Analyzing System Log Based on Machine Learning Model

Chia-Mei Chen, Gen-Hong Syu, and Zheng-Xun Cai
(Corresponding author: Chia-Mei Chen)

Department of Information Management, National Sun Yat-sen University
No. 70, Lianhai Road, Gushan District, Kaohsiung 804, Taiwan
(Email: cchen@mail.nsysu.edu.tw)
(Received Feb. 7, 2020; revised and accepted June 8, 2020)

Abstract

Cyberattacks become one of the most concerning threats to governments, businesses, and people. Efficient incident investigation is vital to identify the root cause of an attack, which requires expertise in analyzing audit logs. System logs are an important audit trace to understand the status of systems and useful for analyzing anomalies in case of a security breach. However, due to the diversity and massive volume of the logs, log analysis requires expertise domain knowledge as well as a large amount of manpower and computing resources. To make an incident investigation more accessible, this study proposes a machine-learning-based system log analysis approach that identifies suspicious event activities automatically. The experimental results show that the proposed neural network-based approach could identify malware efficiently with the precision of 95.5% and outperforms SVM.

Keywords: Anomaly Detection; Big Data Analysis; Machine Learning

1 Introduction

The explosive expansion of electronic commerce offers unprecedented opportunities for businesses to expand their markets. Government, businesses, and people rely on seamless ubiquitous computing services heavily, putting valuable and confidential data over the internet and on clouds. However, these convenience services have been accompanied by a commensurate increase in cyber-attacks and caused serious damages and financial losses. Cybercrimes kept increasing worldwide these years. Not only have financial firms suffered serious losses from cyberattacks, but also high-tech companies, governments, and academic institutes have experienced severe data breaches.

Attackers attempt to exploit vulnerabilities at different layers, such as most common application technical or logic attacks [12], and penetrate target systems. In order to detect suspicious behaviors, systems record important activities performed. The process of recording events during the execution of the operating system, process, network, or application is called logging, which produces log files composed of useful information associated with events that occurred in the system, network, or application [11] and is useful for analyzing anomalies in case it is compromised. Incident investigation faces enormous data collection as well as data analysis, which consumes a lot of time for system administrators or incident investigators to discover suspicious behaviors from a massive amount of system logs. An efficient system log analysis method is desired to identify suspicious events.

There are two common attack detection approaches: misuse detection and anomaly detection. Misuse detection is a rule-based approach that contains the signatures of intrusion patterns and effective in identifying known attacks but usually performs poorly on new attacks or variants of known attacks. Anomaly detection approaches profile normal or abnormal behaviors and apply deviations or similarities from the respective profile to anomaly detection, which may employ statistics, machine learning, or data mining techniques to train the detection model.

Most studies analyzed network traffic or alerts from defense systems, but the issue of analyzing system logs was not yet fully explored in the literature. In real-world cases, attackers circumvented the defense mechanisms and implanted malware into target systems, and the defense systems failed to discover them.

Identifying attacks efficiently is time-critical in order to reduce the damage caused by an attack. The more informative audit trails are, the more efficiently the detection model could build to identify attacks. System logs provide informative data about the activities performed on systems and are useful for detecting suspicious events. However, analyzing audit trails is labor-intensive, and most organizations are short of security professionals as well as resources to perform the task promptly and efficiently. An automatic and efficient system log analysis method is desired.

Over 78% of systems are Windows-based [19], and in

addition, Windows has become attackers' favorite hacking platform [20]. Event logs provided by Sysmon (System Monitor) are comprehensive and useful for identifying suspicious activities on Window-based systems. Hence, this study focuses on analyzing system logs and identifying anomalous events for Windows-based systems.

Neural Networks (NNs) are one of the most popular machine learning algorithms at present. It has been decisively proven over time that NNs outperform other algorithms in accuracy and speed [13]. Recurrent neural networks (RNNs), a variation of NNs, handle time-series data efficiently like system logs. In order to make system log analysis and anomaly detection accessible for short-of-staff organizations, this study proposes an RNN-based system log analysis approach to automatically identify suspicious behaviors.

The rest of the paper is structured as follows: Section 2 briefly reviews the related studies on attack detection and log analysis and a background study of Windows system logs; Section 3 elaborates the proposed detection; The performance evaluation is presented in Section 4 followed by the concluding remarks and future work.

2 Related Work

Raftopoulos and Dimitropoulos [16] asserted that the alerts from intrusion detection systems often produce lots of false positives. They proposed an alert reduction method that employs entropy-based information-theoretic criteria to find recurring alerts. Zargar *et al.* [23] proposed an intrusion detection framework for a distributed computing environment where service providers collaborate to cope with attacks. Lo [9] proposed a framework for detecting attacks by exchanging alert information with other intrusion detection systems. A comprehensive trust management scheme is required to support the trust relationship among the service providers.

