detection, and log management capabilities. It includes many tools; we used sgul and squirt for logs analysis. Snort is CLI based, where logs are not properly analyzed. Therefore, sgul is GUI for snort and logs are easily visualized. Security onion has two network interface cards (NICs). One interface enp0s3, which has the IP address of 192.168.56.4, is used for the management of the security onion, while other interfaces enp0s8 work in the promiscuous mode. It means that it is connecting to the virtual server and monitoring all the network traffic presented in this network. It sends packets to all network interfaces of the complete network to obtain information about the connected devices. We have captured the log files through the snort IDS. These logs are regenerated through security onion. There are many ways to import these logs as follows

- **tcpreplay:** Logs captured at the logging server can be imported as the new traffic with current timestamp.
- **so-replay:** Import all logs samples in /opt/samples and replay them with the current timestamp.
- **so-import-pcap:** Logs captured at the logging server can be imported as the new traffic with the original timestamp and date.

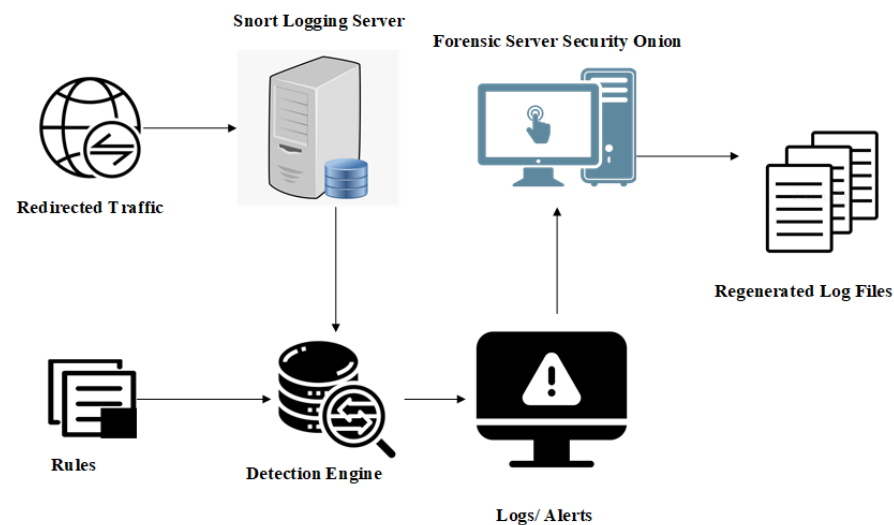


Figure 6. Integration of logging server and forensic server.

Sguil with a user-friendly interface gives you access to real-time events, session statistics, and raw packets captured. While setting up the sguil, we have to check NIC whom we want to monitor. For forensic purposes, the log files from the logging server are analyzed through Wireshark and sguil. Figure 7 shows the regenerated alerts of the logs and details about the attacker IP address, port number, etc. Each type of attack is with a different color. It is basically a GUI for snort being used as the logging server. When packet's inspection pane and obtaining the details of each log record, details about the attack can easily be fetched. When brute force attack generated from Kali Linux to obtain the SSH connection, then it applies the IDS rule "ET SCAN Potential SSH Scan OUTBOUND" and shows that IP address 192.168.5.3 is trying to obtain SSH service access on IP address 192.168.5.101, but that exceeds the SSH login limit within the specific period.

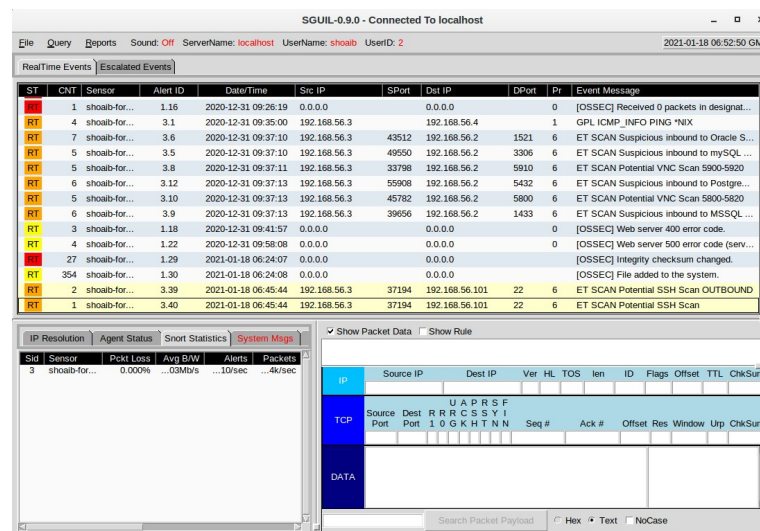


Figure 7. Sguil interface shows the alert for attacks.

