Anomaly Based Unknown Intrusion Detection in Endpoint Environments

Sujeong Kim, Chanwoong Hwang and Taejin Lee *

Department of Information Security, Hoseo University, Asan 31499, Korea; sjkim395@gmail.com (S.K.); hcw85123@gmail.com (C.H.)
* Correspondence: kinjece@hoseo.edu

Received: 27 April 2020; Accepted: 16 June 2020; Published: 20 June 2020

Abstract: According to a study by Cybersecurity Ventures, cybercrime is expected to cost $6 trillion annually by 2021. Most cybersecurity threats access internal networks through infected endpoints. Recently, various endpoint environments such as smartphones, tablets, and Internet of things (IoT) devices have been configured, and security issues caused by malware targeting them are intensifying. Event logs-based detection technology for endpoint security is detected using rules or patterns. Therefore, known attacks can respond, but unknown attacks can be difficult to respond to immediately. To solve this problem, in this paper, local outlier factor (LOF) and Autoencoder detect suspicious behavior that deviates from normal behavior. It also detects threats and shows the corresponding threats when suspicious events corresponding to the rules created through the attack profile are constantly occurring. Experimental results detected eight new suspicious processes that were not previously detected, and four malicious processes and one suspicious process were judged using Hybrid Analysis and VirusTotal. Based on the experiment results, it is expected that the use of operational policies such as allowlists in the proposed model will significantly improve performance by minimizing false positives.

Keywords: anomaly detection; endpoint security; local outlier factor; AutoEncoder; anomaly score; attack profile

1. Introduction

Security threats continue to increase as the connection between individuals, businesses and countries is strengthened due to the development of technologies such as 5G, Internet of things (IoT), and artificial intelligence (AI). According to the Statist Research Department, the number of devices connected to the IoT is expected to reach 75 billion by 2025 [1]. In addition to IoT, various devices such as smartphones and PCs make up the endpoint environments. However, most cyber threats come from the endpoint. Cyber attackers infect malware on endpoints in order to gain access to a country’s or corporate’s internal systems. Malware is gradually developing in an intelligent form, such as advanced persistent threat (APT) attacks, and attacks using new and unknown malware are also increasing [2]. In recent years, fileless attacks that do not create a file have been increasing, and security has become a necessity [3]. In the case of fileless malware, a malicious dynamic link library (DLL) is injected into a normal process or a malicious script is executed. For example, it works by inserting a malicious VBScript into a Microsoft Office document like a normal word. As such, it runs on the system or changes the registry to threaten the endpoint. Since fileless attacks do not create files, there are disadvantages that cannot be detected with existing signature or rule-based security solutions. In addition, it detects all known malicious behavior, but attacks such as APT are difficult to detect because they attack the system continuously, not one-time. Therefore, since new attacks are constantly appearing in addition to the known attacks, it is necessary to respond to the unknown attacks.
We propose an anomaly detection approach to detect unknown intrusions in the endpoint environment. Anomaly detection techniques have not been suddenly developed in recent years, they have been studied for a long time. However, with big data, they have been getting attention again relatively recently. Anomaly detection is an important data analysis task that detects anomalous or abnormal data from a given dataset. The term is classified as a meaning of the viewpoint of data analysis, and academically, it is viewed as part of data mining. Anomaly detection is an important tool to detect abnormalities in many different domains including financial fraud detection, computer network intrusion, human behavioral analysis, gene expression analysis and many more. In recent years, smartphones can even do financial work, so the interest in anomaly detection research on fraud detection has emerged in the financial sector. We use anomaly detection techniques using event logs at the system level to detect unknown attacks at the endpoint. Log-based analysis is widely used to detect anomalies in a variety of environments [4–7]. Most of the existing log-based detection techniques work by using network logs to detect and update the database [8]. In recent years, effective research has been conducted on log analysis based on machine learning or large-scale logs, but it is insufficient [9–13]. Ahmed et al. [14] explained the research challenge of not having a publicly labeled data set available despite the many techniques available for anomaly detection. However, anomaly detection techniques make it easier for endpoints to recognize previously unseen attack behaviors, reducing the time and manpower cost of dealing with new attacks. Therefore, using anomaly detection and attack profile rules in endpoint logs will be more efficient and secure.

In this paper, two models are used to detect suspicious behavior that deviates from normal behavior. Anomaly detection analyzes the event log to generate anomaly scores. Based on the anomaly score, it detects threats corresponding to the rules created by the attack profile. Attack profiles identify the type of attack based on the frequency of event occurrence and show threats based on rules generated based on the association of suspicious events. The major contributions of the proposed work are summarized as follows:

- We design the model using local outlier factor (LOF) and Autoencoder for efficient anomaly detection. In addition, we propose analysis of attack profile for detected anomalies. It shows threats to anomalies based on various attack scenarios.
- Existing studies detect supervised learning-based attack behavior by using labeled data such as Normal, denial of service attack (DOS), remote to local attack (R2L), user to root attack (U2R), and Probe in network traffic. However, this study learns normal behavior based on unsupervised learning that is not labeled and detects deviations from it as suspicious behavior.
- Due to the increasing number of managed devices and the occurrence of numerous networks and event logs, real-time detection is limited by existing security methods. The proposed model is capable of large-scale processing according to the operation policy, and detects the user’s behavior-based suspicious behavior in real time in the endpoint log and shows the corresponding threat.
- It applies the allowlist operation policy and reduces the burden of security administrators by reducing the analysis target. In addition, it is efficient because it can set the learning period required for the operation policy.

Section 2 describes related studies on log analysis and anomaly detection. Section 3 proposes a model for anomaly-based unknown intrusion detection at the endpoint. Section 4 provides results for unknown intrusion detection using the proposed model. Section 5 discusses operational policies for improving the model using the results from Section 4. Finally, Section 6 has a conclusion.

2. Related Work

There are various studies to detect abnormal symptoms of data [15]. Anomaly-based intrusion detection system (IDS) detects intrusions that differ significantly from normal behavior. Jabez et al. [16] introduces a new approach called outlier detection in IDS. Outlier data is calculated by neighborhood
outlier factor (NOF) and consists of a big-dataset with a distributed storage environment. Figure 1 shows the results of comparing the proposed NOF outlier detection and execution time for the existing approach.