Liu *et al.* [18] proposed an alert correlation model for constructing attack scenarios that require the given attack graphs and signature rules to correlate security events. Siraj *et al.* [1] proposed a framework of attack prediction, which includes the following components: alert normalization, reduction, prioritization, and attack scenario construction and prediction. To obtain effective detection results, Amini *et al.* [2] proposed a detection method that combines multiple classifiers: neural network, fuzzy clustering, and stacking combination methods. The experimental results showed that the proposed multi-classifier approach performed better than single classifiers.

A significant amount of research has been contributed to attack detection. Various approaches including data mining, finite state machines, etc. have been explored in order to propose efficient approaches to identifying anomalies.

Serketzis *et al.* [17] proposed a log management system that collects the audit logs from network equipment, security devices, operating systems, and applications. Users could facilitate search function for discovering suspicious events. Dwyer and Truta [6] proposed a statistic-based approach for identifying anomalies in Windows event log data using standard deviation. The average number and standard deviation of the events of a specific type at any time of a day for any server or user are calculated; an anomaly is determined if an event goes outside of the standard deviation. The results show the proposed solution lowers the amount of logs to be reviewed to a feasible amount.

A Windows event log is a detailed record of the system, security, and application notifications stored by the Windows operating system and useful for identifying system faults and attacks. It can be broadly categorized into two levels of logs: the operating system and applications. Both can utilize event logs to record important events. Windows system logs record software installation, security management, system setup operations on initial startup, and problems or errors; the other level is service logs that record application related events. To identify suspicious events on a system, this study focuses on analyzing Windows system logs.

System Monitor (Sysmon) is a Windows system service to monitor and record system activities to the Windows event logs that provide detailed information about process creations, network connections, and changes to file creation time. Sysmon does not provide event analysis, but the official document claims that, by collecting and analyzing the event logs, anomalous activities can be discovered.

Various machine learning algorithms have been developed recently based on deep learning that exhibits remarkable capabilities in classification and clustering. Among them, supervised machine learning algorithms have been applied to anomaly detection. A classification task involves separating data into categories based on the information learned from the labeled training data, where each instance in the training data set contains one "target value" (*i.e.* class label) and some "attributes" (*i.e.* features or observed variables). The goal of a supervised learning algorithm is to produce a model (based on the training data) which classifies or predicts the target values of the test data given only the test data attributes.

The supervised learning machine model, SVM (support vector machine), is widely used in classification. Given a training set of instance-label pairs (xi; yi), an SVM model is to find a linear hyperplane with the maximal margin to separate the data. Four basic types of kernel functions are used in modeling the hyperplane: Linear, polynomial, radial basis function (RBF), and sigmoid. In general, the RBF kernel is a reasonable choice. This kernel nonlinearly maps the samples into a higher-dimensional space, so it can handle the case of a nonlinear relation between class labels and attributes. The linear kernel function is a special case of the RBF, and the sigmoid kernel behaves like RBF for certain parameters.

Zidi *et al.* [24] employed SVM classification model to

identify failures in wireless sensor networks and claimed that the fault detection has to be precise to avoid negative alerts and rapid to limit loss. Comparing with other algorithms, their study indicates that SVM is efficient. Wang *et al.* [22] proposed an efficient intrusion detection framework based on SVM and concluded a similar finding that SVM achieves a better performance than the existing methods in terms of accuracy, detection rate, false alarm rate, and training speed. Anton *et al.* [3] applied SVM to network anomaly detection on industrial environments and the experimental results showed that SVM performs well.

NN is one of the most popular machine learning algorithms. Convolutional Neural Networks (CNNs) perform well on pattern recognition and image categorization but are not suitable for data with time sequence. The neural network models have been employed on anomaly detection. Radford *et al.* [15] showed that RNN can represent sequences of communications on a network and discover anomalous network traffic. Prasse *et al.* [14] analyzed HTTPS network flows, employed a natural language model to extract features from domain names, and proposed an LSTM-based detection method, where LSTM (Long Short-Term Memory) is an RNN model dealing with time-series data. Their experimental results show that the LSTM classification model outperforms a random forest model. Kim and Ho [8] employed CNN to extract spatial features and LSTM temporal characteristics and proposed a neural network for detecting anomalies on web traffic.

3 System Design

According to the literature review, attackers might customize their attack to evade the detection mechanism. Sysmon logs provide detailed information about the actions performed on a system, including network connections, running processes, registry files, and file systems. Analyzing audit logs with informative details could improve detection performance. RNN models are suitable for analyzing time sequence data like such audit logs. This study proposes an RNN-based log analysis method to automatically classify suspicious events, where the system model is plotted in Figure 1. This research collected malware behaviors by emulating attacks in a controlled environment and collected normal user behaviors from a campus network. Both parts of the labeled data were verified manually.

Our preliminary study discovered that most attacks contain the following four types of behaviors: process, file access, registry, and network access, and all can be captured by the system logs. Log records are informative, but most are benign. Therefore, the preprocess module extracts security-relevant event records from log files in order to reduce the processing time in the following steps, where the security-relevant event records extracted represent the behavior of a given process. The proposed RNN model learns to classify misbehaviors from the labeled data collected from benign users and malware. The detail of the proposed method is explained below.