Squert, on the other hand, uses metadata and time series representations to help provide further context to the attack events. When the DoS attack is launched from Kali Linux 192.168.56.3 to IoT device, Raspberry Pi 192.168.56.101, snort generates the alerts for SYN DoS attack on the IoT device and logs are stored. When these are regenerated in the forensic server, squert provides a holistic summary about each attack. Without having to understand the underlying data or events, squert's visualization tools help detect suspicious sessions or activities. As the views are changed, the events are displayed in a variety of formats, making it easier to understand the packets and information. This shows the series of events when the attack occurred with targeted IP address and ports as shown in the Figure 8.

Log file of the DoS attack is also analyzed through the squert. Squert analysis shows that their source, address 192.168.55.3, attacks, 18,705 times on the SSH service of destination address 192.168.56.101. Squert also shows that during a DoS attack, the attacker does not establish a communication connection with the target and bombarded the packets having no payload. On the forensic server, the log file gathered from the log server is analyzed and found the traces of NMAP port scanning. It is further explored to find the information of the IoT device such as operating system, open ports, and services, SSH-Host Key, Traceroute, Hope count, and MAC address, etc.

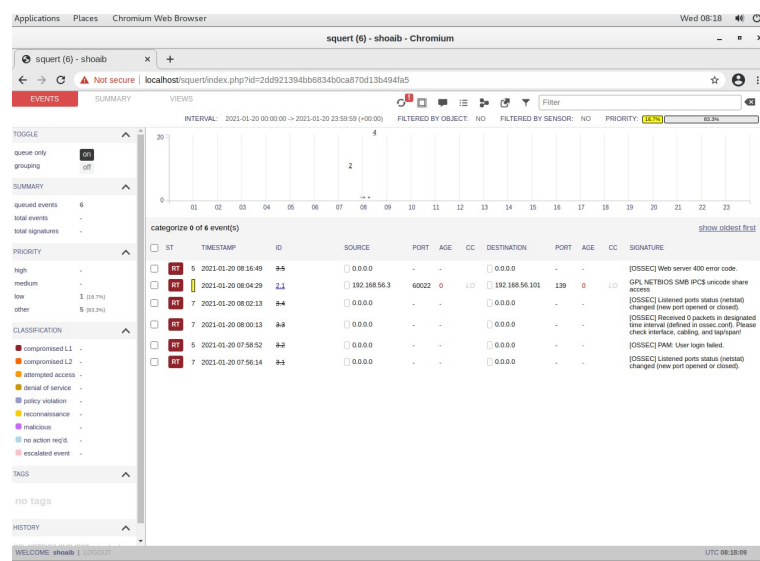


Figure 8. Squert interface shows attacks information.

4.5. Forensic Analysis Using Machine Learning

The purpose of forensic analysis using machine is to automate the process of attack detection. Numerous models of ML are used on the dataset generated from the logs captured by the logging server. We used various feature extraction and selection techniques such as feature importance, k-best, and backward elimination. After performing different tests, we finally used the k-best feature extraction and selection technique to use the most efficient feature of the dataset. This dataset is divided into 70% and 30% ratio for training and testing. To evaluate the models, we incorporated several parameters such as accuracy, recall, F1 score, and precision. Table 5 displays performance comparison of ML algorithms. A confusion matrix displays performance of various classifiers applied on a test dataset through known true values. The terms used in the confusion matrix are defined as False Positive (FP), True Positive (TP), False Negative (FN), True Negative (TN), respectively.

Table 5. Performance comparison of ML algorithms.