Shadi Aljawarneha et al. [17] proposed a model for anomaly-based intrusion detection in IDS. For the detection of an anomaly, this paper created a hybrid model using decision tree, neural network, and nearest neighbor method. The dataset used the KDD-99 dataset used for The Third International Knowledge Discovery and Data Mining Tool Competition [18]. The KDD-99 dataset was used for preprocessing the features required for model training, and only the features with a certain value were selected by calculating the information gain using the difference in entropy. This suggests that the proposed model can improve the accuracy and reduce the detection time. Table 1 compares the accuracy for the four classifiers. Although the proposed model has no difference between true positive (TP) rare and false positive (FP) rate, it achieved the highest accuracy.

Table 1. Comparison of four classifiers.

| Classifier      | TP    | FP    | Correctly Classified Instance | Incorrectly Classified Instance |
|-----------------|-------|-------|-------------------------------|--------------------------------|
| Naive Bayes     | 0.903 | 0.102 | 90.2876                       | 9.7124                         |
| J48             | 0.997 | 0.003 | 99.74                         | 0.26                           |
| Random Tree     | 0.997 | 0.003 | 99.74                         | 0.25                           |
| Proposed Model  | 0.997 | 0.003 | 99.81                         | 0.25                           |

In addition, studies on the meaning and usability of log time value and payload have been conducted [19]. Ke Wang and Salvatore J. Stolfo have proposed a system for detecting network intrusions based on anomaly payloads [20]. The system detects when anomaly payloads occur that are different from the expected behavior through profiling normal payloads. In this paper, the basic design criteria and operational objectives of anomaly detection systems are automation with less human intervention, accuracy of detecting anomaly events, low false positive rate, responsiveness of mimicry attack, and efficiency to reduce processing time. For the feature preprocessing, payload information such as payload length and flow direction were used, and n-gram and Mahalanobis distance techniques were used. They also experimented with the distribution of payload bytes for each port number and proposed a model for detecting anomaly by port.

Wei-Chao Lin et al. proposed an intrusion detection system based on the distance between the cluster center and the closest data [21]. First, find the distance between k-means-based cluster core and the data to be detected. The distance between the data to be detected and the closest data in the cluster corresponding to the found core is obtained. Then create the feature using the sum of all distances. The more anomaly data, the larger the sum of the distances, indicating that the distance-based feature is meaningful for anomaly detection. By comparing k-means and k-Nearest Neighbors (k-NN), they explained that the proposed system is fast, with little difference in detection
performance. Figure 2 shows the difference in performance using different dimension dataset for the two models.

![Figure 2. The performance of cluster center and nearest neighbor (CANN) and k-NN over 5 classes.](image)

Dominique T. Shipmon et al. proposed a method for predicting time series data and detecting abnormal symptoms using various models such as deep neural network (DNN), recurrent neural network (RNN), long short-term memory (LSTM), and Fourier model [22]. They convert Unix timestamps into various forms in a series of stream data and use them as features and byte counts as labels. Figure 3 shows an example for detecting abnormal symptoms using DNN. The blue line is the actual values, the green line is the predictions, and the shaded-red areas are the detected anomalies.

![Figure 3. Example of time series data analysis result for abnormal symptom detection using deep neural network (DNN) model.](image)

The DarkTrace is equipped with machine learning-based technology to learn and judge network anomalies by itself. Existing APT solutions and network forensic solutions rely on existing anomaly data, so they cannot respond to new patterns or threats not found in data. Darktrace’s enterprise immune system learns, infers, and visualizes user and device and network behavior as shown in Figure 4. In addition, DarkTrace builds over 250 threat models, as shown in Table 2, and detects threats based on this. This is an overwhelming number compared to 32 competitors’ modeling. This allows for more sophisticated threat detection.

Jae-sung Yun et al. [23] proposed an efficient mobile malware classification method by profiling behaviors of mobile malware using profiling techniques. They use DroidBox, a dynamic emulator tool, to parse the integrated system log. It creates malware profiles and classifies applications according to their behavior patterns.

Wu Liu et al. [24] proposed a malware detection algorithm based on malicious behavior functions. This paper investigates the malware behavior extraction technology and presents the MBF (malware behavior feature) extraction method. They designed and implemented the MBF based malware...
detection system based on a malware detection algorithm. The basic detection process uses malware behavior data to calculate the Boolean expression of MBF.

![Diagram](image_url)

**Figure 4. DarkTrace enterprise immune system operation process.**

**Table 2. Darktrace’s built threat model.**

| Detection Classification | Number of Threat Models | Classification of Detection Threats |
|--------------------------|-------------------------|-------------------------------------|
| Anomalous Connection     | 17                      | IGB Outbound, Active RDP Tunnel, Active SSH Tunnel Etc. |
| Anomalous File           | 12                      | Incoming RAR File, Masqueraded File Transfer, Outgoing RAR File Etc. |
| Anomalous Server Activity| 15                      | Data Transfer—DC to Client, DC External Activity, Domain Controller DynDNS SSL or HTTP Etc. |
| Attack                   | 2                       | Attack and Recon Tools, Exploit Kit, GoNext redirection |
| Compliance               | 42                      | Bitcoin Activity, External SNMP, External Windows Communications Etc. |
| Compromise               | 26                      | Beaconing to Rare Destination, Connection to Sinkhole, CryptoLocker Etc. |
| Device                   | 17                      | Address Scan, External DNS Domain Pointing at Local IP address, New User Etc. |
| Experimental System      | 66                      | Excessive HTTP Errors, Heartbleed SSL Success, International Domain Name Etc. |
| Unusual Activity         | 14                      | Unusual Activity, Unusual Activity from New Device, Unusual External Activity, Unusual External Connections |
| User                     | 7                       | Brute forcing, Kerberos Bruteforce, Multiple New Credentials on Device Etc. |

3. Proposal Model

3.1. Overview

In order to detect anomalies occurring at the endpoint, the data characteristic difference between the existing log and the newly generated log is used. A large difference between normal log data in the database and new log data that does not exist in the database can be identified as an anomaly. This section proposes an anomaly detection method for event logs such as files, processes and modules using LOF and AutoEncoder. In addition, this section proposes single event rules and complex event rules generated through attack profiles to detect possible threats. Figure 5 shows the overall structure of the proposed model. First, it extracts features for analyzing anomalies based on logs collected from the endpoint. LOF and AutoEncoder are applied to calculate the anomaly score representing the difference between data. LOF is assigned an LOF score per event, and AutoEncoder is assigned a loss value per event. These values are used to identify anomaly data with abnormal symptoms after the cumulative distribution function (CDF) is computed using a standard normal distribution. The generated CDF values are used as anomaly scores to detect a single suspicious event. These models can also detect suspicious Internet Protocol addresses (IPs) and processes and classify data by process for detailed analysis in statistical process. If anomalies are detected using an anomaly score, you can also consider the collective anomaly technique using flow data collection [25,26]. It also suggests how to detect suspicious threats through analysis of attack profiles. This model analyzes events step by step according to the attack scenario. At this time, if each process detects data and same event occurs continuously, it is judged as high risk. It can reduce false alarms by weighting logs that constantly perform malicious actions.
LOF is one of the typical anomaly detection techniques. The advantage of LOF is that it detects anomalies even if it is a little far away from a very dense cluster. In other words, LOF is calculated and statistically interpreted to detect abnormal symptoms in the endpoint log [27]. LOF is calculated based on k-NN. The k-NN algorithm computes the Kth nearest neighbor between the data. If the test data is far from the normal data, the distance value can be used as a score to determine whether the test data is anomaly. We use the k-distance equation for k-NN, which uses the Minkowski distance. The Minkowski distance \((X, Y)\) are limited to a maximum value and a minimum value, as in Equation (1):