3.1 Security-Related Events

To identify the misbehaviors of a process, process behaviors should be captured and analyzed. In this study, process behaviors are delineated as a sequence of the events captured by the system logs. Based on our preliminary study, the above four types of security-relevant events are selected from Sysmon event logs for identifying critical activities and status changes of a system. As event attributes provide the detailed information of an event, key event attributes are selected to improve the description of an action performed. In summary, by describing process behaviors in a sequence of the performed events with the associated event attributes, the proposed anomaly detection method plots the status changes of the system and discovers anomalous behaviors. The security-relevant events and attributes selected in the proposed method are explained below.

Process Events

A binary image file needs to be loaded into memory to be able to run this program, while some malware injects its executable file to another legitimate process or kills a process. Such behaviors can be captured by process events. Based on the above injection anomaly and the literature review on malware misbehaviors, the selected critical process events/behaviors include creating a process, terminating a process, loading image, creating a remote thread, and accessing a process, where Table 1 summarizes the selected process events and attributes.

File Access Events

Malware accesses file system for various purposes. Downloader or dropper may download additional malware or payloads to perform further attack; ransomware accesses and encrypts documents and files; some malware steals and sends out confidential information. The file access events record the access behaviors of the file system. The selected critical file access events include changing file creation time, accessing a file, creating a file, and creating a file hash, where Table 2 summarizes the selected file access events and attributes.

Registry Events

The registry [10] is a hierarchical database that contains data about the operation of the Windows operating system, applications, and services. The data is structured in a tree format. Each node in the tree is called a key. Each key can contain both subkeys and data entries called values. Sometimes, the presence of a key is all the data that an application requires; other times, an application opens a key and uses the values

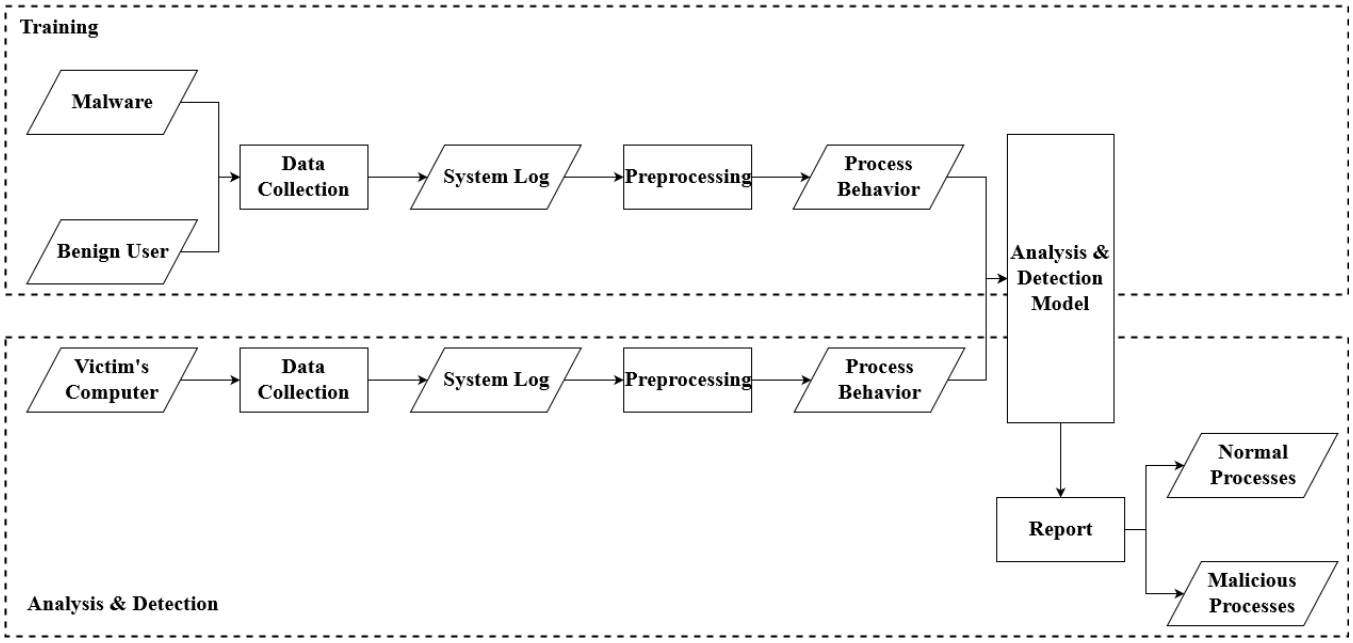

Figure 1: The proposed log analysis and anomaly detection method

associated with the key. Registry file contains important information including applications installed, files and paths recently accessed, network setup, and account information. Malware [7] might modify registry keys in order to achieve persistence on a system, like exploiting Run, RunOnce, BootExecute, Winlogon, and Startup Keys. The selected critical registry events include setting a registry key value, renaming a registry key and value, creating and deleting a registry object, where Table 3 summarizes the selected registry events and attributes.

