Insider Threat Detection with Long Short-Term Memory

Jiuming Lu
School of Computer Science and Engineering
University of New South Wales
Sydney, New South Wales
z3343727@student.unsw.edu.au

Raymond K. Wong
School of Computer Science and Engineering
University of New South Wales
Sydney, New South Wales
wong@cse.unsw.edu.au

ABSTRACT

Most organizations these days are increasingly threatened by malicious insiders. The traditional cybersecurity system uses historical logs to investigate/prevent attacks from outside a company. However, for insider threats, new models and techniques are required to differentiate normal behaviour from malicious acts. This paper proposes a system, called Insider Catcher, that bases on a deep neural network with Long Short-Term Memory (LSTM) to model system logs as a natural structured sequence. Our system captures patterns that indicate users' normal usage behaviour to differentiate normal behaviour from malicious acts. Experiments show the superior performance of the proposed system over the existing log-based anomaly detection strategies. This is particularly for real-time online cases.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**.

KEYWORDS

Anomaly detection; insider threat; long short-term memory

ACM Reference Format:
Jiuming Lu and Raymond K. Wong. 2019. Insider Threat Detection with Long Short-Term Memory. In *Proceedings of the Australasian Computer Science Week Multiconference (ACSW '19), January 29–31, 2019, Sydney, NSW, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3290688.3290692

1 INTRODUCTION

The insider threat is widely recognized as a challenge faced by large enterprises. How to detect or find such kind of threats from a massive amount of company's historical event log records is a crucial issue. According to The Breach Level Index [9], nearly 9 % of data breaches are caused by insider threats. Moreover, this number is continuously increasing. It will become another vital cybersecurity issue that every company needs to deal with.

Anomaly detection problems have always been taken seriously in cybersecurity protection [32]. It is an essential procedure that needs to be considered in the development of cybersecurity systems. Especially for vast enterprises, security is under challenge every day. The Internet is exploded online in which there are a plethora of data leaking incidents. The more reputable a company is, the more likely it is to be attacked. Certain valuable information can be found attractive for attackers. Researchers have done great research on the prevention of intrusion detection from Internet attacks [32]. Virus scanning software is installed in case virus or torrent is accidentally installed on enterprise computers by emails or users [8]. However, the insider threat is paid more and more attention in recent years.

In comparison with traditional threat events, the insider threat is more complex. It is more difficult to define and detect [3, 26]. Users may not break their authorities but still cause profit loss to the company. The insider threat can be classified into three types [26, 31]. Firstly, the users do not use their computers or remote devices properly exposing the company to the risk of threats. Secondly, information or valuable data may be leaked and stolen. The third problem is the cross authority by which employees log into other's computer for personal purpose. All these insider threats happen in the circumstances that a user is usually granted certain rights to use a company's resources. The use of company computers and the access to confidential data are routinely seen as normal computer usage actions. This is the reason why their improper actions are not easily identified by traditional cybersecurity protection methods.

Recent years, researchers make great contributions to big data analysis. Relevant techniques are widely used in the area like energy industry, health-care, supply chain, finance and cybersecurity [11]. To solve insider threat problems, user and entity behaviour analytics (UEBA) was introduced in recent years. Inspired by these ideas, machine learning methods can be used to analyze users' behaviour, hence increase the performance of the detection of insider threat.

1.1 Challenge

System event logs are used for insider threat detection. Practically log files exist on every computer or server. They are kept for the purpose of debugging and system diagnosis. The availability and usability usually result in log files having a massive size accumulated from time to time. It is a time consuming progress to analysis such a large amount of data source [23]. How to extract useful information and shrink the dataset to smaller size is of paramount significance. Although log data can be easily obtained, most data can not be directly used for analysis [7, 13]. Most of the log data is unstructured and log files may be recorded in different formats [6]. Accordingly, log data files cannot be directly put into machine learning models because that they are not numerical [12, 16]. To overcome this challenge, we need to deploy log data parsing. Log files contain rich information such as timestamps, different ID and recorded texts. Most of the existing analysis methods solely using

ACSW '19, January 29–31, 2019, Sydney, NSW, Australia
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6603-8/19/01…$15.00
https://doi.org/10.1145/3290688.3290692

timestamps or log key in their training process [6, 7, 13], which limit the ability of machine learning models.

Then the log records need to be sequenced timely [7]. In this paper, the core method is the LSTM network, which requires the input data following the time sequence. Given that, the input data are sourced from different log files and in different formats, we are required to manage and reorganize them to make them consistent before indexing a sequence by time.

Afterwards, we attempt to overcome the last challenge-insider threat detection. The insider threat is more difficult to find out for some special reasons [17, 31]. Under normal circumstance, users are empowered with the authority to use enterprise computers and recourses, and their actions are just basic computer usage activities. These are not characterized by some particular behavioural patterns that can easily be recognized or traced. An enterprise may omit to detect some insider threats arising from the act of the internal users in the course of their routine operation. How to detect insider threat events from users' normal operation activities is a problem.

1.2 Contribution

Our proposed system uses a Recurrent Neural Network (RNN) as the core model [22]. This machine learning method can use a recursive structure to save and forward the output of current training stage to next input layer. It allows the system to be influenced by behavioural patterns of the new record on top of behavioural patterns of the historical records. The LSTM is one of the most efficient versions of RNN neural networks [10, 30]. The LSTM remembers long-term dependencies instead of tracking the influence of recent records, which can be accomplished by RNN. Moreover, LSTM now has been widely used in machine translation [30], anomaly detection [21] and medical self-diagnosis [19].