Algorithm	Accuracy	Precision	Recall	F Score
Random Forest	0.96	0.96	0.88	0.92
Decision Tree	0.97	0.96	0.89	0.93
Naive Bayes	0.45	0.25	0.84	0.39
LDA	0.77	0.73	0.77	0.73
MLP	0.81	0.72	0.81	0.73
Ensemble (Voting Classifier)	0.97	0.97	0.97	0.97

4.5.1. Accuracy

The accuracy can be defined as a total classifications successfully predicted in a model divided by the total predictions made, and it can be represented as below [59].

$$Accuracy = \frac{TN + TP}{FN + TN + TP + FP} \quad (1)$$

4.5.2. Precision

The precision can be defined as a total true positives divided by total true positives added to the total false positives.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

4.5.3. Recall

It is the ratio that describes how many samples are correctly predicted positive samples in all relevant collections of samples. It is also known as “sensitivity measure”.

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

4.5.4. F-Score

F1 score is a process of integrating the model’s accuracy and recall. It can also be defined by the harmonic means of these parameters [59].

$$F1\text{-score} = \frac{2 * Recall * Precision}{Recall + Precision} \quad (4)$$

4.6. Results of Proposed Framework ML Model

In the ML approach, a dataset derived from logs is utilized. In order to preprocess the data, we used a variety of techniques. We remove unnecessary fields, remove any characteristics that do not contribute to classification, encode categorical features, and

numerical features are also scaled within the range of 0 and 1 to ensure data consistency, completeness, and soundness. For feature selection and extraction, we used a variety of techniques such as feature importance, backward elimination, and k-best. Changing k-values in the k-best technique yielded efficient results. The dataset is divided into two parts: training and testing, with a 70/30 split. As mentioned in the previous section, various classifiers are used to train and test ML models. The decision tree classifier achieved 97.29% accuracy after training and testing multiple classification models. The accuracy of various machine learning models used in our proposed system is depicted in Figure 9. In the forensic process, these models aid in the detection of attacks faster and more reliably.

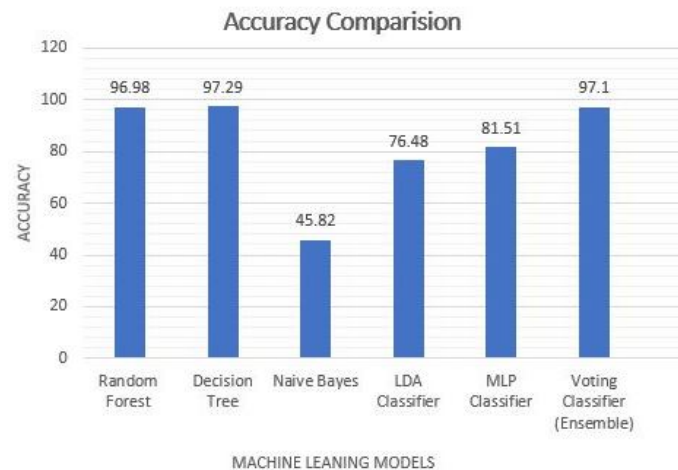


Figure 9. Accuracy comparison of ML models.

4.7. Comparison with Existing Framework

When compared to existing anomaly detection models, the results of our proposed framework outperformed the “Scheme for Generating a dataset for Anomalous Activity Detection in IoT Networks” (ullah2020scheme) machine learning technique for attack detection [62]. The accuracy of various ML approaches is shown in Figure 10. The authors first described the flaws in several intrusion detection datasets in this framework. Second, they provided a new dataset for detecting IoT device attacks. Finally, a new intrusion detection technique is proposed based on the generated dataset. For classification, various machine learning models are used. The proposed techniques achieve an accuracy of 88%. On the other hand, our proposed framework not only provides a scheme to generate the dataset for IoT devices but also performs the forensic analysis using forensic server-based tools and machine learning models with accuracy of 97.29%.

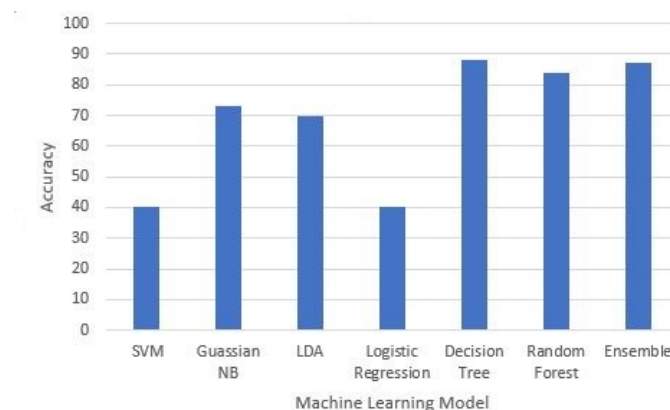


Figure 10. Comparison with existing framework.