Network Events

Most malware performs network connections for various purposes. For example, botnets connect to the command and control server for reporting victim information and receiving attack instruction; miner malware connects to the mining pool; Some malware attempts to exploit and infect more machines. Hence, network connections are critical in identifying misbehaviors. Table 4 summarizes the chosen network event and attributes.

3.2 Log Analysis and Anomaly Detection Method

The Sysmon system logs record the events performed by all the running processes in a system but lose the time order of process events. However, the time sequence is crucial to understand process behaviors. This study utilizes Process ID to link all the event records of a process in chronological order, and the observed process events are outlined in Figure 2.

The traditional neural network models perform poorly on time-series datasets; based on literature review LSTM is suitable for dealing with time-series data. This study employs an improved RNN as well as a variation of LSTM: GRN (Gated Recurrent Unit) [4],as it eliminates the vanishing gradient problem faced by standard RNN and produces equally excellent results as LSTM. The proposed GRU classification model consists of the input

Table 1: The selected process events and attributes

Event	Attribute
Event ID 1: Process creation	Event ID
	Image
	User
	ParentImage
Event ID 5: Process terminated	Event ID
Event ID 7: Image loaded	Event ID
	ImageLoaded
	Signed
Event ID 8: CreateRemoteThread	Event ID
	TargetImage
Event ID 10: ProcessAccess	Event ID
	TargetImage
	GrantedAccess

Table 2: The selected file access events and attributes

Event	Attribute
Event ID 2: A process changed a file creation time	Event ID
	TargetFilename
	CreationUtcTime
	PreviousCreationTime
Event ID 9: RawAccessRead	Event ID
Event ID 11: FileCreate	Event ID
	TargetFilename
Event ID 15: FileCreateStream-Hash	Event ID
	TargetFilename

Table 3: The selected registry events and their attributes

Event	Attribute
Event ID 12: RegistryEvent (Object create and delete)	Event ID
	EventType
Event ID 13: RegistryEvent (Value Set)	Event ID
Event ID 14: RegistryEvent (Key and Value Rename)	Event ID
	EventType

Table 4: The selected network event and attributes

Event	Attribute
Event ID 3: Network connection	Event ID
	Protocol
	Initiated
	SourcePort
	DestinationPort

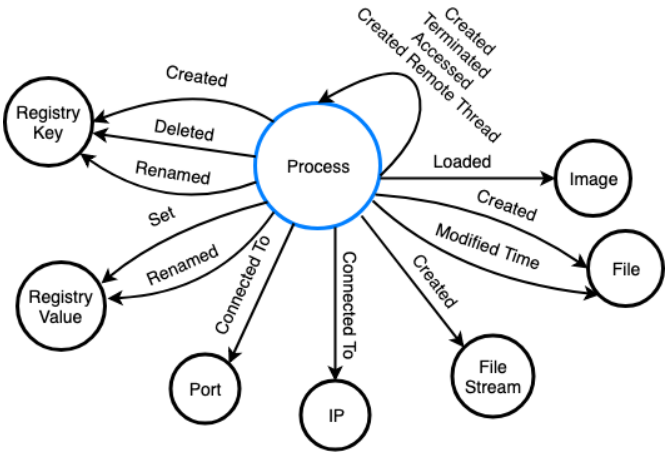

Figure 2: The events of a process

layer x, embedding layer E, GRU layer G, and output layer y, where the proposed model structure is outlined in Figure 3.

Non-numerical input data needs to be encoded in order to feed into the input layer of the proposed GRU classification model. The proposed features belong to one of the following value types: numerical, categorical, and string data. Most of the proposed features are numerical data like event ID; some belong to categorical data; attributes like file name or environment variables belong to string data. The numerical data is straightforward and does not need any encoding. The categorical data is encoded by numbers, where each category is denoted by a different number. One-hot encoding is commonly used for encoding string data but is inefficient in handling sparse data. This study employs label encoding to reduce the dimension embedding and to reduce the processing time.

The input to the input layer is composed of a sequence of the events performed by a process as described above. Let N_{events} be the maximum number of the events to be captured to represent the behavior of a process and N_{attr} be the maximum number of the associated event attributes. All time-series need to have the same length. In order words, the behavior of a process is represented as a matrix of $N_{events} \times N_{attr}$ at the input layer and zero is padded if it is needed to match the required dimension. The GRU classification model applies vector transformation on the embedding layer. The GRU layer consists of N_{events} neurons, matching with the number of the events in a process, and learns the relationship of the events and attributes among benign and malicious processes from the training data.

4 Performance Evaluation

To evaluate if the proposed method can identify malicious behaviors and unknown malware, this study emulated machines compromised by various types of malware. Each injected attack was executed for 5 minutes in a controlled environment, and the system logs were collected during the execution.

10051 malware samples from 36 different malware families were retrieved from Malware Knowledge Base hosted by the National Center for High-performance Computing, where 8889 have been analyzed by VirusTotal, and the rest had not been uploaded or analyzed at the time the evaluation conducted and were considered as unknown malware in this study. A total of 10048 event records were collected from the aforementioned experiments.