LSTM may help us to figure out the potential relationships among two or few adjacent records as a matter of the fact that many user computing records are well-structured in a form of log sequence. The Insider Catcher is a well-designed system, which can do online anomaly detection with LSTM. It also does insider detection with work-flow models. It can examine individual computing records and analyze the user behaviour for each employee. With the help of big data analysis, it can recognize the log sequences for each user's normal working hours and apply it for online anomaly detection over the incoming entries. As a whole, our system is capable of building a neural network model based on the records of previous cyber threats and can be used to capture different types of insider threats.

We also develop a data process strategy that transforms different log records into sequence blocks for insider threat analysis. It also extracts useful information and replaces the redundant records from the dataset. By doing so, we reduce the total size of the dataset by 70%, which can increase the speed and accuracy of our system remarkably.

2 PRELIMINARY

2.1 Data

The dataset used in this paper is called CERT Insider Threat Dataset [4]. It was collected from a real enterprise and contains some insider threat scenarios with sensitive information hashed for privacy reason. It includes daily computer usage records of 4000 users for a period of 1.5 years and consists of a total of 135 million records in 5 different categories: logon data, device data, HTTP data, email data and file data. Each category is briefly described as follows:

- [logon.csv:] id, date, user, PC, activity (logon / logoff)
  - Weekends and statutory holidays are included as long as there are computing records on those days.
  - No user is able to log onto a machine where another user is already logged on and has not logged off yet.
  - Some logons may occur after-hours. The normal working hour of an employee is determined by his/her duty and position.
  - A total of 4000 users are recorded and a unique computer is assigned to each user. There are also 400 shared machines in the computer lab and can be publicly accessed.
  - Different employees may have different workloads (i.e., work hours), but each of them usually starts and ends his/her work around the same time every day, with occasionally after-hour work.
  - Some employees may leave the organization, which will show in the log that no more logons are recorded from the day of termination.
  - Some users occasionally log onto others' computers.
- [device.csv:] id, date, user, PC, file-tree, activity (connect / disconnect)
  - Some users use removable devices such as USB thumb drives.
  - Disconnect events may be missing, since users may power off the computer before removing the removable devices.
  - Most users have a regular usage pattern (time, duration, frequency) when using removable devices. Deviations from these patterns shall be particularly considered.
- [http.csv:] id, date, user, PC, URL, activity, content
  - The words contained in a URL are usually related to the web page topics.
  - The content field consists of the list of content keywords (can be from multiple topics).
  - Activity can be "WWW Visit", "WWW Download" and "WWW Upload".
- [email.csv:] id, date, user, PC, to, cc, bcc, from, activity, size, attachments, content
  - An email may have multiple receivers that can be employees, non-employees or both.
  - Employees are expected using company's email addresses, however some employees also use their personal emails.
  - Email may have attachments, but size does not include attachments.
  - Email content consists of a list of content keywords.
  - Activities can be Send, Receive or View.
- [file.csv:] id, date, user, PC, filename, activity, to removable media, from removable media, content
  - File activities (open, copy, write, delete) will involve files from a removable device.
  - Content contains a hexadecimal encoded file header and a list of content keywords.

– Most employees perform regular file operations everyday. Deviations can be considered significant.

2.2 Data Parsing

To transform this dataset into machine-readable data, a special data washing and management procedure is designed. Also, useful information need to be extracted to improve accuracy and training speed. The design transforms these heterogeneous tracking sources into a numeric form which is suitable for training. The idea is to give different user actions a unique numeric label. Then user action matrices can be generated according to these labels.

Here shows the detail of data labeling process. In order to distinguish each user action, time is an important character to be considered. However, considering each user separately by time will tremendously increase the training cost. To solve this problem, we divide a whole day into four time periods which are 00:00 to 06:00, 06:00 to 12:00, 12:00 to 18:00 and 18:00 to 24:00. Different time weights W_t are set to each user record. Then each action is given a unique action weight A_{t1}. If a particular action has a subcategory, it will be assigned with another action weight A_{t2}. Then all the weights are multiplied together to generate a unique user action label. The detail of how to distinguish each user action are shown in Figure 1. For each data category, we give it a label section according to how many different user actions in each category. For example, the action in the logon dataset is given a label through 1 to 8, which dues to the eight different user actions in the logon dataset. Continuously, the device dataset is given a label through 9 to 16. Because each user is a single component in the Internet environment, all user actions are grouped by their user IDs and sorted by time. As shown in Figure 1, after the first round of data process, all the actions are separated into 187 different action catalogs with unique labels. Yuan et al. [41] also use labeled datasets to perform LSTM modeling to detect insider threats. However, they only use 16 unique labels. Moreover, to increase the detection accuracy, they manually label the data based on key features of different insider threat scenarios. Their methods are not total automatic detection.

For each user, LSTM blocks of user actions are generated accordingly. For different n size of blocks, we firstly take one user action record X_n and then follow by $n - 1$ previous user actions $X_1, X_2, ..., X_{n-2}, X_{n-1}$ to generate the whole n size blocks. Then we take the $n + 1$ user action X_{n+1} and all its $n - 1$ previous action records to generate the next user block. These blocks will be used for the LSTM training process to simulate a user's online usage behaviour, hence to do anomaly detection.