4.8. Real-Time Deployed Model

Because the Raspberry Pi has an onboard device that can connect to multiple IoT devices such as sensors and cameras [63], it can be used as an IoT device. We connected a Pi camera to the Raspberry Pi to test the efficiency of our proposed system. As a result, we used the Pi camera traffic dataset to evaluate our proposed system's efficiency. The accuracy of the model is slightly decreased, but it is still efficient than the existing approach mentioned in the above section. The progress of the trained model on the testing dataset and real-world data using the Pi camera is shown in Figure 11. The blue bar shows the accuracy of models in the training testing phase of models, while the orange bar shows the accuracy of a real-time deployed model with a Pi camera attached to the Raspberry Pi IoT device.

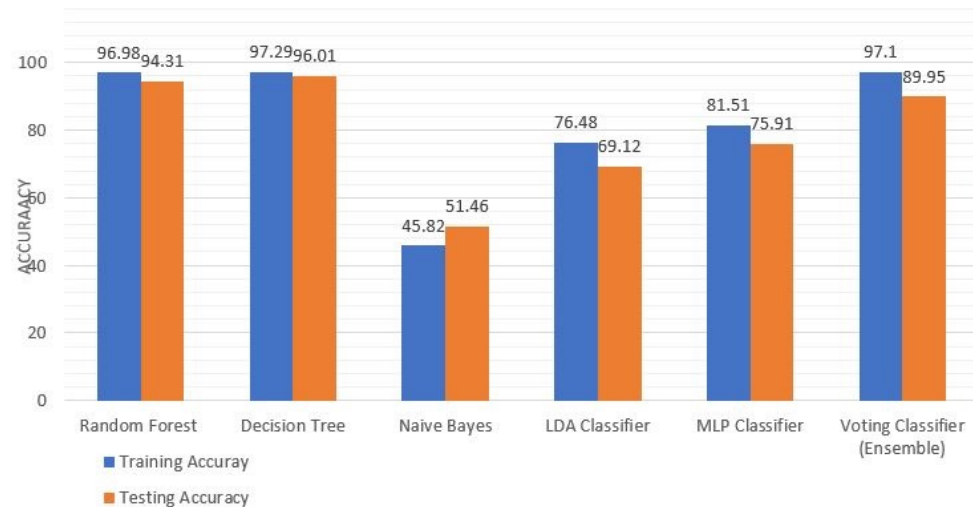


Figure 11. Accuracy comparison of testing and real-time deployed testing.

4.9. Comparative Analysis with Various Frameworks

IoT devices are gaining traction in every industry, from smart homes to smart cities. However, because of the security flaws in these devices, entire systems are compromised, and no evidence is found. This is due to a lack of processing power and memory constraints. To address these issues, we proposed this framework which stores logs as evidence of these attacks on a third-party logging server called snort and then applies forensic analysis to them using forensic server security onion and machine learning algorithms. Through attacks that are quickly detected, ML-based analysis improves the framework's efficiency. It is applicable not only in a post-analytical setting, but also in a real-time setting. Snort rules are required for all of this work. Real-time alerts and logs are generated when traffic matches these rules, which were also investigated in order to determine the source of the attack and the perpetrator. The comparison of our proposed framework with similar frameworks is shown in Table 6. Based on various parameters that distinguish our research from previous work, our framework outperforms these frameworks.

Table 6. Comparative analysis with various frameworks.

