network

MDPI

Article

AIS for Malware Detection in a Realistic IoT System: Challenges and Opportunities

Hadeel Alrubayyi, Gokop Goteng and Mona Jaber *

School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, UK; h.s.alrubayyi@qmul.ac.uk (H.A.); g.l.goteng@qmul.ac.uk (G.G.)
* Correspondence: m.jaber@qmul.ac.uk

Abstract: With the expansion of the digital world, the number of Internet of things (IoT) devices is evolving dramatically. IoT devices have limited computational power and a small memory. Consequently, existing and complex security methods are not suitable to detect unknown malware attacks in IoT networks. This has become a major concern in the advent of increasingly unpredictable and innovative cyberattacks. In this context, artificial immune systems (AISs) have emerged as an effective malware detection mechanism with low requirements for computation and memory. In this research, we first validate the malware detection results of a recent AIS solution using multiple datasets with different types of malware attacks. Next, we examine the potential gains and limitations of promising AIS solutions under realistic implementation scenarios. We design a realistic IoT framework mimicking real-life IoT system architectures. The objective is to evaluate the AIS solutions' performance with regard to the system constraints. We demonstrate that AIS solutions succeed in detecting unknown malware in the most challenging conditions. Furthermore, the systemic results with different system architectures reveal the AIS solutions' ability to transfer learning between IoT devices. Transfer learning is a pivotal feature in the presence of highly constrained devices in the network. More importantly, this work highlights that previously published AIS performance results, which were obtained in a simulation environment, cannot be taken at face value. In reality, AIS's malware detection accuracy for IoT systems is 91% in the most restricted designed system compared to the 99% accuracy rate reported in the simulation experiment.

Keywords: artificial immune systems (AIS); Amazon Web Services (AWS); cloud computing; Internet of things (IoT); malware detection

Citation: Alrubayyi, H.; Goteng, G; Jaber, M. AIS for Malware Detection in a Realistic IoT System: Challenges and Opportunities. *Network* **2023**, *3*, 522–537. https://doi.org/10.3390/network3040023

Academic Editor: Sidi Mohammed Senouci

Received: 1 June 2023
Revised: 20 October 2023
Accepted: 3 November 2023
Published: 16 November 2023

1. Introduction

The Internet of things (IoT) paradigm is continuously revolutionizing the world we live in through innovative IoT applications. As a result, the number of IoT devices has increased exponentially since 2002 when small security cameras were installed, and the market is expected to grow to USD 11 billion in 2026 [1]. IoT systems consist of lightweight devices connected to the internet giving a real-time interaction in advanced networks [2]. Lightweight devices indicate devices with a small battery, small memory, and constrained processing power, thus with limited options for complex security methods [3]. The proliferation of IoT devices and their spiral role in critical applications, such as e-health and smart cities, increases the risk of security attacks. In fact, in the first six months of 2021, there were almost 1.5 million attacks against IoT devices [4]. As detailed in [3], the interconnectivity of IoT devices and the exposure of often private data pose a major security challenge to the spread of IoT systems. Postquantum cryptosystems for the Internet of things is discussed in [5] with a focus on the emerging lightweight lattice-based cryptography (LW-LBC) as a promising solution to secure data transmission in view of its low-power footprint, narrow area, lightweight bandwidth requirements, and good performance. This work focuses on IoT malware attacks which are malicious files that are

installed on devices by hackers without the user's knowledge or permission. One of the major malware attacks was in March 2021 against Verkada [6]. Hackers managed to gain access to the company's security cameras invading the privacy of many major locations, such as hospitals and police stations.

In the past years, many effective methods have been proposed to detect such attacks. These methods can be categorized as either signature-based or behavior-based [3]. The signature-based technique consists in comparing incoming files to an existing list of known malware attacks. This method suffers from two drawbacks. Firstly, often times, malware files are encrypted which would then require extensive processing time to extract their signature. Secondly, signature-based techniques are limited to detecting known malware and fail to detect new attacks. In contrast, the behavioral-based technique can detect unseen malware since it is designed to analyze the behavior of the file rather than reading its signature. To this end, this technique first collects and interprets information about the incoming file and matches it with the known legitimate behavior of benign files. However, behavior-based techniques are complex and computationally expensive, and their effectiveness is negatively impacted with the increase in the number of files to be examined. Readers are encouraged to refer to [3] for a detailed survey of malware detection techniques in the IoT and other systems.

Signature-based methods are not sufficient to secure IoT devices from unknown malware attacks. Moreover, behavior-based methods are expensive to install in IoT devices. Indeed, online detection of malware-attacks on IoT devices is extremely difficult, which has led researchers to consider an alternate approach which consists in shrinking the attack surface in order to reduce the threat of the attack [7]. Such an approach would require IoT devices to undergo stringent security tests before being deployed. A promising emerging approach for malware detection in currently deployed IoT systems leverages artificial immune systems (AISs). AIS methods emulate the behavioral patterns of the adaptive immune system within the human body to identify and detect attacks [8]. AIS methods are lightweight, adaptive, and distributed, hence fitting the requirements of IoT malware detection. The advantages of using AISs to secure the IoT systems are discussed in [3,9], showing that the NPS is the most promising method compared to state-of-the-art solutions. However, these reported results are based on ideal simulation environments and do not reflect the constraints of realistic IoT systems in terms of memory and processing power. In this work, we examine the realistic gains and reveal hidden challenges of leading AIS methods by implementing these in actual IoT systems with differing characteristics. Our aim is to delve into the implementation implications of an AIS within a realistic IoT system, shedding light on its practical applications and challenges. While prior work validates IoT security methods at a device level using field-programmable gate array (FPGA) or a Raspberry Pi, such as [10,11], this article aims to validate the AIS methods at a system level with multiple devices. For this reason, we utilize Amazon Web Services (AWS) to simulate realistic IoT system scenarios.

Contribution and Paper Structure

State-of-the-art work in AISs such as [3,9] demonstrates the potential of AISs in IoT malware detection; however, no existing work validates this approach in a realistic implementation. In this work, we present the first study that examines the applicability of AIS methods in an IoT-like architecture under different constraints and datasets:

- We use multiple state-of-the-art datasets with different types of malware attacks in the IoT to run AIS solutions for malware detection. We benchmark the results against the state-of-the-art intrusion detection methods that use the same datasets.
- We propose an AWS-enabled validation framework for the evaluation of AIS malware detection solutions, under realistic architecture and characteristics.
- The proposed framework is used to evaluate the performance of two leading AIS solutions under constrained systems. The memory size is found to be the most limiting

factor that results in under-par performance compared to the reported simulation results for both solutions.

- We propose the first trial of transfer learning within IoT systems to combat the constrained memory in IoT devices. We demonstrate the transfer learning effectiveness of AIS solutions in securing the IoT.