2.3 Statistics

The dataset used in this paper is CERT Insider Threat Dataset [4], which is published by Carnegie Mellon University. This dataset was collected from a real-world enterprise with real insider threat cases and it is the latest version V6.2. In this paper, the insider threat detection focuses on large enterprises. In order to protect the security and privacy of the data source, all data used has been undergoing an anonymization progress. Real usernames and sensitive information have gone through certain hash functions and have been masked. The detail of the dataset has been discussed in the previous section. For the experiment purpose, the dataset has been put into two different sections: the training part and the testing part. In order to verify the efficiency of the model, 85% of the data is used as a training set and the rest is to be a testing set. The dataset contains about 135 millions data records. More specifically, it records 4000 employees' computer usage events in 1.5 years, which is 516 days. Table 1 shows the details of the combination of training and testing datasets.

Table 1: Data Description

	Training	Testing
Date Range	Days 1-400	Days 401-516
#Device Events	1,264,337	287,491
#Logon Events	2,556,721	973,564
#Http Events	89,526,358	27,498,858
#Email Events	8,692,518	2,302,439
#File Events	1,325,412	689,471
#Totle Events	103,365,346	31,751,823
#Threat Events	217	211
#Threat User-Days	26	21

3 INSIDER CATCHER ARCHITECTURE

Big data analysis and deep neural network have been drawing attention in recent years. It is widely used in classification and intrusion detection areas. Our system uses a neural network to generate users' Internet usage behaviour patterns by analyzing daily online behaviour. The system can continuously monitor a user's Internet usage status and find anomaly event records.

The model used in this paper is RNN because the input data source is continuously and structured, there may be certain relationship between the previous records and the current records. Moreover, LSTM method was used for anomaly detection in this paper.

Our system includes two parts. The first part is historical user's computing usage behaviour analysis, which is based on a particular user's historical log records to train for behaviour patterns of each user. It will distinguish anomaly user actions from the normal ones according to each user's behaviour. The second part is the instance online monitoring detection process. It records most recent normal log records and uses an LSTM model to predict the next action of the current user. If an abnormal action has been predicted or the predictions have been continuously different from the real actions, it or they would need to be investigated.

3.1 Baseline Model

In order to test the effectiveness of the Insider Catcher system, we use several famous anomaly detection methods to test our system. By comparing the performance and cost of accomplishing the detection, it gives us a basic and convincing result to illustrate whether our system is competitive with the existing ones.

Decision Tree is one of the effective methods used by virus detection systems [29]. However, it is based on a huge amount of experience and knowledge from security expert to make cybersecurity rules. It is hard for us to verify its effectiveness for lacking of support.

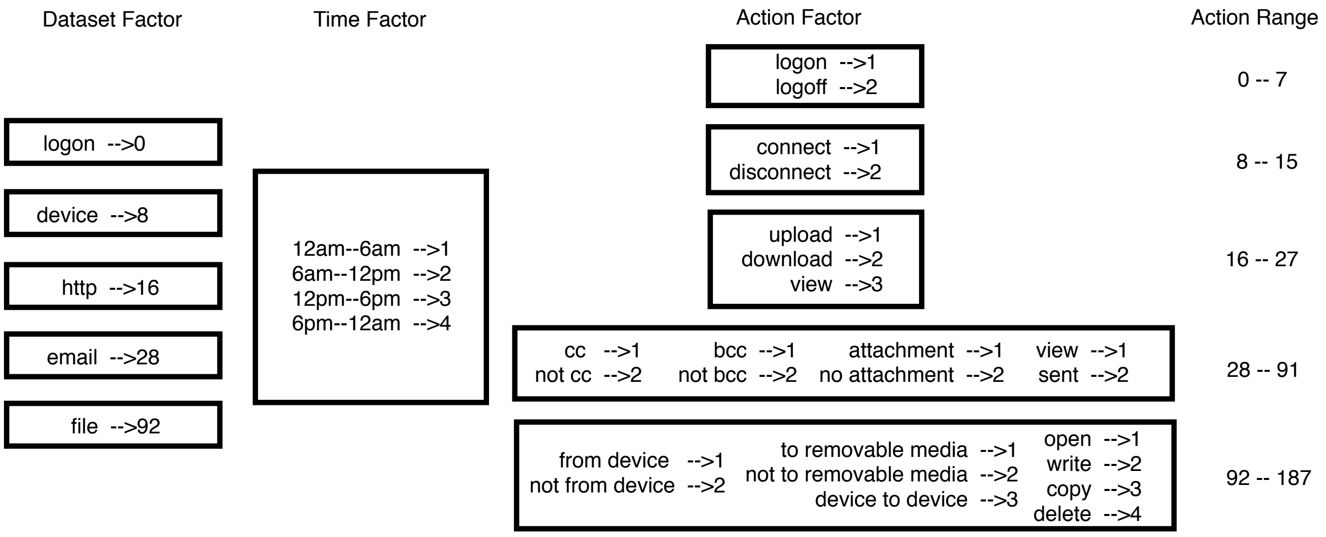

Figure 1: Labeling of User Actions

The Principal Component Analysis (PCA) is also a simple but efficient way for anomaly detection [2, 37]. The method firstly builds up data matrices, which can represent the summary of input data within a time period. The matrices is constructed by sessions. The session matrices may contain a large number of elements where each element will represent different types of user actions. In addition, the value of the element represents the total accumulation of such user action within a session. For different sessions, the PCA calculates the projection length of the matrices vector on the residual subspace of the coordinate system. By computing the length difference between normal and anomaly records, anomaly detection is accomplished.