Research Paper	Logs Acquisition	Logs Analysis Using Forensic Server	ML Based Logs Analysis	Real-Time Traffic Monitoring	Snort Rules Generation for Attacks	Dataset Generation Scheme
Forensic Analysis on Internet of Things (IoT) Device using Machine to Machine (M2M) Framework Smart Digital Forensic	✓	✓	✓	✓	✓	✓
Readiness Model for Shadow IoT Devices [40]	✓	×	×	✓	×	×
Roadmap of Digital Forensics Investigation Process with Discovery of Tools [33]	✓	✓	×	×	×	×
A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework [32]	×	×	✓	✓	×	×
A Comprehensive Review on Malware Detection Approaches [36]	×	×	✓	×	×	×
DistLog: A distributed logging scheme for IoT forensics [47]	✓	✓	×	✓	×	×
Identification of IoT Devices for Forensic Investigation [42]	✓	✓	×	×	×	×

5. Conclusions and Future Work

Forensic analysis refers to the deep investigation of the crime after it happened to explore the reasons behind the crime. Our proposed forensic analysis system overcomes the low power and low memory limitation of IoT devices. The proposed forensic system makes the forensics of IoT devices in a directly connected environment more efficient and reliable. Without interruption of communication between devices, network traffic is redirected to the logging server, and traffic is analyzed by comparing with rules. These logs of malicious traffic are stored and can be regenerated in the forensics server by different methods. After capturing the logs of attacks launched on IoT devices, not only are the logs regenerated, but a dataset is also created. Multiple machine learning models are trained and tested for the detection of attacks. The decision tree algorithm performed well, with the highest accuracy of 97.29%. Our proposed system is tested in a real-time environment when a Pi camera is installed in the network. Performance of machine learning models was slightly decreased with decision tree with the highest accuracy of 96.01%. Then, multiple reports are generated to describe the details about the type of attack, the number of times an attack was launched, and the recommended action that could be taken. This forensic detail will help to draw the complete attack scenario and make the tracing of attackers possible.

The area of this research can be extended by adding more attacks with categories and sub-categories. Our dataset is limited to the most common attack on IoT devices. The attributes/features can be enhanced by applying more techniques. We can also include the dataset of IoT devices that we used in our daily routine to enhance the footprint of ML-based forensic analysis.

Author Contributions: Conceptualization, M.S.M., Y.S., J.A., A.A., M.H.J. and H.H.; methodology, M.S.M., Y.S. and J.A.; software, M.S.M., Y.S., A.U.R. and M.S.; validation, Y.S., J.A., M.H.J. and H.H.; formal analysis, M.S.M., Y.S., J.A., M.H.J., A.U.R. and H.H.; investigation, M.S.M., Y.S. and J.A.; resources, A.A., M.H.J., M.S. and H.H.; data curation, M.S.M., Y.S. and J.A.; writing—original draft preparation, M.S.M., Y.S. and M.H.J.; writing—review and editing, A.A., Y.S. and M.S.; visualization, M.S.M. and J.A.; supervision, A.A., M.S. and H.H.; project administration, A.A., M.S. and H.H.; funding acquisition, A.A., M.S. and H.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the King Saud University, Riyadh, Saudi Arabia, through researchers supporting project number RSP-2021/184.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data may be provided on request.

Acknowledgments: The authors also acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC) and the New Brunswick Innovation Foundation (NBIF) for the global project. These granting agencies did not contribute to the design of the study and the collection, analysis, and interpretation of data.

Conflicts of Interest: All authors declare no conflict of interest.