\[
\text{Minkowski distance} \left( X, Y \right) = \left( \sum_{i=1}^{n} \left| x_i - y_i \right|^p \right)^{\frac{1}{p}}
\]

LOF is an approach based on local outlier density. The local density of Equation (2) is inversely proportional to the mean distance of the k-distance. The \(LOF(X)\) in Equation (3) is calculated as the \(\text{local density}_{avg}(X)\) divided by \(\text{local density}(X)\) of the data.

\[
\text{Local density}_{k}(X) = \frac{|N_k(p)|}{\sum_{k\_distance(X,Y)}}
\]

\[
LOF(X) = \frac{\text{local density}_{avg}(X)}{\text{local density}(X)}
\]
The numbers in Figure 6 represent the LOF score. You can see that outliers close to very dense areas have higher LOF values. The downside is that you must decide where to set the criteria that you think are outliers. It is also complex as the dimension increases. A value of 1.1 in one dataset is an outlier, but a value of 2 in another dataset can be normal.

Table 3. Feature preprocessing of Endpoint log.

| Used Field     | Method of Preprocessing                  | Input Example          |
|---------------|------------------------------------------|------------------------|
| Process Name  | SHA256 (2-g (ProcessName)) mod 15        | chrome.exe             |
| IP address    | MinMaxScaling (IP address)               | 192.168.0.1            |
| Destination IP| MinMaxScaling (octet.split (Dst IP))     | 127.0.0.1              |
| Event Time    | SHA256 (2-g (EventTime)) mod 2          | 1564519160346          |
| File Name     | SHA256 (2-g (FileName)) mod 15          | CHROME_PATCH.PACKED.7Z |
| Event Type    | If (EventType = file):
               | feature = 0
               | else if (EventType = module):
               | feature = 1
               | else if (EventType = process):
               | feature = 2
| Event sub Type| If (subtype = fileCreate):
               | feature = 0
               | else if (subtype = docOpen):
               | feature = 1
               | else if (subtype = moduleLoad):
               | feature = 2
               | else if (subtype = childProcessCreate):
               | feature = 3

Figure 6. Examples of local outlier factor (LOF) values in different density populations.

We simply use LOF values to statistically analyze the data. Anomaly detection takes advantage of the fact that the larger the distance between data, the more different it is from normal data. We analyze the data statistically by calculating the z-score and CDF using the LOF values. Figure 7 shows the cumulative distribution function. The CDF indicates the probability that a random variable is less than or equal to a certain value for a certain probability distribution. Therefore, CDF is calculated for statistical analysis using LOF values and the calculated CDF is used as an anomaly score. If the anomaly score is greater than the set threshold, it is classified as a single suspicious event.
### 3.2.2. AutoEncoder Based Anomaly Detection

Unsupervised learning-based AutoEncoder is used to calculate anomaly scores. AutoEncoder is simply a neural network that copies inputs to outputs. If the AutoEncoder model only trains normal events, it uses the fact that the loss values are large when predicting abnormal data \([28,29]\). The extracted features are created by referring to a single event rule that can cause threat. However, the system behavior model and network behavior model were independently generated for accurate model learning. The process path and event time apply equally to both network behavior and system behavior features. Features using the process path are extracted by applying feature hashing to the string. The event time is based on the day of the week and the time it occurred. In the network behavior feature, there was a destination IP address field that we tried to access, and the entire feature hashing was performed. To add the suspicious local IP address of the destination IP address, the min–max scaling was applied using only the A and B classes. For the system behavior feature, feature values are specified according to the process type, event type, and file type. Table 4 shows the processing methods and sample results of some feature vectors.

#### Table 4. Example feature preprocessing of AutoEncoder model.

| Network Behavior Features | System Behavior Features |
|---------------------------|--------------------------|
| **Used Field**            | **Method of Preprocessing** | **Input Example** |
| Destination IP            | SHA256 (2-g (IP)) mod 10  | 127.0.0.1          |
| Event Time                | MinMaxScaling (IPA._Class, IPB._Class)  | 127.0 |

| Process Path               | SHA256 (2-g (ProcessPath)) mod 10                      | C:\Program Files (x86)\Microsoft Office\Office15\POWERPNT.EXE |
|----------------------------|--------------------------------------------------------|---------------------------------------------------------------|
| Event Time                 | If (EventTime = Monday): feature = 1 else if (EventTime = Tuesday): feature = 2 else if (EventTime = Sunday): feature = 7 | 1564519160346 |
|                            | If (EventTime = Weekday): feature = 1 else if (EventTime = Weekend): feature = 2 |                                                             |
|                            | If (EventTime = 0–3 time): feature = 0 else if (EventTime = 3–6 time): feature = 1 else if (EventTime = 21–24 time): feature = 8 |                                                             |

**Figure 7.** Cumulative distribution function example.
Network behavior model features and system behavior model features are configured differently. This is for the AutoEncoder model to learn only necessary information in network and system behavior. Therefore, in order to obtain an accurate anomaly score, an independent model must be created.

The AutoEncoder configuration consists of the same input layer and output layer. Accordingly, the number of input layer nodes and the number of output layer nodes are the same. Each model 20 has epochs. Loss values can be obtained for each test data and used to statistically analyze the loss values. We calculated the z-score and CDF using the loss values. The CDF indicates the probability that a random variable is less than or equal to a certain value for a certain probability distribution. Therefore, the CDF is calculated for the statistical analysis using loss values, and the calculated CDF is used as an anomaly score. If the anomaly score is greater than the set threshold, it is classified as a single suspicious event.