The support vector Machine (SVM) is also a supervised learning method for classification problem [15]. This method works effectively when it gets multiple dimensions of a dataset. Given a set of the training sample with labels telling their categories, SVM training algorithm maps the sample into a space and the samples from different categories will be divided by a gap. The algorithm guarantees the gap to be a maximum margin and hence it clearly distinguish one from the other categories. Also, the new sample can be mapped into the space to tell which category it belongs to. For anomaly detection, SVM can separate suspect user actions from the normal ones with the historical log dataset which records the detail of enterprise computing system under threatening records.

3.2 LSTM for Anomaly Detection

As discussed in the previous section, the input of LSTM model is in a format of sequence blocks $[X_1, X_2, ..., X_{n-2}, X_{n-1}, X_n]$. During the training process, hidden state vectors $[h_1, h_2, ..., h_{n-2}, h_{n-1}, h_n]$ are generated to illustrate the potential relationship between each input element. Unlike normal deep-learning neural networks which only discover the computed hidden state vector as a function of the current input sequence, LSTM has memory to generate hidden state vectors as well according to previous inputs. A more complex hidden state function is generated by introducing forget gate vectors f_t, input gate vectors i_t and output gate vectors o_t. So, after each hidden layer, combined with the hidden state vectors, a long-term memory cell C_t will be generated to remember the influence of previous inputs. In order to increase the ability of the training model, multiple hidden layers will also be used. The detail is shown in the following section:

$$f_t^u = \delta(W_{(f,x)}x_t + W_{(f,h)}h_{t-1} + b_f) \quad (2)$$

$$i_t^u = \delta(W_{(i,x)}x_t + W_{(i,h)}h_{t-1} + b_i) \quad (3)$$

$$o_t^u = \delta(W_{(o,x)}x_t + W_{(o,h)}h_{t-1} + b_o) \quad (4)$$

$$C_t^u = f_t \odot C_{t-1} + i_t \odot \delta_c(W_c x_t + U_c h_{t-1} + b_c) \quad (5)$$

$$h_t^u = o_t \odot \delta_h(c_t) \quad (6)$$

Figure 2 gives a brief description of how the system is trained. Each LSTM block reflects the status of the system. The status is influenced both by the previous and current input data. Also, the current status is passed to next LSTM block. The strength of the influence is decided by the remember weights and the forget weights. This procedure illustrates how the historical data is maintained and passed to LSTM blocks.

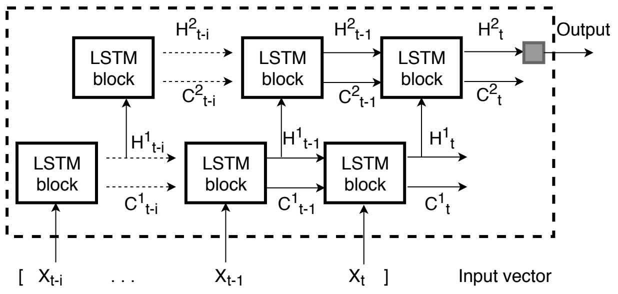

Figure 2: LSTM Model

Each LSTM block maintains a hidden vector H_t and a cell state vector C_t. Both two vectors are passed to the next LSTM block to initialize a state. Within a single LSTM block, gating functions are used to decide how much the previous cell state influences the current state, how the influence from previous outputs and current inputs are balanced and how the new output with all these factors are generated. As shown before, the gating function is constructed with remember weights, forget weights and output weights. These parameters control the amount of information to be kept from both previous output and input data. They also control the information to be passed to the next step. After the training process, the system will find the best model to maximize the probability predicting a user action x_t that follows the user's input sequence. It maximizes the probability of the predicted user's next possible action to match a user's real action.

3.3 LSTM Insider Threat Detection System

LSTM model is trained based on employees' online behaviour to predict a user's next possible action. As long as the prediction and the real user action do not have a significant difference, we can argue that the user follows his or her normal online behaviour and the action is normal.

However, if we do not allow any tolerance, the precision and recall are not very satisfied, which are simply 14% and 54% respectively. The system can minimize the loss between a prediction value and a real value, hence predicts employee's next possible actions. However, each user action category may contain several subcategories with similar labels. For example, the label for employee writing emails from 12 pm to 6 pm is 71 whereas that for viewing emails is 72. There is small variance between similar actions. Also, the system generates a prediction to illustrate which action is most likely to be the next movement. In most of cases, about 20% of the chosen one exactly matches the real action and several prediction choices are generated with similar probabilities. Moreover, the predictions are similar in certain aspects. It is hard to generate correct predictions continuously, especially in a long run.

In order to increase the accuracy of the prediction, we introduce a tolerance factor g. If the real user action is within the top g most likely predictions, we conclude that the prediction is correct. Otherwise, we deem the action as an anomaly action and record it for further analysis.

As mentioned in the previous section, anomaly events may not be an insider threat. A further insider threat detection procedure is needed. After completing the anomaly detection procedure, all the records for potential threat event will be stored and passed to the insider threat analysis. During the analytical process, the total user actions in each designed period will be calculated. Out of them, the percentage of anomaly records over total actions will be calculated. These records will be compared with users' historical records before the identification of insider threat.