References

1. Vishwakarma, R.; Jain, A.K. A survey of DDoS attacking techniques and defence mechanisms in the IoT network. *Telecommun. Syst.* **2020**, *73*, 3–25. [CrossRef]
2. Yang, Y.; Wu, L.; Yin, G.; Li, L.; Zhao, H. A survey on security and privacy issues in Internet-of-Things. *IEEE Internet Things J.* **2017**, *4*, 1250–1258. [CrossRef]
3. Javaid, M.; Khan, I.H. Internet of Things (IoT) enabled healthcare helps to take the challenges of COVID-19 Pandemic. *J. Oral Biol. Craniofac. Res.* **2021**, *11*, 209–214. [CrossRef] [PubMed]
4. Hossain, E.; Khan, I.; Un-Noor, F.; Sikander, S.S.; Sunny, M.S.H. Application of big data and machine learning in smart grid, and associated security concerns: A review. *IEEE Access* **2019**, *7*, 13960–13988. [CrossRef]
5. Alladi, T.; Chamola, V.; Sikdar, B.; Choo, K.K.R. Consumer IoT: Security vulnerability case studies and solutions. *IEEE Consum. Electron. Mag.* **2020**, *9*, 17–25. [CrossRef]
6. Almogren, A.S. Intrusion detection in Edge-of-Things computing. *J. Parallel Distrib. Comput.* **2020**, *137*, 259–265. [CrossRef]
7. Sikder, A.K.; Petracca, G.; Aksu, H.; Jaeger, T.; Uluagac, A.S. A survey on sensor-based threats to internet-of-things (IoT) devices and applications. *arXiv* **2018**, arXiv:1802.02041.
8. Hussain, F.; Abbas, S.G.; Husnain, M.; Fayyaz, U.U.; Shahzad, F.; Shah, G.A. IoT DoS and DDoS attack detection using ResNet. In Proceedings of the 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 5–7 November 2020; pp. 1–6.
9. Welch, L.O. More Than Half of IoT Devices Vulnerable to Severe Attacks | Statista. Available online: <https://threatpost.com/half-iot-devices-vulnerable-severe-attacks/153609/> (accessed on 3 May 2021).
10. Paul, D. IoT Devices See More Than 1.5bn Cyberattacks So Far This Year. Available online: <https://www.digit.fyi/iot-security-kaspersky-research-attacks/> (accessed on 14 February 2022).
11. Stergiou, C.; Psannis, K.E.; Kim, B.G.; Gupta, B. Secure integration of IoT and cloud computing. *Future Gener. Comput. Syst.* **2018**, *78*, 964–975. [CrossRef]
12. Hussain, F.; Abbas, S.G.; Fayyaz, U.U.; Shah, G.A.; Toqeer, A.; Ali, A. Towards a universal features set for IoT botnet attacks detection. In Proceedings of the 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 5–7 November 2020; pp. 1–6.
13. Rawlinson, K. HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack. Available online: <https://www.hp.com/us-en/hp-news/press-release.html?id=1744676> (accessed on 15 February 2022).
14. Yousefnezhad, N.; Malhi, A.; Främling, K. Security in product lifecycle of IoT devices: A survey. *J. Netw. Comput. Appl.* **2020**, *171*, 102779. [CrossRef]