In Section 2, we present relevant research concerning malware detection in IoT systems utilizing AIS methodologies. In Section 3, we present using multiple datasets to run AIS solutions for malware detection. In Section 4, we present the AIS solutions' implementation in a realistic setup. We describe the IoT systems' architectures and the problem formulation. In Section 5, we present the implementation results, performance analysis, and discussion. We finally conclude in Section 6.

2. State-of-the-Art AIS Solutions

AIS is a discipline influenced by the immune system of the human body. In the adaptive immune system of the human body, B-cells and T-cells work together as primary agents to recognize antigens [12]. Antigens are any foreign substance entering the body. Once an antigen is recognized, the immune system response is triggered to produce antibodies. Similarly, AIS methods use classifiers to detect any malicious files that are not part of the system [13,14]. AIS methods require less computational power and time; therefore, they are a good fit for malware detection in the IoT [3].

We present promising state-of-the-art malware detection solutions using AIS techniques. The first AIS algorithm we review is a combination of negative selection and neural networks methods (NSNN) for intrusion detection in the IoT [15]. The goal of designing NSNN is to meet the IoT devices' properties, mainly lightweight and distributive. The highest F1-score that NSNN achieve is 0.77 for denial-of-service attack detection. This approach is restricted to creating negative selection and employing a neural network. Also, only the simulation results of the methods are published, and there is no presented way to implement this method online.

The second AIS algorithm we review is a combination of negative and positive selection for intrusion detection in the IoT (MNSA) [16]. The findings indicate that the MNSA algorithm can identify up to 34% of intrusions without prior knowledge about nonself. Furthermore, the MNSA can validate over 90% of the identified files. However, a significant drawback of this approach is that the results were achieved using random strings rather than real malware files. Moreover, the memory needed to generate detectors as calculated in [3] was very high, which does not make it a good candidate to secure the IoT systems.

The third AIS algorithm we review is the NPS algorithm presented in [9]. The NPS uses a combination of negative detectors to recognize nonself data and positive detectors to recognize self-data. Also, the original negative selection algorithm generates 12-bit-size detectors, while the NPS generates 16-bit-size detectors. The authors prove the efficiency of combining both techniques and generating large-size detectors by presenting a high detection accuracy rate with minimal false negatives. The results presented are for a simulation of the NPS method and it was not run on a real-time platform.

Since the NPS is the recent promising solution for malware detection in the IoT, in this work we first ran the NPS using multiple datasets. The aim of this experiment was to validate the accuracy of the detection results of the method under different circumstances, such as different malware attack types and different file sizes. Second, we implemented the NPS using a real-time platform. We created different IoT system architectures to test the AIS solutions efficiency in detecting unknown malware attacks while minimizing the memory utilization in the IoT device. We also implemented the MNSA, which is the second promising AIS solution for malware detection, using the same scenarios. We conducted a quantitative analysis studying the detection behavior of AIS solutions in a realistic setup. In the next sections, we present the AIS solutions methodology, the simulation results using multiple datasets, the implementation including the IoT system architectures and problem formulation, and results and discussion.

Most of the presented AIS methods have inherent limitations when applied to IoT malware detection. For instance, NSNN employ a neural network that requires a high computation power whereas the MNSA consumes a high amount of memory to generate the required detectors. Given that IoT devices are often constrained with respect to both computation power and memory, the promising model performance may not be applicable in a realistic scenario. The NPS algorithm stands out as the best-performing AIS model with the least memory and computation power requirements. Nonetheless, the NPS algorithm has not yet been validated in a realistic IoT architecture with differing datasets. This work aims to bridge this gap and therefore offer a comprehensive study of AISs in IoT malware detection under realistic conditions.

The methodology of leading AIS solutions, NPS and MNSA, is similar and comprises two stages: the detectors generation stage and the detection stage (see Figure 1). In the detectors generation stage, two different sets of detectors are generated: the negative detectors, represented as C_{Neg}, which do not match self-data, and the positive detectors, represented as C_{Pos}, which match self-data. In the detection stage, if an incoming file matches one of the negative detectors, it is tagged as a malicious file. In contrast, if an incoming file matches one of the positive detectors, it is tagged as a benign file.

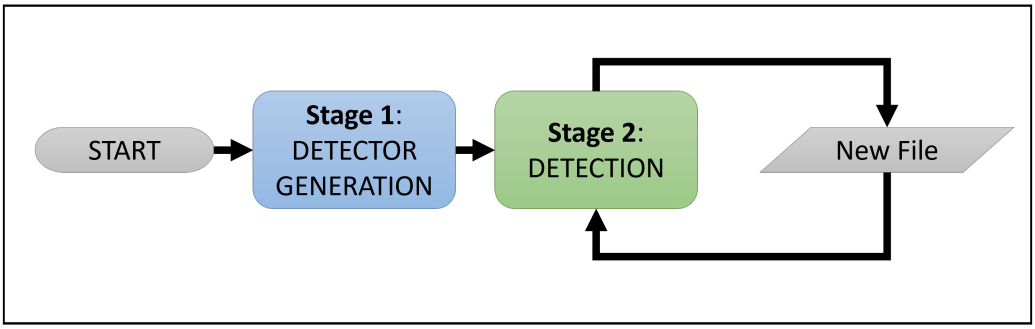


Figure 1. Stages for detector generation and detection in artificial immune systems. The new file represents incoming files which may be benign or malicious.

3. NPS Simulations and Evaluation

In this section, we first describe the datasets used in our simulations to evaluate the leading AIS solution, the NPS algorithm [9]. In addition to the widely used NSL-KDD dataset [17] used in [3,9], we present new NPS results with two recent datasets that were created by the intelligent security group at the University of New South Wales (UNSW) Canberra. The first dataset is the Bot-IoT dataset and the second one is the UNSW-NB15. Next, we benchmark the newly obtained results with existing works that employ the same datasets to highlight the systematic superiority of the NPS algorithm regardless of the dataset.

3.1. NSL-KDD Dataset

The NSL-KDD dataset is an extension of the KDD'99 dataset [18], aiming to eliminate redundant records present in the original KDD'99 dataset. This refinement led to a reduction in the number of borderline records, setting it apart from other datasets. It has gained widespread popularity for evaluating AIS methods, including NPS and NSNN methods. The dataset's traffic data were captured using 420 machines and 30 servers across 5 distinct departments. While not IoT-specific, the NSL-KDD dataset encompasses diverse malware attack types and provides a range of file features, making it well suited for experimentation in this context. See the details of the dataset in Table 1.

Table 1. NSL-KDD dataset.