4 EXPERIMENTS

To test the effectiveness of this model, several experiments are done based on the machine learning python package TensorFlow [1] [1]. The details of the dataset, parameter setting, result and analysis will be explained in this section. Three case studies, which focus on different insider threats, will also be accomplished after experiments to demonstrate how the real-world scenarios work.

In this paper, LSTM machine learning strategy is used as the main algorithm for our insider detection system. The parameters are tested and tuned for the best performance. The number of hidden layers is set to 5 and the number of hidden units is set to 50. Learning rate is proposed as 0.006 whereas time step is defined as 20 for backpropagation. Other parameters and settings will be discussed in the following section.

We design several experiments to verify the efficiency of Insider Catcher. Each experiment figures out the factor or the performance of the system. The details are discussed in two separate parts below which are the anomaly detection and the online insider threat monitoring.

4.1 Anomaly Detection

We firstly verify the LSTM anomaly detection model and the result is shown in Figure 3. The red line represents the user's real action while the blue line represents the system predictions. It is clear that the predictions tend to match with the user's real actions. As long as the prediction and the user's real action do not deviate significantly, we can confirm that the user follows his or her online behaviour and the action is normal. To evidence this, Figure 3 shows that the LSTM represents the user's normal online computer usage behaviour and predicts potential anomaly events.

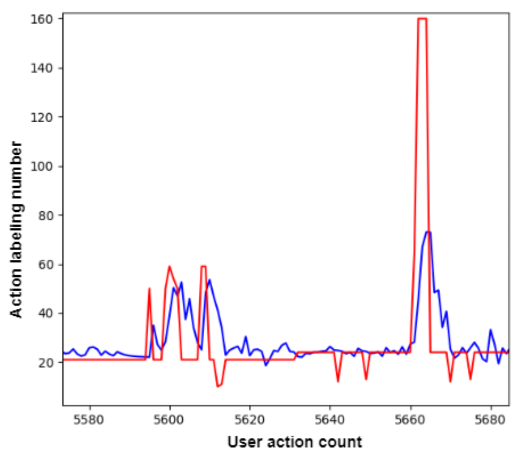

Figure 3: Anomaly Detection

we compare Insider Catcher with the traditional anomaly detection methods like PCA and SVM. We use precision and recall to indicate the effectiveness of each system. The results are shown in Figure 4.

From the plot, the precisions of the three methods are relatively satisfied in the way most anomaly events identified by the systems are related to insider threats. However, the recalls of PCA and SVM are out of our expectation. Low recall means the system cannot find out all or most of the insider threat events. This implies that

[1] https://www.tensorflow.org

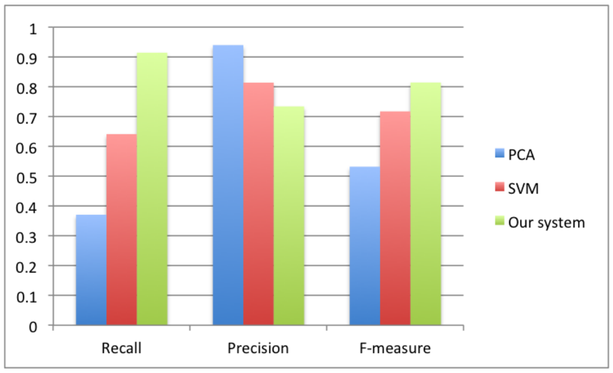

Figure 4: Method Comparisons

the accuracy of the strategy is in a decent level but still needs to increase the ability of the system. The f-measure is the harmonic mean of the precision and recall. It can be used for verifying the effectiveness of the system. From the plot, it is shown that Insider Catcher has the highest f-measure when compared with the other two.

4.2 Insider Threat Detection

The insider threat detection analysis will be accomplished after the anomaly detection and the analysis result is illustrated in Figure 5.

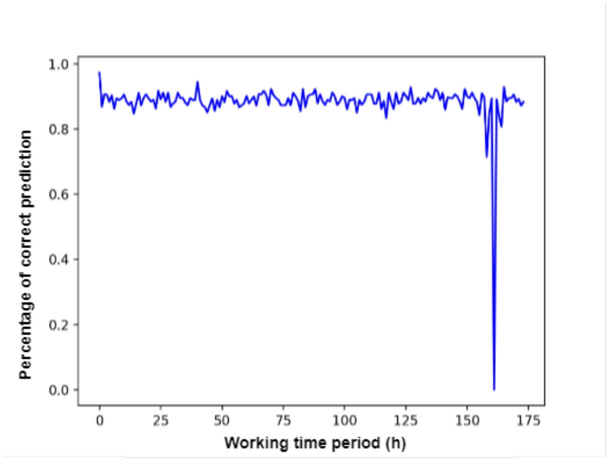

Figure 5: LSTM Insider Threat Detection

From Figure 5, it can be seen that during normal computing usage sessions, most of the users' actions are similar to the predictions, users follow their normal computer usage behaviour and there is no insider threat taken place. However, there is a sudden drop on the percentage of correct prediction when the working time period is close to 162. It means a series of anomaly events is detected. This implies that the user may act in an unusual way which is divergent in comparison with their previous normal behaviour. Moreover, in most of the cases, there may be some insider threats take place. To verify this, historical event records can be checked to verify whether the user did anything harmful to the company.