15. Tawalbeh, L.; Muheidat, F.; Tawalbeh, M.; Quwaider, M. IoT Privacy and security: Challenges and solutions. *Appl. Sci.* **2020**, *10*, 4102. [CrossRef]
16. Mariyanayagam, D.; Shukla, P.; Virdee, B.S. Bio-inspired framework for security in IoT devices. In *Intelligent Sustainable Systems*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 749–757.
17. Gupta, D.N.; Kumar, R.; Kumar, A. Federated Learning for IoT Devices. In *Federated Learning for IoT Applications*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 19–29.
18. Rughani, P.H. IoT evidence acquisition—Issues and challenges. *Adv. Comput. Sci. Technol.* **2017**, *10*, 1285–1293.
19. Karabiyik, U.; Akkaya, K. Digital forensics for IoT and WSNS. In *Mission-Oriented Sensor Networks and Systems: Art and Science*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 171–207.
20. Nayak, R.K. Forensic Analysis. Available online: <https://www.sciencedirect.com/topics/chemistry/forensic-analysis> (accessed on 14 April 2021).
21. Lord, N. What Is Security Incident Management? The Cybersecurity Incident Management Process, Examples, Best Practices, and More. Available online: <https://digitalguardian.com/blog/what-security-incident-management-cybersecurity-incident-management-process> (accessed on 28 March 2021).
22. Dosal, E. How a Network Security Audit Works & Why It's Important. Available online: <https://www.compuquip.com/blog/network-security-audit-works> (accessed on 13 April 2021).
23. Haider, S.K.; Jiang, A.; Almogren, A.; Rehman, A.U.; Ahmed, A.; Khan, W.U.; Hamam, H. Energy Efficient UAV Flight Path Model for Cluster Head Selection in Next-Generation Wireless Sensor Networks. *Sensors* **2021**, *21*, 8445. [CrossRef] [PubMed]
24. Horsman, G. An “order of data acquisition” for digital forensic investigations. *J. Forensic Sci.* Available online: <https://pubmed.ncbi.nlm.nih.gov/34997585/> (accessed on 13 April 2021).
25. Ghabban, F.M.; Alfadli, I.M.; Ameerbakhsh, O.; AbuAli, A.N.; Al-Dhaqm, A.; Al-Khasawneh, M.A. Comparative Analysis of Network Forensic Tools and Network Forensics Processes. In Proceedings of the 2021 2nd International Conference on Smart Computing and Electronic Enterprise (ICSCEE), Cameron Highlands, Malaysia, 15–17 June 2021; pp. 78–83.
26. MacDermott, A.; Baker, T.; Shi, Q. Iot forensics: Challenges for the ioa era. In Proceedings of the 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 26–28 February 2018; pp. 1–5.
27. Kebande, V.R.; Ray, I. A generic digital forensic investigation framework for internet of things (IoT). In Proceedings of the 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), Vienna, Austria, 22–24 August 2016; pp. 356–362.
28. Babun, L.; Sikder, A.K.; Acar, A.; Uluagac, A.S. Iotdots: A digital forensics framework for smart environments. *arXiv* **2018**, arXiv:1809.00745.
29. Nieto, A.; Rios, R.; Lopez, J. IoT-forensics meets privacy: towards cooperative digital investigations. *Sensors* **2018**, *18*, 492. [CrossRef] [PubMed]
30. Zia, T.; Liu, P.; Han, W. Application-specific digital forensics investigative model in internet of things (IoT). In Proceedings of the Proceedings of the 12th International Conference on Availability, Reliability and Security, Reggio Calabria, Italy, 29 August–1 September 2017; pp. 1–7.
31. Nieto, A.; Rios, R.; Lopez, J. A methodology for privacy-aware IoT-forensics. In Proceedings of the 2017 IEEE Trust-com/BigDataSE/ICISS, Sydney, NSW, Australia, 1–4 August 2017; pp. 626–633.
32. Koroniotis, N.; Moustafa, N.; Sitnikova, E. A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework. *Future Gener. Comput. Syst.* **2020**, *110*, 91–106. [CrossRef]
33. Patil, A.; Banerjee, S.; Jadhav, D.; Borkar, G. Roadmap of Digital Forensics Investigation Process with Discovery of Tools. *Cyber Secur. Digit. Forensics* **2022**, *100*, 241–269.
34. Meffert, C.; Clark, D.; Baggili, I.; Bretinger, F. Forensic State Acquisition from Internet of Things (FSAIoT) A general framework and practical approach for IoT forensics through IoT device state acquisition. In Proceedings of the 12th International Conference on Availability, Reliability and Security, Reggio Calabria, Italy, 29 August–1 September 2017; pp. 1–11.
35. Oreški, D.; Andročec, D. Genetic algorithm and artificial neural network for network forensic analytics. In Proceedings of the 2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 28 September–2 October 2020; pp. 1200–1205.
36. Aslan, Ö.A.; Samet, R. A comprehensive review on malware detection approaches. *IEEE Access* **2020**, *8*, 6249–6271. [CrossRef]
37. Alrashdi, I.; Alqazzaz, A.; Aloufi, E.; Alharthi, R.; Zohdy, M.; Ming, H. Ad-iot: Anomaly detection of iot cyberattacks in smart city using machine learning. In Proceedings of the 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 7–9 January 2019; pp. 0305–0310.
38. Pilli, E.S.; Joshi, R.; Niyogi, R. A generic framework for network forensics. *Int. J. Comput. Appl.* **2010**, *1*, 1–6. [CrossRef]
39. Al-Sadi, M.B.; Chen, L.; Haddad, R.J. Internet of Things digital forensic investigation using open source gears. In Proceedings of the SoutheastCon 2018, St. Petersburg, FL, USA, 19–22 April 2018; pp. 1–5.
40. Fagbola, F.I.; Venter, H. Smart Digital Forensic Readiness Model for Shadow IoT Devices. *Appl. Sci.* **2022**, *12*, 730. [CrossRef]
41. Rizal, R.; Riadi, I.; Prayudi, Y. Network forensics for detecting flooding attack on internet of things (IoT) device. *Int. J. Cyber-Secur. Digit. Forensics* **2018**, *7*, 382–390.
42. Scheidt, N.; Adda, M. Identification of iot devices for forensic investigation. In Proceedings of the 2020 IEEE 10th International Conference on Intelligent Systems (IS), Varna, Bulgaria, 28–30 August 2020; pp. 165–170.