The normal behaviors with common benign program execution were collected from a campus network, including Windows system processes and common user processes such as document editing software, web browsing, and other benign applications. A total of 47175 event records were obtained from the benign.

4.1 Performance Evaluation of the Proposed Model

The proposed ML-based detection system aims for analyzing system logs and classify anomalies efficiently. This study adopts accuracy as the performance measurement as classifying benign and malicious behaviors correctly is equally important. Accuracy is expressed below, which is calculated from the confusion matrix summarized in Table 5.

Table 5: The selected network events and their attributes

	Benign (Detected)	**Malicious (Detected)**
Benign	True Negatives (TN)	False Positives (FP)
Malicious	False Negatives (FN)	True Positives (TP)

Table 6 lists the parameter settings of the proposed GRU model, where N_{events} is set to 100 and N_{attr} is set to 100 and Nattr is set to 8; 100 events are extracted to represent the behaviors of a process and the maximum of 8 event attributes from an event. The experiments on 10-fold cross-validation with different ratios (training:

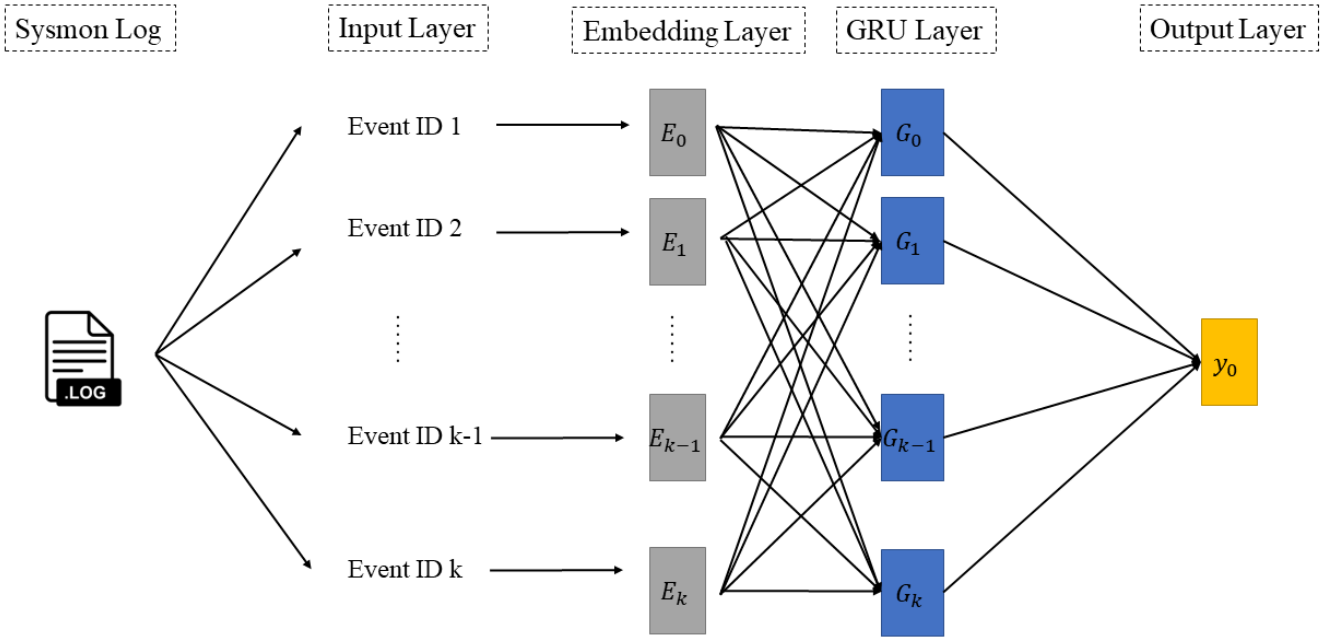

Figure 3: The proposed GRU model

Table 6: System parameters of the GRU model

Layer	Parameter	Parameter setting
Input layer	Input size	(None, ∞)
	Output size	(None,800)
Embedding layer	Input size	(None,800)
	Output size	(None,800,21)
	Mask zero	True
GRU layer	Input size	(None,800,21)
	Output size	(None,100)
	Dropout	0.2
Output layer	Input size	(None,100)
	Output size	(None,1)
	Activation	sigmoid
Other parameter setting	No. of Epochs: 60 Batch size: 400 Loss function: binary cross entropy Optimizer: adam Evaluation: accuracy	

Testing ranging from 2:8, 5:5, to 2:8) were tested and the detection results are plotted in Figure 4. The proposed model gives good accuracy even the training data is 20%, and its performance improves when it has more training data to learn anomalous behaviors.