	NSL-KDD Dataset Features
Total number of records	over 1,074,992
Attack Files	262,178
Benign files	812,814
Types of attacks	Brute-force, Heartbleed attack, botnet, denial of service, distributed denial of service, Web attacks, infiltration of the network from inside
Number of traffic features	80
Some of the traffic features	Destination port, flow duration, average size of packet, number of forward packets per second, number of backward packets per second

3.2. Bot-IoT Dataset

The Bot-IoT dataset was developed by simulating a realistic network environment that encompasses both regular and botnet-related network activity. This dataset encompasses various types of attacks such as DDoS, DoS, operating system (OS) and service scan, keylogging, and data exfiltration. (see Table 2). The Bot-IoT is introduced and explained by the authors in [19–23].

Table 2. Bot-IoT Dataset.

	Bot-IoT Dataset Features
Total number of records	over 73,000,000
Attack files	73,360,900
Benign files	9543
List of attacks	DDoS, DoS, operating system (OS) and service scan, keylogging and data exfiltration attacks
Number of traffic features	46
Some of the traffic features	Destination port, flow duration, average size of packet, number of forward packets per second, number of backward packets per second

3.3. UNSW-NB15 Dataset

The UNSW-NB15 dataset was generated using the IXIA PerfectStorm tool, blending actual contemporary normal activities with artificially created contemporary attack behaviors. This dataset encompasses nine attack categories: fuzzers, analysis, backdoors, dos, exploits, generic, reconnaissance, shellcode, and worms (refer to Table 3). The authors introduced and provided an explanation of the UNSW-NB15 dataset in [24–29].

Table 3. UNSW-NB15 Dataset.

UNSW-NB15 Dataset Features	
Total number of records	over 25,000,000
Attack files	321,283
Number of benign files	2,218,761
List of attacks	Fuzzers, analysis, backdoors, DoS, exploits, generic, reconnaissance, shellcode and worms
Number of traffic features	49
Some of the traffic features	Destination port, flow duration, average size of packet, number of forward packets per second, number of backward packets per second

3.4. Performance Analysis of NPS

In this section, we first summarize the performance of published malware detection algorithms using both UNWS datasets followed by the NPS simulations results. First, the authors in [30] presented the LSTM-based unsupervised deep learning model for malware detection in the IoT. The authors reported a 96% detection rate accuracy for running their model using both the Bot-IoT and UNSW-NB15 datasets combined. The main focus of this work is on detecting DoS and DDoS attacks. Second, the method presented in [31] achieved a 99% detection accuracy rate using the UNSW-NB15 dataset; however, the number of classes used for the classification was unclear. Next, the method presented in [32] achieved a 70% detection accuracy rate using the UNSW-NB15 dataset. The results were obtained using only 10% of the total number of records in the dataset. Similarly, the authors in [33] used only 10% of the total number of records in the UNSW-NB15 dataset and reported a 89% detection rate accuracy. Finally, the feed-forward ANN (FNN) and self-normalized neural network (SNN) presented in [34] used the Bot-IoT for their experiment. They reported a 95% detection accuracy rate for the FNN and 91% for the SNN. They only used 20% of the total number of records and 10 best features to run the proposed solutions.

We ran the NPS algorithm with each of the UNSW datasets and reproduced the results with the NSL-KDD dataset under two scenarios: 20 detectors (see Figure 2) and 30 detectors (see Figure 3). Details of the implementation and a description of detectors are available in [9]. The detection accuracy rate was calculated as in Equation (1) presented in [9] where:

- True positive (TP): malware is detected as a malicious application;
- True negative (TN): a benign application is detected as a nonmalicious application;
- False positive (FP): a benign application is detected as a malicious application;
- False negative (FN): malware is detected as a nonmalicious application.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

In addition, we calculated the precision (Equation (2)), recall (Equation (3)), and F1-score (Equation (4)) (as presented in [9]) to underline the performance of the algorithm regardless of the imbalance ratio in each dataset.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1\text{-}Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

The figure below (Figure 2) shows the performance analysis of running the NPS algorithm using the NSL-KDD, Bot-IoT, and UNSW-NB15 datasets with 20 detectors in each set, C_{Pos} and C_{Neg}. The NPS algorithm achieves high detection performance using all three datasets. The detection accuracy is up to 92%, the precision rate is up to 95%, and the recall rate is up to 97%. The results of creating 30 detectors in each set, negative and positive, using multiple datasets are shown in Figure 3. The detection accuracy goes up to 99%, the precision rate is 99%, and the recall rate is close to 100%.

Figures 2 and 3 show a negligible gap when running the NPS using the three datasets. This difference is justifiable by the different number of records and types of attacks in each of the datasets. Therefore, this performance analysis shows the effectiveness of the NPS algorithm in detecting unknown malware attacks in IoT systems.

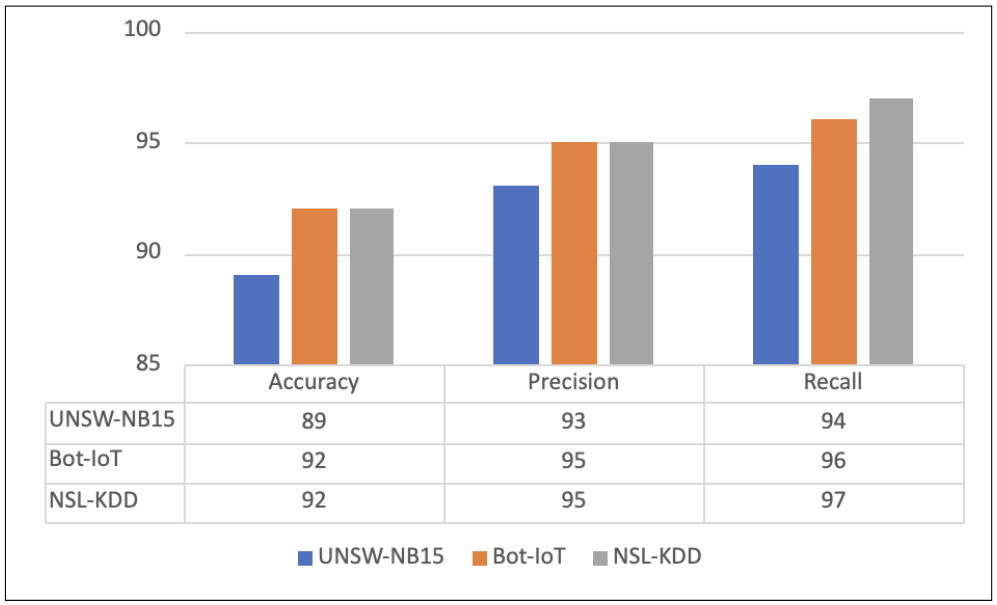


Figure 2. Running the NPS algorithm using multiple datasets—20 detectors results.