43. Shrivastava, R.K.; Bashir, B.; Hota, C. Attack detection and forensics using honeypot in IoT environment. In *Proceedings of the International Conference on Distributed Computing and Internet Technology*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 402–409.
44. Liu, J.; Kantarci, B.; Adams, C. Machine learning-driven intrusion detection for Contiki-NG-based IoT networks exposed to NSL-KDD dataset. In *Proceedings of the Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning*, Linz, Austria, 13 July 2020; pp. 25–30.
45. Baig, Z.A.; Sanguanpong, S.; Firdous, S.N.; Nguyen, T.G.; So-In, C. Averaged dependence estimators for DoS attack detection in IoT networks. *Future Gener. Comput. Syst.* **2020**, *102*, 198–209. [[CrossRef](#)]
46. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [[CrossRef](#)]
47. Noura, H.N.; Salman, O.; Chehab, A.; Couturier, R. DistLog: A distributed logging scheme for IoT forensics. *Ad Hoc Netw.* **2020**, *98*, 102061. [[CrossRef](#)]
48. Pal, M. Random forest classifier for remote sensing classification. *Int. J. Remote Sens.* **2005**, *26*, 217–222. [[CrossRef](#)]
49. Jagannathan, G.; Pillaipakkamnatt, K.; Wright, R.N. A practical differentially private random decision tree classifier. In *Proceedings of the 2009 IEEE International Conference on Data Mining Workshops*, Miami, FL, USA, 6 December 2009; pp. 114–121.
50. Feng, X.; Li, S.; Yuan, C.; Zeng, P.; Sun, Y. Prediction of slope stability using naive Bayes classifier. *KSCE J. Civ. Eng.* **2018**, *22*, 941–950. [[CrossRef](#)]
51. Balakrishnama, S.; Ganapathiraju, A. Linear discriminant analysis—a brief tutorial. *Inst. Signal Inf. Process.* **1998**, *18*, 1–8.
52. Windeatt, T. Accuracy/diversity and ensemble MLP classifier design. *IEEE Trans. Neural Netw.* **2006**, *17*, 1194–1211. [[CrossRef](#)]
53. Ruta, D.; Gabrys, B. Classifier selection for majority voting. *Inf. Fusion* **2005**, *6*, 63–81. [[CrossRef](#)]
54. Pajankar, A. Introduction to Raspberry Pi. In *Practical Linux with Raspberry Pi OS*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 1–34.
55. Krishna, G.S.; Kiran, T.S.R.; Srisaila, A. Testing Performance of RaspberryPi as IDS Using SNORT. Available online: <https://www.sciencedirect.com/science/article/pii/S2214785321006994> (accessed on 15 November 2021).
56. Heenan, R.; Moradpoor, N. Introduction to security onion. In *Proceedings of the First Post Graduate Cyber Security Symposium*, Edinburgh, UK, 16 May 2016.
57. Wazuh Inc. Log Data Collection. Available online: <https://documentation.wazuh.com/current/user-manual/capabilities/log-data-collection/index.html> (accessed on 15 November 2021).
58. Lashkari, A.H. CICFlowMeter. Available online: <https://github.com/ahlashkari/CICFlowMeter> (accessed on 15 February 2022).
59. Anjana, K.; Urolagin, S. Churn Prediction in Telecom Industry Using Machine Learning Algorithms with K-Best and Principal Component Analysis. In *Proceedings of the Applications of Artificial Intelligence in Engineering*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 499–507.
60. Mao, K.Z. Orthogonal forward selection and backward elimination algorithms for feature subset selection. *IEEE Trans. Syst. Man, Cybern. Part B (Cybern.)* **2004**, *34*, 629–634. [[CrossRef](#)]
61. Hooker, S.; Erhan, D.; Kindermans, P.J.; Kim, B. Evaluating Feature Importance Estimates. Available online: <https://research.google/pubs/pub47088/> (accessed on 15 February 2022).
62. Ullah, I.; Mahmoud, Q.H. A scheme for generating a dataset for anomalous activity detection in iot networks. In *Proceedings of the Canadian Conference on Artificial Intelligence*, Ottawa, ON, Canada, 13–15 May 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 508–520.
63. AlMajed, H.; AlMogren, A. A secure and efficient ECC-based scheme for edge computing and internet of things. *Sensors* **2020**, *20*, 6158. [[CrossRef](#)]