3.3. Attack Profile Analysis

3.3.1. Attack Scenario

The attack profile creates rules to detect malicious behavior by profiling when the attack log corresponding to the scenario occurs. This alerts you to the threat when two or more suspicious events occur. It is effective in detecting advanced attacks such as APT attacks on endpoints. Visual understanding can be found in the Appendix A. The scenarios applied to the attack profile are as follows:

Drive by Download
1. Create malicious executable file using web connection, malicious link, or email attachment.
2. The malicious executable file periodically connects to command and control (C&C) server and receives attacker commands.
3. Various malicious behaviors are performed such as scanning, access to internal main server, receiving additional malicious files, and leaking information to the outside.

Ransomware
1. Users open Chrome and download files that they think are safe.
2. The executed file starts PowerShell, deletes the local backup data, and then encrypts all data on the disk.

Cryptojacking
1. Script-based coin mining takes place within the web browser through scripts embedded in the web page.
2. The computing power possessed by the web page visitor is used to exploit cryptocurrency mining attacks throughout the web page.

Fileless-1
1. The user visits a specific site using a web browser.
2. Visiting this site loads a flash to exploit the vulnerability. Flash can use PowerShell to execute certain commands.
3. PowerShell connects to the C&C server to download and execute malicious scripts.

Fileless-2
1. The user opens a Microsoft Word document.
2. Inside a Word document is a macro that executes VBScript.
3. When the macro runs, the Word process reaches the C&C server specified by the attacker and downloads the DLL.

4. The DLL is loaded and allocates memory so that the DLL can be inserted into the running process.

Drive by Download is the most common among attack scenarios. Drive by Download is a hacking technology that allows malicious software to be downloaded to a user’s device without the user’s knowledge when the user accesses a specific email or website. Most malware is infected by the Drive by Download method. However, fileless attacks do not download files such as malware, so there is no evidence to analyze and bypass traditional anti-virus.

3.3.2. Single Event Rules

Referring to the attack scenario above, we need to create rules for a single event step by step. The single event rules are:

Unusual network connection
1. Network connection occurs to a rare destination from a PC that has never had access records in the past.
2. A network connection occurs on an IP address that has no connection history in the same group.
3. A network connection occurs to a rare destination at an abnormal time (e.g., 10 p.m. to 6 a.m. weekend).
4. Network connections occur to rare destinations at irregular intervals.

Unusual download and upload of data
1. Create a portable executable (PE), zip, script, or dll file at an unusual time, path, or interval.
2. Create a PE, zip, script, or dll file of the capacity you did not download.
3. Create a file whose process is a PE, zip, script, or dll.

Unusual data transfer of process
1. An internal user performs a document open, file open, file delete, or file move at an abnormal time.
2. Run a process such as PowerShell or WMI that a user has not used in the past.
3. Download Script and execute process like wscript.exe or cscript.exe.

Single event rules are classified into known attack patterns and unknown attack patterns. For example, if a user tries to connect to a network with a rare IP address, it is a known attack pattern if the IP address is in the denylist or allowlist. Conversely, if a rare IP address not in the denylist or allowlist, it is classified as an unknown attack pattern. Additionally, if a process that has not been used in the past is in a denylist or allowlist, it is classified as a known attack pattern, and if it is not in a denylist or allowlist, it is classified as an unknown attack pattern. Therefore, it is easy to respond to known attack patterns, but it is difficult to respond to unknown attack patterns. In this paper, we propose an anomaly score-based anomaly detection approach to detect a single, suspicious, unknown event. Furthermore, the event log of the same process can be an abnormal log when malicious behavior is constantly detected. This can increase the detection rate of malicious behavior rather than detecting a single event by applying complex event rules generated through an attack profile.

3.3.3. Complex Event Rules

Complex event rules analyze a single suspicious event from an endpoint and determine the threat accordingly. In general, users run files like doc and pdf without question. Using this point, some attackers can use Microsoft Office macros to run PowerShell and inject script from outside. In addition, they can attempt to execute the preceding process to execute the script. In the case of a process, when wscript.exe, cscript.exe, PowerShell, or WMI are executed on a user’s PC, it may be detected as malicious behavior. Detecting these targets through complex event rules can improve
anomaly detection. In other words, using Microsoft Office is one normal single event, but when an abnormal time or an abnormal process event occurs, it is considered dangerous by complex event rules. Complex event rules are a model for detecting threats using the results of each single event rules. The detailed flow chart is shown in Figure 8. The rectangles represent single event rules. Dotted lines represent complex event rules that combines two or more single events. In addition, it detects as a threat even if suspicious activity continues in the same process. For example, if 10 or more suspicious activities color in the same process, it is detected as a threat.

4. Experimental Results and Analysis

4.1. Dataset

The dataset used in this experiment is different from the data used for LOF and the data used for AutoEncoder. Table 5 shows the dataset configuration used. Dataset-1 is a dataset used for LOF-based anomaly detection. This is data collected by itself from 5 common PCs; 664,928 data collected from 11 July 2019 to 29 July 2019 were used as training data, and 98,872 data collected from 30 July 2019 to 3 August 2019 were used as test data.

| Dataset | Collection | Training | Test |
|---------|------------|----------|------|
| Dataset-1 | Self-collection | Collection period 11 July 2019–29 July 2019 | Event Count 664,928 | Collection period 30 July 2019–3 August 2019 | Event Count 98,872 |
| Dataset-2 | Genians | Collection period 1 May 2019–31 May 2019 | Event Count 2,201,780 | Collection period 1 December 2019–25 December 2019 | Event Count 67,364 |

The main fields of the dataset consist of process name, file name, event type, event sub type, event time, IP address, remote IP address, Local IP address, process path, file path, and file type. Because it is data collected by itself, the file name and file path are not encrypted and consist of normal data without security incident. Dataset-2 is a dataset used for AutoEncoder-based anomaly detection. This was provided by Genians. Genians is a Korean integrated security platform company. The training data used about 2,201,780 data collected in May 2019, and the test data used about 67,364 data collected in December 2019. The main fields of the dataset consist of process name, file name, event type, event sub type, event time, IP address, remote IP address, Local IP address, process path,
file path, and file type. Since the data is provided by the company, the file name and file path are encrypted. Therefore, the file name and file path fields cannot be used.

4.2. Anomaly Event Analysis Results

4.2.1. LOF Based Anomaly Detection Results

Based on the anomaly scores generated from the LOF value, we detected a single suspicious event. As a result of the experiment, 5 suspicious processes were detected in the detected events. The process name was analyzed using the Hybrid Analysis (HA) site [30]. Table 6 shows the results of analyzing the processes judged to be anomaly. Two of these have been allowlisted, but the process has confirmed that the process performed a suspicious action such as a suspicious indicator. This confirmed that the proposed model was effective in detecting suspicious logs. It operates based on unsupervised learning, which causes some false positives, but it is expected that stable operation will be possible if allowlisting is applied.