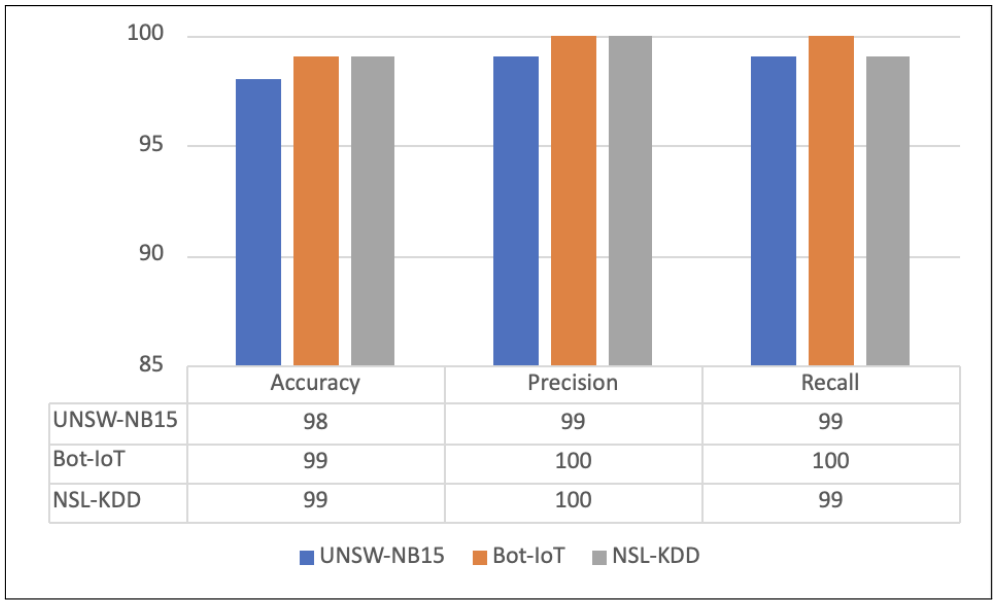


Figure 3. Running the NPS algorithm using multiple datasets—30 detectors results.

As anticipated, the NPS algorithm outperforms the state-of-the-art solutions listed here, achieving a higher detection accuracy rate by up to 42% whilst requiring less computation

power than those based on neural networks [30,31,34]. This validates the claim that the NPS algorithm is fit to secure the IoT from unknown malware attacks.

Despite the encouraging results obtained with the NPS algorithm across the three different datasets, it still remains to be validated in a realistic IoT system which is the target of the next section.

4. AIS Solutions' Implementation

This work investigates the relationship between, on one hand, the achievable performance of AIS malware detection algorithms and on the other hand, the hardware and system architecture limitations. First, we present the IoT system architectures in Section 4.1. Then, we formulate the malware detection problem as a function using these factors and the related labeled dataset in Section 4.2. We identify the hardware and software factors and their realistic range in Section 4.3.

We implemented the NPS algorithm and the MNSA, the leading state-of-the-art AIS solutions proposed in [9,16], using a real-time platform. This allowed us to conduct a quantitative performance analysis and study the behavior of the AIS solutions in realistic setups.

4.1. IoT System Architecture

In this work, we used AWS to create the desired architecture for the implementation. We introduce the main services used in this section and a brief description of each service [35].

- Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the AWS Cloud. Users have total control over the EC2 configuration as it is not an AWS managed service.
- Virtual Private Cloud (Amazon VPC) creates a logically isolated virtual network when launching AWS resources. This service provides an extra layer of security to the implementation by using public and private subnets, and network access control list configuration.
- AWS IoT Core enables IoT devices connected to the AWS cloud. It supports a large number of devices and messages by providing reliable and secure services.
- Device Gateway: the entry point for IoT devices connected to AWS
- AWS Lambda runs programming code in a serverless computing service in response to events and automatically manages the underlying computing resources.
- Cloudwatch: monitoring and management services for AWS resources.

In this implementation, since we had total control over the configuration of the EC2, we used it to configure the IoT device to be connected to the network. We created two system architectures (see Figures 4 and 5) using AWS to run this experiment to mirror real system scenarios and conduct a performance analysis.

4.2. Problem Formulation

In this implementation, we used the CSE-CIC-IDS2018 dataset (available in [17]). We used this particular dataset for mainly two reasons. First is the fact that it was used in the simulation experiment for both methods, NPS and MNSA. The other reason for using this dataset is that AWS computing platforms were used to collect the traffic data for this dataset. Consequently, to obtain more coherent and accurate results, we ran the implementation on AWS using the IDS2018 dataset. This dataset contains seven different malware attacks: brute-force attack, Heartbleed attack, botnet, denial-of-service attack, distributed denial-of-service attack, web attacks, and infiltration of the network from inside [3].

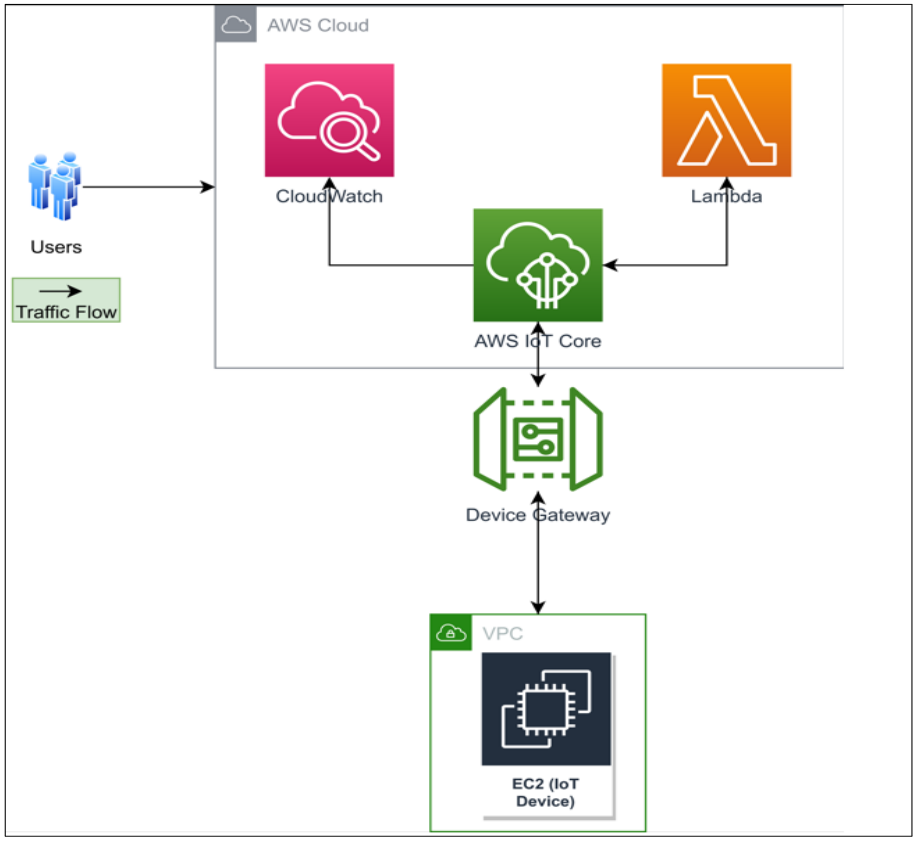


Figure 4. IoT system architecture implementation using AWS-1-.