Other researchers like Aaron et al. [35] also use LSTM to detect insider threats, however, their systems can only find the suspected events related to insider threats. They cannot confirm whether the events are an insider threat or not. Neither can they distinguish which kind of insider threats is detected. They think the cost of a missed detection is substantially larger than that of a false positive. The idea is reasonable, and the results shown in this paper give a good improvement to their work.

We also introduced another two important parameters, the tolerance factor g and the input block element size n, to increase the accuracy of the system. By applying different tolerance factors g, the precision and recall can be shown in Figure 6.

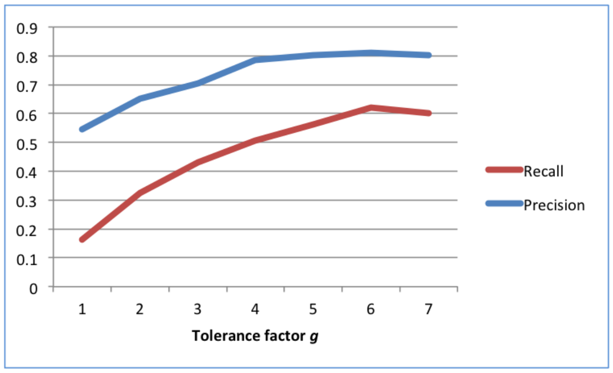

Figure 6: Precision & Recall for Different Tolerance

From Figure 6, both precision and recall increase with the tolerance factor g. When the tolerance is 6, the recall reaches 0.60 whereas the precision is as high as 0.84. What tolerance level is to be set up by enterprises vary to a great extent primarily dependents on their cybersecurity environment and company policies. This needs to be passed to cybersecurity specialists for consideration from the information security point of view.

Another factor to be discussed is the input block element size n. This factor represents how many employees' previous actions are taken into consideration to predict his or her next action. In testing the influence of this factor, we take n from 1 to 10 by increasing one more previous action into consideration. The result is shown in Figure 7.

From the plot, it can be seen that when we increase the block element size n from 1 to 4, the loss of the test drops rapidly. At the same time, the accuracy of the test increases slowly. After n reaches 4. the loss almost remains the same regardless of the number of actions taken into consideration. However. the accuracy of the test continually increases while n increased from 4 to 8. Afterward, the accuracy starts to decrease. However, subsequent changes to the accuracy by verifying n are limited. The possible explanation is that the LSTM method has already got the function to memorise historical records influence. By introducing more historical data does not help too much on increasing the performance of the model.

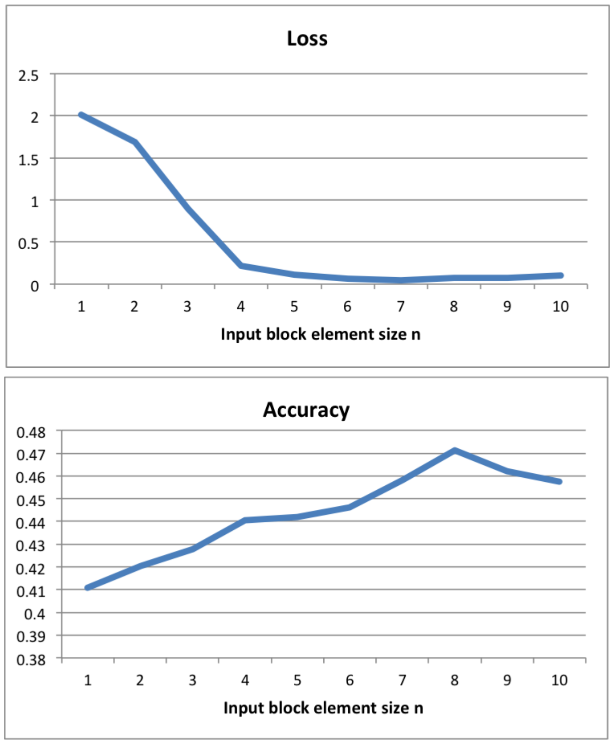

Figure 7: Verifying Block Size

5 CASE STUDY

To illustrate what happens in the real world, several case studies are simulated. Utilizing realistic scenarios is more persuasive to verify the performance of Insider Catcher.

5.1 Case One

The first case is that an employee trying to access the company's computer after work hours. He tries to copy some data files from a company's computer to his own removable storage device for personal use at home. By analyzing the historical records of this employee with Insider Catcher, the result is shown in Figure 8.

From the figure, it can be seen that the percentage of correct prediction is between 70% and 100% during the whole process of prediction. On average, the value is 80% indicating that the user computing behaviour is not as stable as other users. Around the time period 3000, the correct prediction rate suddenly drops below 30%. After that, the average correct prediction rate also drops too. This highlights that some insider threats may occur. By tracing back to the original records, some actions of the user are noted. Details are shown in Table 2.