4.2.2. AutoEncoder Based Anomaly Detection Results

In the case of AutoEncoder, system behavior and network behavior were independently tested. The AutoEncoder method also detects a single suspicious event detected based on the anomaly score generated from the loss value. As a result of the experiment, 5 suspicious processes were detected in the detected events. Three of them were detected in network behavior and the other 2 were detected in system behavior. The process name was analyzed using Hybrid Analysis site. Table 7 shows the results of analyzing the processes judged to be anomaly. Among them, it was not confirmed as an allowlist, and the process confirmed that suspicious actions such as suspicious indicators were performed. This confirmed that the proposed model was effective in detecting suspicious logs. It operates based on unsupervised learning, which causes some false positives, but it is expected that stable operation will be possible if allowlisting is applied.

| Process Name (Event Type) | Anomaly Score | Suspicious Indicators in Hybrid Analysis |
|---------------------------|--------------|------------------------------------------|
| InvColPC.exe (Process)    | 1.0%         | • Contains ability to start/interact with device drivers  
|                           |              | • PE file has unusual entropy sections  
|                           |              | • Found a cryptographic related string  
|                           |              | • Contains ability to find and load resources of a specific module  
|                           |              | • Drops executable files  
|                           |              | • The input sample dropped very many files  
|                           |              | • Possibly tries to implement anti-virtualization techniques  
|                           |              | • Reads the active computer name  
|                           |              | • Contains ability to lookup privileges  
|                           |              | • Contains ability to elevate privileges  
|                           |              | • Imports suspicious APIs  
| cleanmgr.exe (File)       | 0.998997%    | • Found a system process name at an unusual pathway  
|                           |              | • Contains native function calls  
|                           |              | • Imports suspicious APIs  
| HxTsr.exe (Module)        | 0.998945%    | • The input sample contains a known anti-VM trick  
|                           |              | • Contains ability to listen for incoming connections  
|                           |              | • CRC value set in PE header does not match actual value  
|                           |              | • Imports suspicious APIs  

Table 6. LOF based single suspicious event analysis results.
Table 6. Cont.

| Process Name          | Anomaly Score | Suspicious Indicators in Hybrid Analysis                                                                 |
|-----------------------|---------------|----------------------------------------------------------------------------------------------------------|
| dismhost.exe          | 0.998945%     | • Launches the WMI Provider Host                                                                         |
|                       |               | • Contains ability to find and load resources of a specific module                                       |
|                       |               | • Contains ability to elevate privileges                                                                  |
|                       |               | • Contains ability to lookup privileges                                                                   |
|                       |               | • Imports suspicious APIs                                                                               |
| HimTrayIcon.exe       | 0.998889%     | • Contains native function calls                                                                        |
|                       |               | • Contains ability to query CPU information                                                              |
|                       |               | • Scans for the windows taskbar (may be used for explorer injection)                                     |
|                       |               | • Imports suspicious APIs                                                                               |
|                       |               | • Installs hooks/patches the running process                                                              |

Table 7. AutoEncoder based single suspicious event analysis results.

| Process Name                        | Anomaly Score | Suspicious Indicators in Hybrid Analysis                                                                 |
|-------------------------------------|---------------|----------------------------------------------------------------------------------------------------------|
| TouchpointAnalyticsClientService.exe | 1.0%          | • Contains ability to open/control a service                                                             |
| (Process)                           |               | • Queries kernel debugger information                                                                   |
|                                     |               | • Creates guarded memory regions (anti-debugging trick to avoid memory dumping)                         |
|                                     |               | • Reads the cryptographic machine GUID                                                                  |
|                                     |               | • Found an IP/URL artifact that was identified as malicious by at least one reputation engine            |
| HPSupportSolutionsFrameworkService.exe | 1.0%         | • Possibly checks for the presence of an Antivirus engine                                               |
| (Module)                            |               | • Found a reference to a WMI query string known to be used for VM detection                             |
|                                     |               | • Found potential IP address in binary/memory                                                            |
|                                     |               | • Contains indicators of bot communication commands                                                     |
| FlashUtil32_32_0_0_303_Plugin.exe   | 1.0%          | • PE file has unusual entropy sections                                                                 |
| (Network)                           |               | • Found an IP/URL artifact that was identified as malicious by at least one reputation engine          |
|                                     |               | • Contains ability to find and load resources of a specific module                                      |
|                                     |               | • Drops executable files                                                                                |
|                                     |               | • Found potential IP address in binary/memory                                                            |
|                                     |               | • Opens file with deletion access rights                                                               |
|                                     |               | • CRC value set in PE header does not match actual value                                               |
|                                     |               | • Imports suspicious APIs                                                                               |
| IntelSoftwareAssetManagerService.exe | 0.999997%     | • Imports suspicious APIs                                                                               |
| (Network)                           |               | • Queries kernel debugger information                                                                   |
|                                     |               | • Queries the internet cache settings                                                                  |
|                                     |               | • Creates guarded memory regions                                                                        |
|                                     |               | • Monitors specific registry key for changes                                                            |
|                                     |               | • Opens the MountPointManager (often used to detect additional infection locations)                   |
|                                     |               | • Sends traffic on typical HTTP outbound port, but without HTTP header                                 |
|                                     |               | • Uses a User Agent typical for browsers, although no browser was ever launched                         |
|                                     |               | • Marks file for deletion                                                                               |
|                                     |               | • Opens file with deletion access rights                                                               |
|                                     |               | • Modifies proxy settings                                                                               |
| 3.5.5.45395.exe                   | 0.999984%     | • Queries sensitive IE security settings                                                               |
| (Network)                           |               | • Imports suspicious APIs                                                                               |
|                                     |               | • Queries kernel debugger information                                                                   |
|                                     |               | • Queries the internet cache settings                                                                  |
|                                     |               | • Creates guarded memory regions                                                                        |
|                                     |               | • Monitors specific registry key for changes                                                            |
|                                     |               | • Opens the MountPointManager (often used to detect additional infection locations)                   |
|                                     |               | • Sends traffic on typical HTTP outbound port, but without HTTP header                                 |
|                                     |               | • Uses a User Agent typical for browsers, although no browser was ever launched                         |
|                                     |               | • Marks file for deletion                                                                               |
|                                     |               | • Opens file with deletion access rights                                                               |
|                                     |               | • Modifies proxy settings                                                                               |
|                                     |               | • Queries sensitive IE security settings                                                               |


4.3. Attack Profile Analysis Results

Attack profile analysis detects threats that can arise with suspicious processes from the anomaly event analysis proposed earlier. Match the single suspicious event detected using the single event rules in Section 3.2.2. Then, threats are detected by the same process/endpoint IP address using the complex event rules in Section 3.3.3. In addition, it is judged as a threat when suspicious events occur continuously by the same process. Table 8 shows the unknown environment proposed in this paper and the results of detecting suspicious processes without prior knowledge. It is the result of the final analysis by applying complex event rules based on the suspicious process detected. It also shows how certain processes intruded from a specific IP address at a specific time. It also shows the threat level in hybrid analysis and the number detected by several antivirus engines provided by VirusTotal (VT) [31].