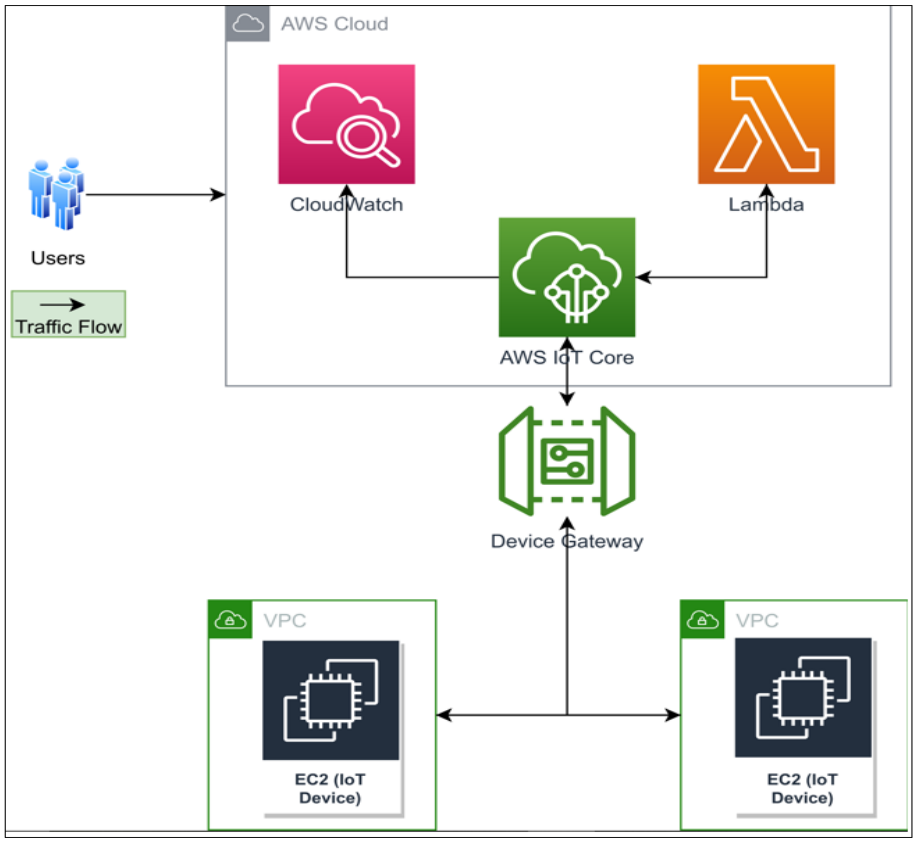


Figure 5. IoT system architecture implementation using AWS-2-.

We represented the total number of dataset records as D_{Total}, and we represented each record in D_{Total} as D_i, where: i = [1 to $|D_{Total}|$].

The size of D_i was represented as Z, and each D_i included the same number of features represented as F_Z, where:

$$F = [f_1, f_2, f_3, \ldots F_Z] \tag{5}$$

Each record D_i was associated with the ground truth Y_i where $Y_i = 0, 1$. $Y_i = 0$ indicates that the record is benign and $Y_i = 1$ indicates that it is malicious. We defined Y as the vector including all the labels of Y_i for: i = [1 to $|D_{Total}|$].

The dataset D_{Total} was split into two parts, D_{Train} and D_{Test}, where:

$$D_{Total} = D_{Train} \cup D_{Test} \tag{6}$$

Different from [3,9], this work examines the realistic implementation of AIS solutions in constrained IoT systems. The objective of malware detection mechanisms is surely to maximize the detection rate of unknown malware while reducing false alarms (when benign files are wrongly classified as malicious). Thus, we maximized the number of correct predictions ($\hat{Y}$) by the AIS solutions (see Equation (7)). To this end, we studied how the different parameters of the AIS algorithms could be tuned to accommodate the given constrained conditions of the IoT systems, while we still achieved a high detection performance of unknown malware. In particular, we found the suitable number of positive and negative detectors, C_{Pos} and C_{Neg}, the size of each detector L, and the possible size of D_{Train} (indicated as $|D_{Train}|$), that would allow the highest number of correct predictions. We defined the range of the minimum and the maximum number of detectors as ND, and the range of the minimum and the maximum number of D_{Train} as NT. Equation (7) presents the problem formulation and the optimization constraint defined by the total memory TM.

$$\max_{|D_{Train}|, C_{Pos}, C_{Neg}, L} \sum_{k=1}^{|D_{Test}|} \hat{Y}_k == Y_k \tag{7}$$

s.t.

$$ND((C_{pos} + C_{neg}) \times L) + NT(D_{Train} \times Z) <= TM \tag{8}$$

4.3. Implementation

The system configuration and the memory size range were inspired by the Internet of medical things (IoMT) devices presented in [36] for heart monitoring. First, the volume size represented the memory size of the IoT device used in each system. The memory size ranged between 30 GB and 128 GB. Since IoT devices are lightweight with a small memory and computation capacity, we only increased the volume size to 128 GB to fit the IoT devices' requirements and mimic real-life scenarios. Second, the size of D_{Train} was 25% of D_{Total} and the size of D_{Test} was 75% of D_{Total}. The size of D_{Total} varied depending on the experiment undertaken (see Table 4). This was decided based on the memory capacity, method performance, and CPU utilization. In all five systems, we used only one memory and one CPU in the IoT device. We set the memory performance from moderate to low, and we used only the TCP protocol. This setup mimicked the lightweight, low-memory-capacity IoT devices often connected to the network.

We explain the setup and the variable values for each system. We set the values for the following variables in Table 4: the volume size, D_{Total}, and the number of devices to be connected to the network.

Table 4. IoT system specifications for all systems.

	System 1	**System 2**	**System 3**	**System 4**	**System 5**
Volume size	30 GB	32 GB	64 GB	128 GB	30 GB
$\mid D_{Total} \mid$	12,000	14,000	28,000	400,000	24,000
Number of IoT devices	One	One	One	One	Two

The table below (Table 5) shows the size and the number of detectors used in this implementation for the NPS algorithm and MNSA.

Table 5. Detectors' size used for the NPS and MNSA.