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

4.2 Performance Comparison with SVM

The literature review informed that SVM performed well on anomaly detection and was selected as the baseline comparison in this study. The kernel function of SVM maps the data to a different space where a linear hyperplane can be used to separate classes. The RBF (radial basis function) is generally used kernel function and achieves a better performance comparing with other kernel functions. The parameters of SVM were optimized during the evaluation in order to build an SVM model with the best detection performance. The parameters of the best SVM model obtained are: the kernel function is Gaussian RBF, $\gamma = 0.00001$, and $C = 1000$, where the hyperparameter γ controls the tradeoff between error due to bias and variance in the SVM model and the hyperparameter C controls the trade-off between the slack variable penalty (misclassifications) and width of the margin. Each experiment was tested by random subsampling 5 times. The results of the performance comparison are outlined in Figure 5 and demonstrate that the proposed GRU model outperforms SVM and classifies both malicious and benign behaviors efficiently.

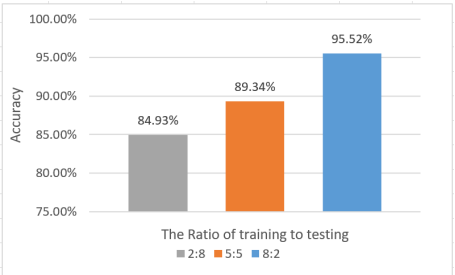

Figure 4: The detection performance on cross-validation

4.3 Performance Comparison with Human Analysis Report

To verify if the proposed system can identify valid malicious events, the generated results were compared

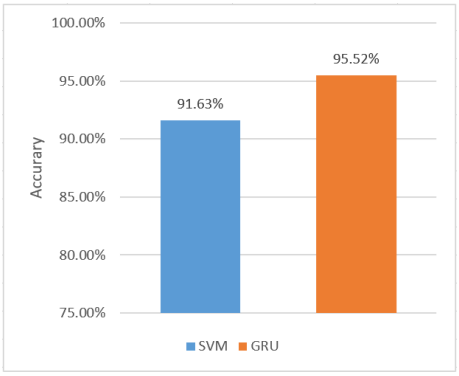

Figure 5: The performance comparison

with an analysis report [21] of PhotoMiner conducted by a security expert. Besides mining cryptocurrency, the malware PhotoMiner (screen.scr) exploited vulnerable FTP servers by password guessing attacks. The human analysis report indicates that the malware invoked cmd.exe to store the mining pool's information (pools.txt) to a temp folder and invoked NsCpuMiner32.exe mining and xcopy.exe spreading it to the disk device of the victim machine. Besides the above behaviors observed by the human analysis, the proposed system could provide additional detail information such as the locations of the suspicious programs and files as shown in Figure 6, where the malware screen.scr created NsCpuMiner32.exe and a html file and spawned a child process (cmd.exe) to create pools.txt at the temp folder.

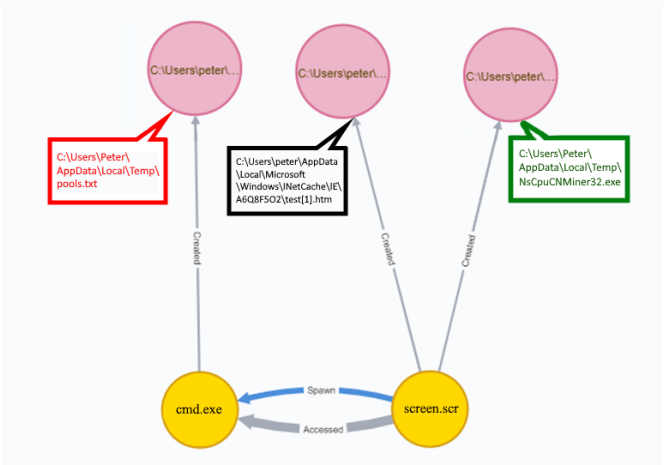

Figure 6: A portion of the detection results on file system

The detection results of the proposed system demonstrate that the malware spawned out many processes and provide a detailed parent-child process relationship. Figure 7 plots the spawned child processes in depth level one, where the red box denotes the malware process (screen.scr), the beige circle on its right indicates its location, the big green one concludes its child processes, each honey colored circle in the green box denotes a spawned child process, and the num-ber on a honey circle represents the locatioin of the child process. It can be seen that the malware spawned many xcopy.exe processes to spread the malware. The relationship of higher depth level can be produced as well.