Table 8. Unknown intrusion detection results. IP is masked with '*' because it can be classified as identification information.

| Dataset | Time             | IP Address     | Process Name               | Complex Event Rules (Attack Profile)                                           | Threat Level (Hybrid)       | A/V Detection (VirusTotal) |
|---------|------------------|----------------|----------------------------|--------------------------------------------------------------------------------|----------------------------|---------------------------|
|         | 2 August 2019    | 210.125.*.*    | InvColPC.exe               | Anomaly system behavior and consistent threshold anomalies in the same process | Malicious                   | 2/71                      |
|         | 31 July 2019     | 210.125.*.*    | cleanmgr.exe               | Anomaly system behavior and consistent threshold anomalies in the same process | -                          | 0/71                      |
|         | 2 August 2019    | 210.125.*.*    | HimTrayIcon.exe            | Anomaly system behavior and consistent threshold anomalies in the same process | Ambiguous                   | 6/54                      |
|         | 14 December 2019 | 172.29.*.*     | TouchpointAnalytics        | Anomaly system behavior and consistent threshold anomalies in the same process | -                          | 0/71                      |
|         | 15 December 2019 | 172.29.*.*     | HPSupportSolutions         | Anomaly system behavior and consistent threshold anomalies in the same process | Suspicious                  | 0/85                      |
|         | 14 December 2019 | 172.30.*.*     | FlashUtil32_32_0_0_303_Plugin.exe | Sequential occurrence of suspicious file creation and anomaly network behavior in the same process | Malicious                   | 0/71                      |
|         | 14 December 2019 | 172.30.*.*     | IntelSoftwareAsset         | Sequential occurrence of suspicious file creation and anomaly network behavior in the same process | Malicious                   | 0/71                      |
|         | 22 December 2019 | 172.30.*.*     | 3.5.5_45395.exe            | Sequential occurrence of anomaly network behavior and suspicious file creation in the same process | Malicious                   | 10/73                     |

Basically, to be classified as an anomaly by a single event rule, the anomaly score must be higher than a set threshold, and to be classified as a consistent suspicious behavior, it must occur 10 times as an anomaly within 10 min. According to the table above, in the process called cleanmgr.exe within 10 min, the anomaly score of the system behavior file creation event is higher than the threshold and is defined as anomaly system behavior & consistent threshold anomalies in the same process if it occurs 10 times. The process called FlashUtil32_32_0_0_303_Plugin.exe creates a suspicious file, connects to the network, and is Sequential occurrence of suspicious file creation & anomaly network behavior in the same process if the anomaly score of the network event is above the threshold. On the other hand, a process called 3.5.5_45395.exe first connects to the anomaly network and a suspicious file is created in the same process. As such, it is the result of attack profile analysis that judges the anomaly using the anomaly score and judges the behavior that occurred by the same process or IP address over time. The detailed performance of the proposed model is as follows. Dataset-1’s approach analyzes 163 events per second. Dataset-2’s approach analyzes 571 events per second. Therefore, the performance in the AutoEncoder method is better than the LOF method.

5. Discussion

This section discusses the interpretation of experimental results, differences from previous studies, and future operational policies. The analysis environment used in the experiment was AMD Ryzen.
Threadripper 1920X 12-Core Processor 3.50 GHz and 32 GB RAM. For our proposed models to operate in real time, the analysis time was measured in the same experimental environment. Analysis time includes the time to fetch data, perform anomaly detection, attack profile analysis, and save the results. Table 9 shows the analysis performance of the proposed models.

Table 9. Performance measurement results of the proposed model.

| Dataset (Proposed Model) | Total Time (s) | Analysis Per Second (Event) |
|--------------------------|----------------|-----------------------------|
| Dataset-1 (LOF)          | 612.93         | 166                         |
| Dataset-2 (Autoencoder)  | 118.38         | 571                         |

The LOF model takes 612.93 s to analyze the final attack profile, which analyzes 163 events per second. The Autoencoder model takes 118.38 s to analyze the final attack profile, which analyzes 571 events per second. The disadvantage of LOF is that it is slower than Autoencoder because it calculates the average distance of all data. On the other hand, Autoencoder is composed of the same input layer and output layer based on deep learning, so it can be said that it is faster than LOF because the anomaly score is calculated as the loss value through learning. The anomaly detection method of the existing studies has a disadvantage in that a security manager needs to analyze the threat and label each log from an operation point of view because it is learned and detected using a label. In this study, it is possible to learn without a label, so it is advantageous to analyze the threat and not to label each log. In addition, the training period is set to train and the test can be operated in real time. Figure 9 shows the procedure in real time. The performance of the anomaly detection model is improved by providing normal behavior information to the next training model using the results of the model trained in a specific training period.

Besides, the proposed technology based on the existing behavior log for reliable anomaly detection. Figure 10 is a flow chart that suggests that the analysis results can be efficiently operated in connection with legacy systems such as allowlist, denylist, and pattern-based policies. The proposed model is a process for detecting suspicious EDR events. Anomaly detection results are displayed for each event, and each event is checked for abnormal behavior.