	NPS	**MNSA**
$\mid C_{Pos} \mid$	30	20
$\mid C_{Neg} \mid$	30	150
$\mid L \mid$	16	12

We used the system architecture shown in Figure 4 for the first four systems where we connected only one IoT device. In the fifth setup of this implementation (System 5), we connected two IoT devices to the IoT core as shown in Figure 5. As demonstrated in the figure, The traffic went both ways between the two IoT devices. The algorithm was implemented and trained on one IoT device, then it was tested using the traffic coming from IoT Core and the other IoT device. Since the load was divided between two devices in this setup, we used the total number of D_{Total} in one of the IoT devices in the detector generation stage of the method. Then, we used the total number of D_{Total} in the other IoT device in the detection stage, meaning D_{Train} and D_{Test} both had a size of 12,000 records.

5. Results and Discussion

In this section, we present the results of our implementation followed by a discussion and interpretation. We calculated the detection accuracy, precision, recall, and F1-score using the equations presented in [9]. An important characteristic in the context of malware detection is to reduce false alarms. This metric is referred to as detection recall. An increase in false alarms may slow down the data acquisition process and may affect the acceptance of a malware detection algorithm. Also, it is critical that any algorithm successfully identifies all malicious files as malware. This metric is referred to as detection precision. To this end, the F1-score of the proposed methods was measured as this captures the accuracy of detecting malware and the rate of false alarms jointly. The detection performance results are presented in Section 5.1. Then, we present the CPU Utilization for the NPS algorithm and MNSA in Section 5.2. Finally, we present the implementation and simulation performance analysis in Section 5.3.

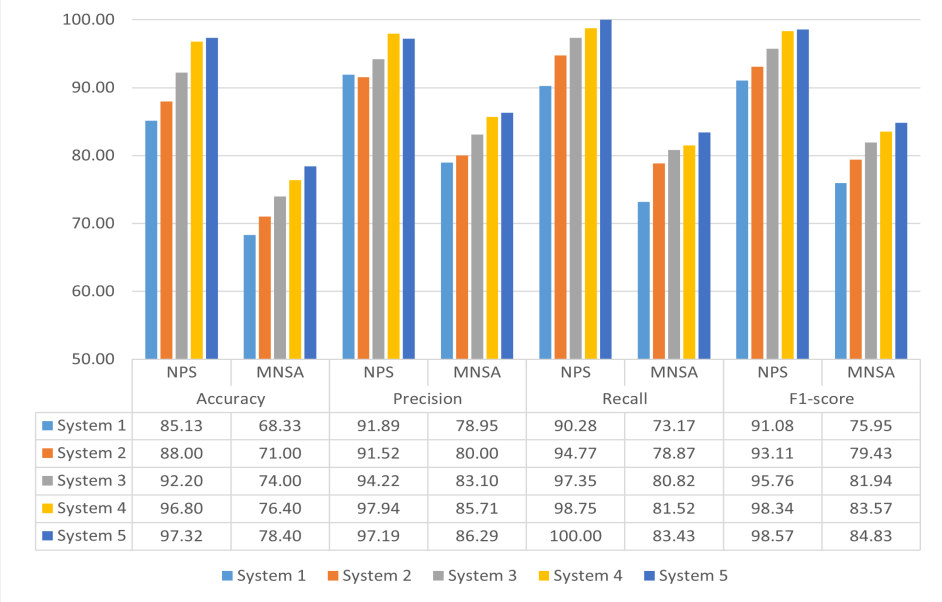
5.1. Detection Performance

The figure below (Figure 6) shows the results for the five system scenarios implemented in this project for both the NPS algorithm and MNSA. Since the F1-score takes both negative and positive detection into account, we used it as the main metric to evaluate the performance in this analysis. First, we started with systems 1 to 4 where we used only one IoT device in the implementation. As predicted in [3,9], the NPS algorithm succeeds in detecting malware better than the MNSA in all four systems, as evidenced by the higher F1-score by up to 20% than that of the MNSA. As anticipated, the performance of both the NPS algorithm and MNSA improved when we moved from system 1 to 4 by 8% and 10% for the NPS algorithm and MNSA, respectively. Increasing the volume size allowed for an increase in the number of D_{Train}. Therefore, this resulted in better detection performance in all four systems for both methods.

In system 5, we connected two IoT devices to the system. That system architecture was used to evaluate the method's transfer learning abilities. Since we trained the method

on one IoT device, we could use the total number of D_{Total} to create the detectors. Both AIS solutions, the NPS algorithm and MNSA, showed the capability of transferring learning and protecting both IoT devices connected to the system. However, as anticipated, the NPS algorithm succeeded in a better detection performance when protecting the two IoT devices than the MNSA by 16%. This shows that the NPS algorithm has better transfer learning abilities than the MNSA. Therefore, the NPS algorithm is better at protecting distributed and robust IoT systems.

We demonstrated the NPS and MNSA detection performance and lightweight abilities on one IoT device using systems 1 to 4. System 5 demonstrated the transfer learning abilities. In this implementation, by increasing the volume size of the device, we could use a larger number of dataset records in the detectors' generation stage. This resulted in a better learning curve and thus a better classification accuracy when detecting malware files. Furthermore, AIS solutions can secure the IoT system with multiple IoT devices if only installed on one IoT device as demonstrated in system 5.



	Accuracy		Precision		Recall		F1-score	
	NPS	MNSA	NPS	MNSA	NPS	MNSA	NPS	MNSA
System 1	85.13	68.33	91.89	78.95	90.28	73.17	91.08	75.95
System 2	88.00	71.00	91.52	80.00	94.77	78.87	93.11	79.43
System 3	92.20	74.00	94.22	83.10	97.35	80.82	95.76	81.94
System 4	96.80	76.40	97.94	85.71	98.75	81.52	98.34	83.57
System 5	97.32	78.40	97.19	86.29	100.00	83.43	98.57	84.83

Figure 6. Malware detection performance results for the NPS algorithm and MNSA using AWS.

5.2. CPU Utilization

One of the AWS services we used in this implementation was Cloudwatch, which is for monitoring and managing services. Cloudwatch shows the CPU average utilization for AWS resources, the implemented IoT devices in this case. The objective was to validate the claims in [3,9] that AIS solutions are lightweight in realistic settings. To this end, the CPU utilization was measured in each system for both the NPS algorithm and MNSA as shown in Figure 7. The NPS algorithm required less CPU utilization by up to 36.6% than the MNSA, hence, it is more suitable for lightweight IoT systems under all conditions depicted by all five systems. The results also showed that the CPU utilization dropped systematically when moving from system 1 to system 5 for both NPS algorithm and MNSA. We saw a decrease of 12.7% and 10.3% in CPU utilization for the NPS algorithm and MNSA, respectively. By increasing the volume size of the IoT device, we can decrease the CPU utilization in the device, which is one of the main objectives when implementing security methods in IoT devices.