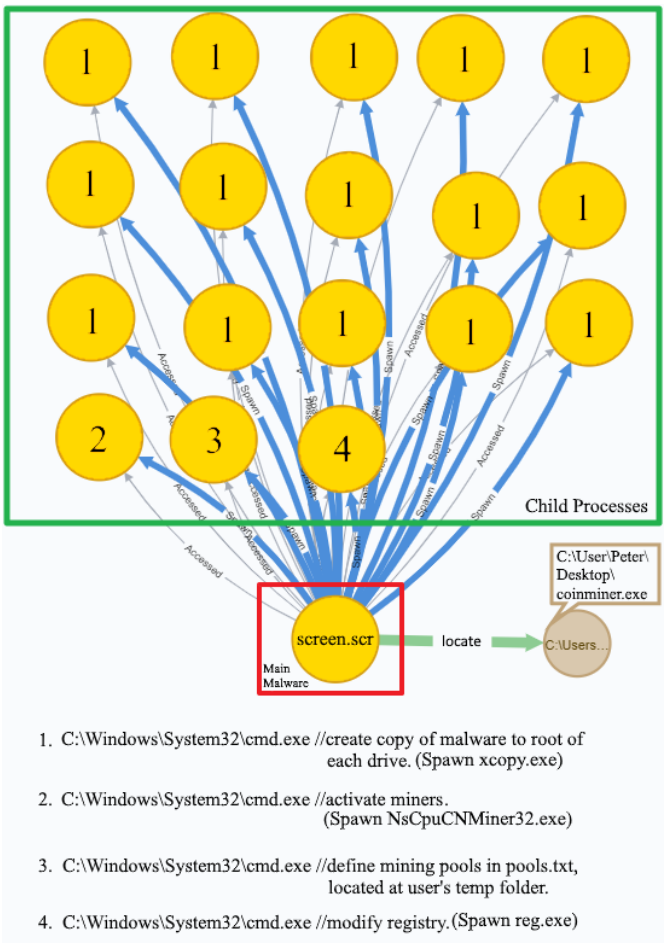

Figure 7: A portion of the detection results on process relationship

As for registry, the human analysis report identifies that the malware invoked cmd.exe executing reg.exe which created a new entry under HKCU\ SOFTWARE\Microsoft\Windows\CurrentVersion\Run in order to auto-start during system reboot. Besides this finding, the proposed system flagged more registry anomalies: network security setting, and multiple registry key values under HKU\ (User)\Software\Microsoft\Windows\CurrentVersion \Internet Settings\ ZoneMap, including ProxyBypass, IntranetName, UNCAsIntranet, AutoDetect, were modified. Because of the massive amount of registry keys, human experts could not identify all the distinguished key values without a valid tool, while the proposed system is useful and could identify registry changes. This comparison concludes that the proposed system could detect suspicious events efficiently.

Table 7: Detection results of unknown malware

Real / Detected	Malicious (Positive)	Benign (Negative)
malicious	97.15%	3.80%
benign	2.85%	96.20%
Accuracy	96.68%	

4.4 Evaluation of Unknown Malware Detection

1162 malware samples had not been analyzed and reported in VirusTotal and were considered as unknown malware. 1072 samples were able to run successfully on a sandbox environment, and a total of 1159 malicious log records was obtained. The same amount of benign behaviors were blended in order to evaluate if the proposed method can classify correctly on unknown malware as well as benign processes. Table 7 lists the detection results and demonstrates that the proposed detection model has a precision of 93.23% and a low false positive rate of 3.8%. The results prove that the proposed solution performs very well in detecting unknown malware.

5 Conclusion

In case of a security attack, it is time-critical matter to identify the cause and to reduce the impact of the damage. Audit logs are a reliable source to discover attacks and should be well-protected to prevent them from being compromised. Many organizations keep their important log files on cloud storage; hence data privacy becomes a concern. Efficient cryptography solutions, such as attribute-based encryption data sharing scheme based on Elliptic Curve Cryptography [5], are suitable for fulfilling the security requirement of cloud storage access.

Most organizations are lack of security experts and manpower to analyze a large number of audit trails and to discover suspicious events. This study proposes a GRU-based detection method that analyzes system logs and identifies malware misbehaviors.

The experiments emulated the attacks as well as normal usages in real-world environments. The experimental results indicate that the proposed solution classifies both benign and malicious behaviors efficiently, identifies unknown malware well, and outperforms SVM detection. In summary, the performance evaluation demonstrates that the proposed machine learning model is practical and efficient. Future research could enhance log analysis and anomaly detection by including additional log files and expertise knowledge to improve detection performance.

References

[1] H. H. T. Albasheer, M. M. Siraj and M. M. Din, "Towards predictive real-time multi-sensors intrusion alert correlation framework," *Indian Journal of Science and Technology*, vol. 8, no. 12, 2015.

[2] M. Amini, J. Rezaeenoor, and E. Hadavandi, "Effective intrusion detection with a neural network ensemble using fuzzy clustering and stacking combination method," *Journal of Computing Security*, vol. 1, no. 4, pp. 293-305, 2014.

[3] S. D. D. Anton, S. Sinha, and H. D. Schotten, "Anomaly-based intrusion detection in industrial data with svm and random forests," *Cryptography and Security*, 2019. (`https://arxiv.org/abs/1907.10374`)

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *Computation and Language*, 2014. (`https://arxiv.org/abs/1406.1078`)

[5] M. A. Doostari, S. Rezaei and M. Bayat, "A lightweight and efficient data sharing scheme for cloud computing," *International Journal of Electronics and Information Engineering*, vol. 9, pp. 115–131, 2018.

[6] J. Dwyer and T. M. Truta, "Finding anomalies in windows event logs using standard deviation," in *The 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pp. 563–570, 2013.