Figure 10 proceeds with anomaly detection if the event does not exist in the denylist. If it is determined to be abnormal, check the allowlist database. Events that are not in the allowlist database are analyzed manually by experts. If it determines malicious behavior, it can update the denylist database and if it is a normal event, it can update the allowlist database. Therefore, it is expected that the allowlist or denylist policy will work effectively because many events occur at the endpoint. Table 10 shows an example of reducing the subject of review through the operation of the allowlist.
we proposed an anomaly score-based detection method and attack profile technique to counter threats. EDR technologies are limited to the role of ensuring visibility of anomalies in the internal network. Policies can be applied for stable anomaly detection on each endpoint device. As an example, 2020 Electronics LOF, 44 by AutoEncoder-based system behavior, and 24 by network behavior. In addition, various applied and operated in real time regardless of an endpoint event log type or label. As a result of caused by malware intrusion. The proposed anomaly detection method is a model that can be rather than using probability values to determine anomalies occurring at endpoints. Therefore, limited to windows-based systems. With these security trends, recent endpoint detection and response and include sensitive functions such as financial services, existing malware-related studies are mainly growing endpoints in hyper-connected society. Although endpoint devices are mostly IoT devices and as a result of antivirus detection, 3 are considered malicious processes, and the remaining 64 suspicious processes are updated with an allowlist. Through this policy, the subject of review will be reduced over time. Thereafter, updating the training model with the latest date and operating it reduces the burden on the security administrator. In this paper, 107 suspicious processes are detected and as a result of antivirus detection, 3 are considered malicious processes, and the remaining 64 suspicious processes are updated with an allowlist. In addition, it detects 66 suspicious processes based on Autoencoder, detects 3 malicious processes as a result of antivirus detection, and updates the remaining 63 suspicious processes to an allowlist.

6. Conclusions

This paper suggests the necessity of security measures against security threats against rapidly growing endpoints in hyper-connected society. Although endpoint devices are mostly IoT devices and include sensitive functions such as financial services, existing malware-related studies are mainly limited to windows-based systems. With these security trends, recent endpoint detection and response (EDR) technologies are limited to the role of ensuring visibility of anomalies in the internal network rather than using probability values to determine anomalies occurring at endpoints. Therefore, we proposed an anomaly score-based detection method and attack profile technique to counter threats caused by malware intrusion. The proposed anomaly detection method is a model that can be applied and operated in real time regardless of an endpoint event log type or label. As a result of the experiment, 107 new suspicious processes that were not previously detected were detected by LOF, 44 by AutoEncoder-based system behavior, and 24 by network behavior. In addition, various policies can be applied for stable anomaly detection on each endpoint device. As an example of

### Table 10. Example of reduction of review targets through allowlist operation.

| Period               | Suspicious Process Detection Results | Allowlist Filtering Count | Number of Suspicious Processes under Review | A/V (VT) Results | Allowlist Count |
|----------------------|-------------------------------------|---------------------------|---------------------------------------------|------------------|-----------------|
| 1–15 March 2020      | 155                                 | 0                         | 155                                         | 6                | 149             |
| 16–31 March 2020     | 173                                 | 106                       | 67                                          | 3                | 213             |
| 1–15 April 2020      | 148                                 | 90                        | 58                                          | 4                | 267             |
| 16–30 April 2020     | 108                                 | 87                        | 21                                          | 1                | 287             |

For example, 155 suspicious processes are detected by the anomaly detection model, and the allowlist filtering count is 0, so the suspicious process to be reviewed does not change. As a result of anti-virus detection, 6 are judged as malicious processes, and the remaining 149 suspicious processes are updated with an allowlist. Next, 173 suspicious processes are detected and 106 processes are filtered by 149 allowlists. The remaining 67 processes are considered suspicious processes to be reviewed, and as a result of antivirus detection, 3 are considered malicious processes, and the remaining 64 suspicious processes are updated with an allowlist. Through this policy, the subject of review will be reduced over time. Thereafter, updating the training model with the latest date and operating it reduces the burden on the security administrator. In this paper, 107 suspicious processes are detected based on LOF, and 10 processes are judged as malicious processes as a result of antivirus detection, and the remaining 97 suspicious processes are updated with an allowlist. In addition, it detects 66 suspicious processes based on Autoencoder, detects 3 malicious processes as a result of antivirus detection, and updates the remaining 63 suspicious processes to an allowlist.

![Proposed policy flow chart.](image-url)

Figure 10. Proposed policy flow chart.
model operation, we also proposed the operation policy of legacy system using anomaly score-based detection. The attack profile technique determines suspicious events by associating consecutive events. This allows us to determine the risk of events occurring in the same process and respond quickly to each scenario. The scenario of the proposed attack profile is expected to be able to detect and analyze EDR threats. In order to ensure the continuous operation and practicality of the proposed model, we plan to verify the data and improve the model in many malware environments.

**Author Contributions:** Conceptualization, S.K.; methodology, C.H. and T.L.; software, S.K. and C.H.; validation, C.H.; formal analysis, C.H.; investigation, S.K., C.H. and T.L.; resources, X.X.; data curation, S.K.; writing—original draft preparation, S.K.; writing—review and editing, C.H.; visualization, C.H.; supervision, T.L.; project administration, C.H.; funding acquisition, T.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by Institute for Information & communication Technology Planning and evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00026, ICT infrastructure protection against intelligent malware threats).

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

**Appendix A**

There are many cases in the attack profile, and it is very important to accurately identify and respond. Accordingly, we introduce two representative detection scenarios according to various attack modules. Figure A1 shows the suspected situation of malware download. This indicates the situation in which a PE or zip file is created after a specific process is connected to a rare place. At this time, a specific process is a process such as powershell, cmd and WMI tool, and such a process is a rare place where there was no usual connection. If a PE or zip file is created by accessing, the situation can be regarded as a suspicious sign for downloading malware.

![Figure A1. Attack Profile Scenarios for suspected malware download.](image)

Figure A2 shows a situation in which C & C access is suspected due to malware infection. It shows a situation where a network connection occurs in a rare place within a certain time after a specific process creates a PE or zip file. If a connection to a rare network that has not been normally accessed occurs within a certain period of time after a file is created through a specific process, this situation is suspected to be an attack.

![Figure A2. Attack Profile C & C access suspicious scenario due to malware infection.](image)
Figure A1. Attack Profile Scenarios for suspected malware download. It shows a situation where a network connection occurs in a rare place within a certain time after a specific process creates a PE or zip file. If a connection to a rare network that has not been normally accessed occurs within a certain period of time after a file is created through a specific process, this situation is suspected to be an attack.