Figure 7. CPU average utilization for IoT devices used in the implementation of the NPS algorithm and MNSAS.

5.3. Algorithm Simulation and Implementation Performance Analysis

In this section, we compare the actual implementation results shown in Figure 6 and the ideal simulation results in [3,9]. In this work, we used the F1-score as the main factor to compare the implementation and the simulation results of the AIS solutions. In this implementation, the size of $C_{Pos+Neg}$ when we ran the NPS algorithm was 60 detectors, and it was 170 detectors for the MNSA. Consequently, to present a valid quantitative analysis, we compared the results for the same number of detectors in the simulation. Moreover, since the simulation was run as one entity, we analyzed the performance of the first four system architectures, where we ran the implementation using one IoT device. Figure 8 shows the analysis of the results for both the ideal simulation and the actual implementation results for the NPS algorithm and MNSA.

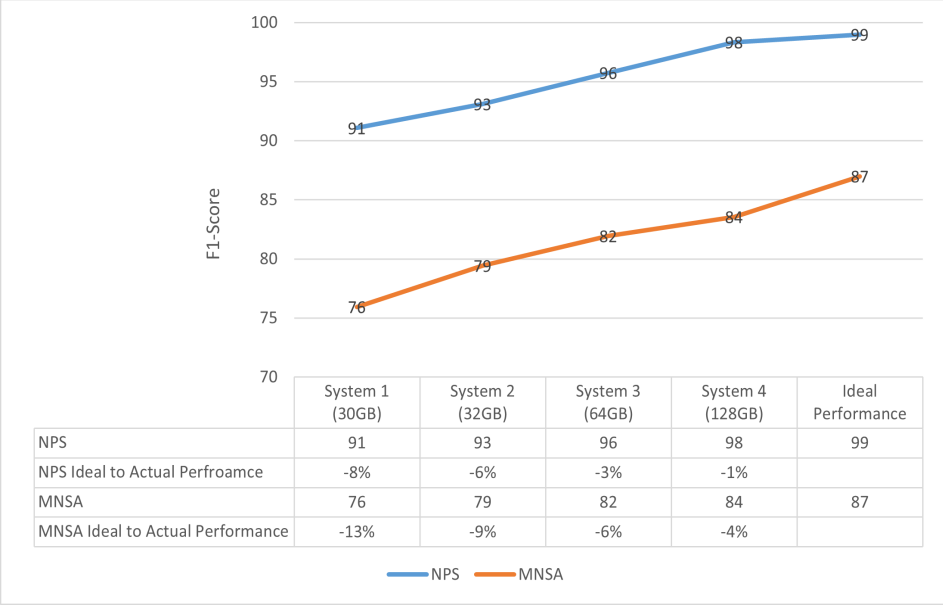


Figure 8. Comparing the actual to the ideal performance of AISs for different IoT systems.

We compared the ideal simulation results with system 1's results since it used one IoT device with the most constrained memory size. The MNSA showed a decrease of almost 12.7% from the simulation results and system 1's results. While the NPS algorithm's performance decreased by 8% when compared to the simulation results. As anticipated, both AIS solutions, the NPS algorithm and the MNSA underperformed compared to their ideal simulation results. This validates the claim that IoT security solutions should be tested in a realistic environment to demonstrate sufficient findings.

6. Conclusions

The growth of IoT devices forms a huge security threat to data privacy, increasing the number of malware attacks. Thus, developing an algorithm that meets the IoT devices' requirements is crucial. AIS methods are not expensive to implement, which makes them a good fit to secure the IoT. Even though the AIS solutions are promising in this context, the published results are based on computer simulations only. In this paper, we first validate the malware detection results of AIS solutions using multiple datasets with different types of malware attacks. Next, we presented the first implementation of AIS solutions using a real-time IoT platform. We used AWS to create different IoT system architectures mirroring real-life scenarios. We demonstrated the lightweight, transfer-learning, and detection capabilities of the AIS methods. The results showed that increasing the size of the IoT device's memory allowed us to increase the size of the dataset to train the module, which led to better detection performance. The results also showed that running an AIS solution on one of the IoT devices could secure the device itself from malware attacks and any other IoT devices connected to the same network. Finally, we validated the claim that AIS security solutions should be tested in a real setup to obtain accurate results.

In future work, we will connect more IoT devices to the network to further investigate the AIS solutions' robustness and fault tolerance. Also, using different datasets with different malware attack files might improve the learning curve, which might lead to better classification results.

Author Contributions: Conceptualization, H.A., G.G. and M.J.; methodology, H.A.; software, H.A.; validation, H.A., G.G. and M.J.; formal analysis, H.A., G.G. and M.J.; investigation, H.A.; writing—original draft preparation, H.A.; writing—review and editing, H.A., G.G. and M.J.; visualization, H.A.; supervision, G.G. and M.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: We used the NSL-KDD dataset to run the performance analysis in this research. The dataset is available at https://www.unb.ca/cic/datasets/nsl.html, accessed on 25 March 2023.

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

Abbreviations

The following abbreviations are used in this manuscript:

AIS	Artificial immune systems
AWS	Amazon Web Services
IoT	Internet of things
NSNN	Neural networks methods
OS	Operating system
IoMT	Internet of medical things

References

1. Wang, J.; Lim, M.K.; Wang, C.; Tseng, M.L. The evolution of the Internet of Things IoT over the past 20 years. *Comput. Ind. Eng.* **2021**, *155*, 107174. [CrossRef]
2. Mazlumi, S.H.H.; Kermani, M.A.M. Investigation the structure of the Internet of things (IoT) patent network using social network analysis. *IEEE Internet Things J.* **2022 9**, 13458–13469.