[7] Infosec Institute, *Common Malware Persistence Mechanisms*, Technical report, 2016. (`https://resources.infosecinstitute.com/common-malware-persistence-mechanisms/#gref`)

[8] T. Y. Kim and S. B. Cho, "Web traffic anomaly detection using C-LSTM neural networks," *Expert Systems with Applications*, vol. 106, pp. 66–76, 2018.

[9] C. C. Lo, C. C. Huang, and J. Ku, "A cooperative intrusion detection system framework for cloud computing networks," in *The 39th International Conference on Parallel Processing Workshops*, pp. 280–284, 2010.

[10] Microsoft, *Structure of the Registry*, Technical report, 2018. (`https://docs.microsoft.com/en-us/windows/win32/sysinfo/structure-of-the-registry`)

[11] N. Mishra A. Tayal and S. Sharma, "Active monitoring & postmortem forensic analysis of network threats: A survey," *International Journal of Electronics and Information Engineering*, vol. 6, pp. 49–59, 2017.

[12] F. Nabi and M. M. Nabi, "A process of security assurance properties unification for application logic," *International Journal of Electronics and Information Engineering*, no. 6, pp. 40–48, 2017.

[13] V. Nigam, *Understanding Neural Networks. From Neuron to RNN, CNN, and deep learning*, Technical report, 2018. (`https://towardsdatascience.com/understanding-neural-networks-from-neuron-to-rnn-cnn-and-deep-learning-cd88e90e0a90`)

[14] P. Prasse, L. Machlica, T. Pevný, J. Havelka, and T. Scheffer, "Malware detection by analysing network traffic with neural networks," in *IEEE Security and Privacy Workshops (SPW'17)*, pp. 205–210, 2017.

[15] B. J. Radford, L. M. Apolonio, A. J. Trias, and J. A. Simpson, "Network traffic anomaly detection using recurrent neural networks," *Computers and Society*, 2018. (https://arxiv.org/abs/1803.10769)

[16] E. Raftopoulos and X. Dimitropoulos, "IDS alert correlation in the wild with EDGe," *IEEE Journal on Seleted Areas in Communi-cations*, vol. 32, no. 10, pp. 1933-1946, 2014.

[17] N. Serketzis, V. Katos, C. Ilioudis, D. Baltatzis, and G. Pangalos, "Towards a threat intellegence informed digital forensics readiness framework," in *Twenty-Fifth European Conference on Information Systems (ECIS'17)*, 2017. (http://eprints.bournemouth.ac.uk/30391/)

[18] A. Singhal, C. Liu and D. Wijesekera, "A model towards using evidence from security events for network attack analysis," *International Workshop on Security in Information System*, pp. 83–95, 2014. (https://doi.org/10.5220/0004980300830095)

[19] Statcounter, *Desktop Operating System Market Share Worldwide*, Technical report, 2020. (http://gs.statcounter.com/os-market-share/desktop/worldwide)

[20] Thycotic, *Black Hat 2018 Hacker Survey Report*, Technical report, 2018. (https://go.thycotic.com/l/101722/2018-09-12/5gf8wq/101722/74015/Report_2018_Black_Hat_Survey.pdf?_ga=2.52829416.1772536159.1560412381-274843393.1560412381)

[21] Taiwan Academic Network Computer Emergency Response Team, *Incident Analysis Report of Miner Trojan Photominer Infecting Campus Machines*, Technical report, 2017. (https://portal.cert.tanet.edu.tw/docs/pdf/201709290109555585771228906051.pdf)

[22] H. Wang, J. Gu, and S. S. Wang, "An effective intrusion detection framework based on svm with feature augmentation," *Knowledge-Based Systems*, vol. 136, pp. 130–139, 2017.

[23] S. T. Zargar, H. Takabi, and J. B. D. Joshi. "DCDIDP: A distributed, collaborative, and data-driven intrusion detection and prevention framework for cloud computing environments," in *The 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom'11)*, pp. 332–341, 2011.

[24] S. Zidi, T. Moulahi, and B. Alaya, "Fault detection in wireless sensor networks through SVM classifier," *IEEE Sensors Journal*, vol. 18, no. 1, pp. 340–347, 2017.

Biography

Chia-Mei Chen has joined in the Department of Information Management, National Sun Yat-Sen University since 1996. She was a Section Chef of Network Division and Deputy Director, Office of Library and Information Services in 2009-2011. She had served as a coordinator of TWCERT/CC (Taiwan Computer Emergency Response Team/Coordination Center) during 1998 to 2013 and established TACERT (Taiwan Academic Network Computer Emergency Response Team) in 2009. She had served as the Deputy Chair of TWISC@NCKU, a branch of Taiwan Information Security Center, for three years. She continues working for the network security society. Her current research interests include anomaly detection, malware analysis, network security, and cyber threat intelligence.

Gen-Hong Syu was a graduate student at the Department of Information Management, National Sun Yat-sen University. He is interested in digital forensics.

Zheng-Xun Cai received master degree at National Sun Yat-sen University, Kaohsiung, Taiwan in 2017. Now he is PhD student in National Sun Yat-sen University, Kaohsiung, Taiwan. His research interests involve digital forensics, network analysis and process analysis.