It is a fact that several anomaly actions were taken place during non-working hours. At around 01:30 the employee tried to access several files from the company's computer and downloaded them to his own removable device. By analyzing this employee's historical records, it can be found that the user seldom uses his own removable devices during working hours. Found out these anomaly user

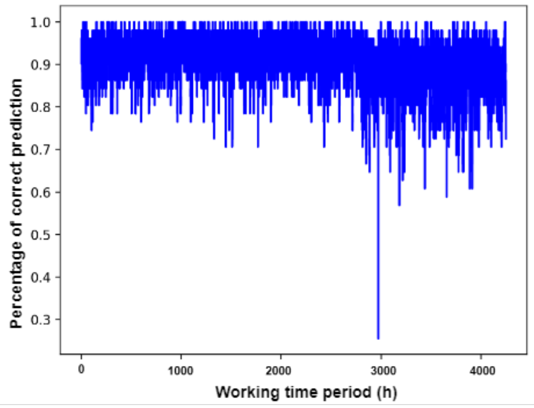

Figure 8: Case One

actions, this employee could be reported to the security staff or company's manager for investigation. Certain warning messages could also be generated to stop any further offense. On the condition that the average correct prediction rate for this user is relatively low, this user should fall into the scope of the potential insiders who needs to be closely monitored by the company on an on-going basis to ensure that their computing using patterns are normal and any abnormal behaviour is timely detected.

5.2 Case Two

The second case study is that the insider logged into other employee's computer searching for interesting files. Then he/she posted the file onto a public website and emailed them to his/her personal email box. This data leakage incident happens frequently in companies bringing tremendous damages to enterprises. The analytical result is shown in Figure 9.

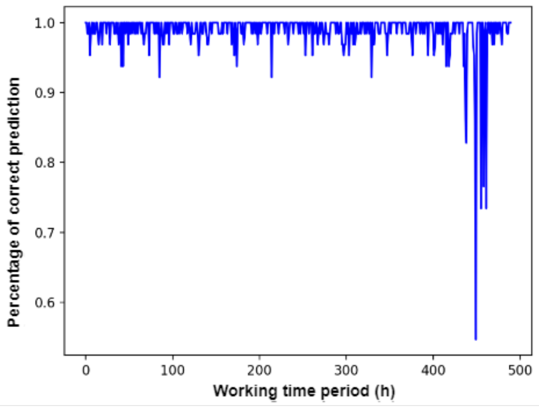

Figure 9: Case Two

From the plot, it can be seen that the user maintains a relatively high correct prediction rate, which most of the cases are over 90%.

Table 2: Case One Anomaly Actions

Action ID	Time Stamp	Action	Fill Access
F0N3-C5GS70QE-4575NRWF	08/18/2010 22:59:20	Connect	R:/;R:/52G6677;R:/782bxm8;R:/ACM2278
L8U4-E1KC17HI-2295QTAV	08/19/2010 01:34:19	File Open	R:/52G6677/B7RGYJZC.jpg
Z8S0-I9WK82HT-2702OXHW	08/19/2010 01:37:20	File Open	R:/52G6677/B7RGYJZC.jpg
M3U7-F9ZW51CX-5408MNTW	08/19/2010 01:38:10	File Open	R:/52G6677/7QO6RIRR.txt
I1R8-Z2OT23MQ-3906SEYL	08/19/2010 01:46:04	File Open	R:/ACM2278/QP8YHH52.zip
E7T1-Z4BN74BJ-2108LJAD	08/19/2010 05:23:05	Disconnect	

It indicates that the user usually behaves the same way during the computing usage. However, after time period 400, the accuracy rate starts to drop, and it maintains at a low level for a while. This indicates that the user has made a series of anomaly events. By checking the records of his historical log, the user once continuously logged into other employees' computers to search useful files for private use.

5.3 Case Three

The third case study shows that instead of working on his or her own jobs, the employee was browsing the Internet and trying to find new job opportunities. The employee also used a removable storage device to store sensitive data at the following days, which data may be traded for profit from competitors of the current company. The analysis result is shown in Figure 10.

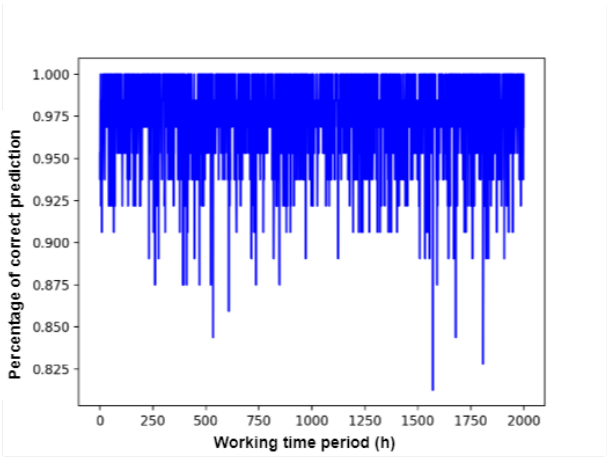

Figure 10: Case Three

It can be concluded that most of time periods the user retains a relatively high accuracy prediction rate. Although there are several time periods in which the prediction accuracy is relatively low, the system still thinks the user following a good computing usage behaviour and no insider threat happened. In other words, the Insider Catcher is not able to be detected in this case. From the case description, this case is about a current employee trying to search for other job positions. The reason for omitting this case is that Insider Catcher lacks of the ability to handle the text content. This can be our future research direction and more functions will be added to our system.

6 RELATED WORK

Since the invention of the computing science and Internet, cybersecurity issues are always accompanied with every technique boost development. Researchers have put much effort in this area to compete with Internet threats.