3. Alrubayyi, H.; Goteng, G.; Jaber, M.; Kelly, J. Challenges of Malware Detection in the IoT and a Review of Artificial Immune System Approaches. *J. Sens. Actuator Netw.* **2021**, *10*, 61. [CrossRef]
4. The Most Vulnerable IOT Devices: Think before You Buy. Available online: https://tdwi.org/articles/2021/11/05/most-vulnerable-iot-devices.aspx (accessed on 1 June 2023).
5. Asif, R. Post-Quantum Cryptosystems for Internet-of-Things: A Survey on Lattice-Based Algorithms. *IoT* **2021**, *2*, 71–91. [CrossRef]
6. Summary: 9 March 9 2021 Security Incident Report. Available online: https://www.verkada.com/uk/security-update/report/ (accessed on 1 June 2023).
7. Keerthi, K.; Roy, I.; Hazra, A.; Rebeiro, C., Formal Verification for Security in IoT Devices. In *Security and Fault Tolerance in Internet of Things*; Chakraborty, R.S., Mathew, J., Vasilakos, A.V., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 179–200. [CrossRef]
8. Jim, L.E.; Islam, N.; Gregory, M.A. Enhanced MANET security using artificial immune system based danger theory to detect selfish nodes. *Comput. Secur.* **2022**, *113*, 102538. [CrossRef]
9. Alrubayyi, H.; Goteng, G.; Jaber, M.; Kelly, J. A novel negative and positive selection algorithm to detect unknown malware in the IoT. In Proceedings of the IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Virtual, 10–13 May 2021; pp. 1–6.
10. Iqbal, M.; Ari Laksmono, A.M.; Prihatno, A.T.; Pratama, D.; Jeong, B.; Kim, H. Enhancing IoT Security: Integrating MQTT with ARIA Cipher 256 Algorithm Cryptography and mbedTLS. In Proceedings of the 2023 International Conference on Platform Technology and Service (PlatCon), Busan, Republic of Korea, 16–18 August 2023; pp. 91–96. [CrossRef]
11. Liu, C.; Zhang, Y.; Xu, J.; Zhao, J.; Xiang, S. Ensuring the Security and Performance of IoT Communication by Improving Encryption and Decryption With the Lightweight Cipher uBlock. *IEEE Syst. J.* **2022**, *16*, 5489–5500. [CrossRef]
12. Netea, M.G.; Schlitzer, A.; Placek, K.; Joosten, L.A.; Schultze, J.L. Innate and Adaptive Immune Memory: an Evolutionary Continuum in the Hostâ€™s Response to Pathogens. *Cell Host Microbe* **2019**, *25*, 13–26. [CrossRef] [PubMed]
13. Pump, R.; Ahlers, V.; Koschel, A. Evaluating Artificial Immune System Algorithms for Intrusion Detection. In Proceedings of the 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), London, UK, 27–28 July 2020; pp. 92–97. [CrossRef]
14. Abdullahi, M.; Baashar, Y.; Alhussian, H.; Alwadain, A.; Aziz, N.; Capretz, L.F.; Abdulkadir, S.J. Detecting Cybersecurity Attacks in Internet of Things Using Artificial Intelligence Methods: A Systematic Literature Review. *Electronics* **2022**, *11*, 198. [CrossRef]
15. Pamukov, M.E.; Poulkov, V.K.; Shterev, V.A. Negative Selection and Neural Network Based Algorithm for Intrusion Detection in IoT. In Proceedings of the 2018 41st International Conference on Telecommunications and Signal Processing (TSP), Athens, Greece, 4–6 July 2018; pp. 1–5. [CrossRef]
16. Pamukov, M.; Poulkov, V. Multiple negative selection algorithm: Improving detection error rates in IoT intrusion detection systems. In Proceedings of the 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Bucharest, Romania, 21–23 September 2017; Volume 1, pp. 543–547. [CrossRef]
17. NSL-KDD Dataset. Available online: https://www.unb.ca/cic/datasets/nsl.html (accessed on 1 June 2023).
18. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
19. The bot-IoT dataset. Available online: https://research.unsw.edu.au/projects/bot-iot-dataset (accessed on 1 June 2023).
20. 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]
21. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Slay, J. Towards Developing Network Forensic Mechanism for Botnet Activities in the IoT Based on Machine Learning Techniques. In Proceedings of the Mobile Networks and Management, Melbourne, Australia, 13–15 December 2017; pp. 30–44.
22. 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]
23. Koroniotis, N.; Moustafa, N.; Schiliro, F.; Gauravaram, P.; Janicke, H. A Holistic Review of Cybersecurity and Reliability Perspectives in Smart Airports. *IEEE Access* **2020**, *8*, 209802–209834. [CrossRef]
24. The UNSW-NB15 Dataset.Available online: https://research.unsw.edu.au/projects/unsw-nb15-dataset (accessed on 1 June 2023).
25. Moustafa, N.; Slay, J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6. [CrossRef]
26. Moustafa, N.; Slay, J. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf. Secur. Journal: Glob. Perspect.* **2016**, *25*, 18–31. [CrossRef]
27. Moustafa, N.; Slay, J.; Creech, G. Novel Geometric Area Analysis Technique for Anomaly Detection Using Trapezoidal Area Estimation on Large-Scale Networks. *IEEE Trans. Big Data* **2019**, *5*, 481–494. [CrossRef]

28. Moustafa, N.; Creech, G.; Slay, J., Big Data Analytics for Intrusion Detection System: Statistical Decision-Making Using Finite Dirichlet Mixture Models. In *Data Analytics and Decision Support for Cybersecurity: Trends, Methodologies and Applications*; Palomares Carrascosa, I., Kalutarage, H.K., Huang, Y., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 127–156. [CrossRef]
29. Sarhan, M.; Layeghy, S.; Moustafa, N.; Portmann, M. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. In *Big Data Technologies and Applications;* Deze, Z., Huang, H., Hou, R., Rho, S., Chilamkurti, N., Eds.; Springer: Cham, Switzerland, 2021; pp. 117–135.
30. Zeeshan, M.; Riaz, Q.; Bilal, M.A.; Shahzad, M.K.; Jabeen, H.; Haider, S.A.; Rahim, A. Protocol-Based Deep Intrusion Detection for DoS and DDoS Attacks Using UNSW-NB15 and Bot-IoT Data-Sets. *IEEE Access* **2022**, *10*, 2269–2283. [CrossRef]
31. Larriva-Novo, X.; VillagrÃ¡, V.A.; Vega-Barbas, M.; Rivera, D.; Sanz Rodrigo, M. An IoT-Focused Intrusion Detection System Approach Based on Preprocessing Characterization for Cybersecurity Datasets. *Sensors* **2021**, *21*, 656. [CrossRef] [PubMed]
32. Guizani, N.; Ghafoor, A. A Network Function Virtualization System for Detecting Malware in Large IoT Based Networks. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 1218–1228. [CrossRef]
33. Yang, Y.; Zheng, K.; Wu, C.; Yang, Y. Improving the Classification Effectiveness of Intrusion Detection by Using Improved Conditional Variational AutoEncoder and Deep Neural Network. *Sensors* **2019**, *19*, 2528. [CrossRef] [PubMed]
34. Ibitoye, O.; Shafiq, M.O.; Matrawy, A. Analyzing Adversarial Attacks against Deep Learning for Intrusion Detection in IoT Networks. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA 9–13 December 2019; pp. 1–6.
35. Perlitz, L.; Elliott, S.G. The Products. 2000. Available online: https://aws.amazon.com/products/ (accessed on 1 June 2023).
36. 8 of the Best Heart Rate Monitors of 2023. 2023. Available online: https://www.healthline.com/health/fitness/heart-rate-monitor (accessed on 1 June 2023).