**Figure A2.** Attack Profile C & C access suspicious scenario due to malware infection.

**References**

1. Statista. Internet of Things (IoT) Connected Devices Installed Base Worldwide from 2015 to 2025. Available online: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ (accessed on 1 June 2020).
2. TÜV Rheinland. Cybersecurity Trends in 2020; TÜV Rheinland: Cologne, Germany, 2020.
3. Trend Micro. The New Norm: Trend Micro Security Predictions for 2020; Trend Micro: Tokyo, Japan, 2019.
4. Pajouh, H.H.; Javidan, R.; Khayami, R.; Dehghantanha, A.; Choo, K.-K.R.; Ali, D. A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks. *IEEE Trans. Emerg. Top. Comput.* 2016, 7, 314–323. [CrossRef]
5. Li, T.; Jiang, Y.; Zeng, C.; Xia, B.; Liu, Z.; Zhou, W.; Zhu, X.; Wang, W.; Zhang, L.; Wu, J. FLAP: An end-to-end event log analysis platform for system management. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 23–27 August 2017; pp. 1547–1556.
6. Zaman, M.; Siddiqui, T.; Amin, M.R.; Hossain, M.S. Malware detection in Android by network traffic analysis. In Proceedings of the 2015 International Conference on Networking Systems and Security (NSysS), Dhaka, Bangladesh, 5–7 January 2015; IEEE: New York, NY, USA, 2015; pp. 1–5.
7. Isohara, T.; Takemori, K.; Kubota, A. Kernel-based behavior analysis for Android malware detection. In Proceedings of the 2011 Seventh International Conference on Computational Intelligence and Security, Washington, DC, USA, 3–4 December 2011; IEEE: New York, NY, USA, 2011; pp. 1011–1015.
8. Sun, J.; Jeng, T.; Chen, C.; Huang, H.; Chou, K. MD-Miner: Behavior-based tracking of network traffic for malware-control domain detection. In Proceedings of the 2017 IEEE Third International Conference on Big Data Computing Service and Applications (BigDataService), San Francisco, CA, USA, 6–7 April 2017; IEEE: New York, NY, USA, 2017; pp. 96–105.
9. Malhotra, P.; Vig, L.; Shroff, G.; Agarwal, P. Long Short Term Memory Networks for Anomaly Detection in Time Series, *Proceedings*; Presses Universitaires de Louvain: Louvain, Belgium, 2015; p. 89.
10. Toledano, M.; Cohen, I.; Ben-Simhon, Y.; Tadeski, I. Real-time anomaly detection system for time series at scale. In Proceedings of the KDD 2017 Workshop on Anomaly Detection in Finance, Halifax, NS, Canada, 14 August 2017; pp. 56–65.
11. He, S.; Zhu, J.; He, P.; Lyu, M.R. Experience report: System log analysis for anomaly detection. In Proceedings of the 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), Ottawa, ON, Canada, 23–27 October 2016; IEEE: New York, NY, USA, 2016; pp. 207–218.
12. Gutierrez, R.J.; Boehmke, B.C.; Bauer, K.W.; Saie, C.M.; Bihl, T.J. anomalyDetection: Implementation of augmented network log anomaly detection procedures. *R. J.* 2017, 9, 354–365. [CrossRef]
13. Garg, S.; Kaur, K.; Kumar, N.; Kaddoum, G.; Zomaya, A.Y.; Ranjan, R. A Hybrid Deep Learning-Based Model for Anomaly Detection in Cloud Datacenter Networks. *IEEE Trans. Netw. Serv. Manag.* 2019, 16, 924–935. [CrossRef]

14. Ahmed, M.; Mahmood, A.; Hu, J. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* 2016, 60, 19–31. [CrossRef]

15. Alhawi, O.M.K.; Baldwin, J.; Dehghananta, A. Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection. In *Malicious Attack Propagation and Source Identification*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 93–106.

16. Jabez, J.; Muthukumar, B. Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach. *Procedia Comput. Sci.* 2015, 48, 338–346. [CrossRef]

17. Aljawarneh, S.; Aldwairi, M.; Yassein, M.B.; Yasin, M.B. Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *J. Comput. Sci.* 2018, 25, 152–160. [CrossRef]

18. KDD Cup 1999 Data. Available online: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 12 June 2020).

19. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization; ICISP: Funchal, Portugal, 2018; pp. 108–116.

20. Wang, K.; Stolfo, S.J. Anomalous Payload-Based Network Intrusion Detection, International Workshop on Recent Advances in Intrusion Detection; Springer: Berlin/Heidelberg, Germany, 2004; pp. 203–222.

21. Lin, W.-C.; Ke, S.-W.; Tsai, C.-F. CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowl. Based Syst.* 2015, 78, 13–21. [CrossRef]

22. Shipmon, D.T.; Gurevitch, J.M.; Piselli, P.M.; Edwards, S.T. Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data. *arXiv* 2017, arXiv:1708.03665.

23. Yun, J.-S.; Jang, J.-W.; Kim, H.K. Andro-profile: Anti-malware system based on behavior profiling of mobile malware. *J. Korea Inst. Inf. Secur. Cryptol.* 2014, 24, 145–154. [CrossRef]

24. Liu, W.; Ren, P.; Liu, K.; Duan, H. Behavior-based malware analysis and detection. In Proceedings of the 2011 First International Workshop on Complexity and Data Mining, Washington, DC, USA, 24–28 September 2011; IEEE: New York, NY, USA, 2011; pp. 39–42.

25. Bontemps, L.; McDermott, J.; Le-Khac, N. Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks. In Proceedings of the International Conference on Future Data and Security Engineering, Can Tho City, Vietnam, 23–25 November 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 141–152.

26. Ahmed, M.; Mahmood, A.N. Network traffic analysis based on collective anomaly detection. In Proceedings of the 2014 9th IEEE Conference on Industrial Electronics and Applications, Hangzhou, China, 9–11 June 2014; IEEE: New York, NY, USA, 23 October 2014; pp. 1141–1146.

27. Amer, M.; Goldstein, M. Nearest-neighbor and clustering based anomaly detection algorithms for rapidminer. In Proceedings of the 3rd RepidMiner Community Meeting and Conference (RCOMM 2012), Budapest, Hungary, 29 August 2012.

28. Chen, J.; Sathe, S. Outlier detection with autoencoder ensembles. In Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, TX, USA, 27–29 April 2017; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2017; pp. 90–98.

29. Zhou, C.; Paffenroth, R.C. Anomaly detection with robust deep autoencoders. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 665–674.

30. Hybrid Analysis. Available online: https://www.hybrid-analysis.com/?lang=es (accessed on 15 April 2020).

31. Anejo-Okopi, J.; Akindigh, T.M.; Markus, N.; Adeniyi, D.S.; Abba, O.J.; Ebonyi, A.O.; Ejeliogu, E.; Audu, O.; Lar, P.; Zumbes, H.J.; et al. Hepatitis B Virus Total Core Antibodies among HIV-1 Infected Hepatitis B Surface Antigen Negative Patients Attending a Tertiary Health Facility in North-central Nigeria. *Br. J. Med. Med. Res.* 2016, 18, 1–7. [CrossRef]