Various big data mining techniques are developed for different systems and environment. Rule-based approaches are one of the most widely used and developed methods[5, 25, 28, 40], which have systematic and scientific developing history. With the accumulation of tremendously historical defense records and the engagement of numerous experts, these methods can be very useful and efficient. However, this also can be a disadvantage. The investigation is vast and domain expertise is needed. Besides, the reaction of the methods is relatively slow when compared with the emerging Internet threats . For example, PerfAugur [28] was designed by specialists using data mining methods to extract features of performance problems. Predicated combination methods are used with a decent level of the knowledge base. Insider Catcher is trying to reduce the requirement of specific domain knowledge. For Insider Catcher, we attempt to increase the role of machine learning instead of human operations.

Other anomaly detection systems can combine machine learning algorithms like PCA, SVM, and RNN etc. Most of these methods design a data processor performing the data filtering and washing [7, 13, 14, 36]. After that, they do data transformation and build up a specially designed data structure for training purpose. Most of data structures contain key values, timestamps and several parameters associated with the threat cases. Anomaly detection models are usually built based on unsupervised machine learning models such as PCA [38, 39], Invariant mining (IM)[20], or supervised machine learning models like SVM [14] or RNN [24]. These anomaly detection methods are usually implemented based on a large historical data record, which requires a long-time data accumulation process. The training process is usually off-line and trained by certain time periods as sessions. It does not have the ability to perform online monitoring or real-time reaction tasks. Kim [18] and Staudemeyer [33] prove the performance of LSTM systems on intrusion detection area. The datasets they used are not related to insider threat and they only performed an off-line testing. However, their works still give us a strong support that LSTM is outstanding on the cybersecurity protection. Insider Catcher combines of the historical data analysis and online monitoring system based on LSTM. It can take a sequence input to predict ongoing cyber threats and the model can be self-adjusted during the online monitoring process.

Researchers have also started to involve in insider threat detection area [2, 14, 27, 34, 35]. Aaron et al. [35] presented an LSTM approach for insider threats detection in structured cybersecurity data stream. The system can generate an anomaly score for each user's online session. However, during the data processing state, the system demands specialists to mark all insider threats related suspicious user actions. It performs insider threat detection at per session level instead of per log record level. It can reduce the reaction speed for anomalies when performing the online analysis. Santosh et al. [2] also developed an LSTM-based anomaly detection system. Instead of performing the log analysis, he analyzed the memory behaviour of data nodes within a large Internet network. He used LSTM to detect anomaly event during the data transformation between each data node. However, he only performed local tests and online performance needs further research. Trishita et al. [34] also combines LSTM with SVM to perform intrusion detection. Unfortunately, they only performed tests based on a generated dataset which means the performance needs to be verified by utilizing real data in the future. Yuan et al. [41] also detected insider threats with LSTM and CNN machine learning methods. To increase the accuracy of detection, they manually labeled the data based on the key features of different insider threats scenarios. It is in fact not a fully automatic detection. Instead of using LSTM approach, Tabish et al. [27] uses Hidden Markov Models (HMM) for insider threats detection. HMM is applied to learn the normal users' behaviour and by identifying the deviations from it, the system is able to identify the threat events. Compared with LSTM, his system requires less data for training process and works well in some particular cases. Despite this, the limitation is that it requires experienced staffs to decide which features should be used for analysis and different cases require divergent features.

7 CONCLUSIONS AND FUTURE WORKS

In conclusion, we have demonstrated how Insider Catcher builds user patterns to indicate online computer usage behaviour and performs anomaly detection based on historical attack datasets. It can also continuously monitor users' behaviour by analyzing the log sequence with LSTM. This anomaly detection system can become a new method to secure the Internet security environment. The insider threat does exist widely in the enterprises but the way to deal with it is still in doubt in recent years. Its special characteristics and behaviour lead to insiders are not easily detected by traditional methods. The demand for the protection of this will draw much attention in the future. To leverage Insider Catcher provides some inspiration for researchers in this science area.

Insider Catcher contains two parts to handle historical records with the analysis system and to accomplish real-time online cybersecurity monitoring with the LSTM analysis system. By investigating the detail of detected threats, we noticed that most of threat events contain basic computing operations. Although it is hard to make a training model with the insider threat records, Insider Catcher can be used to analyze the employees' computing usage behaviour. By distinguishing the anomaly actions from the scheduled ones, we are likely to figure out some hints to detect such threats. We think Insider Catcher will help security staffs on cybersecurity protection by improving the speed and accuracy of detection.

This paper uses the CERT Insider threaten dataset to evaluate the effectiveness of Insider Catcher. The system was tested to verify whether it is able to find out the real insider threats. Meanwhile, the influences of some key factors were also tested. The results can be used to help enterprises or other researchers to develop systems which are suitable to their own Internet environment. On top of this, we studied several cases with Insider Catcher. All the case studies are real-world insider threat scenarios.Upon completion of analysis, it gives us an illustration of the reaction of the system and what experience we can gain from the testing.

In respect of the areas of improvement for the system, we make the following recommendations. Firstly, other RNN machine learning algorithms can be tested to compare the performance of different scenarios instead of utilizing the LSRM model. Text and content analytical ability can be enhanced. Even more features can be extracted from the raw dataset for analysis. It will increase the accuracy of the prediction. For example, considering the position and task for each employee, the relation with other staffs, etc. Some social statuses of employees can also be considered to classify the user pattern for a particular user.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. TensorFlow: A System for Large-Scale Machine Learning.. In *OSDI*, Vol. 16. 265–283.

[2] Santosh Aditham, Nagarajan Ranganathan, and Srinivas Katkoori. 2017. LSTM-Based Memory Profiling for Predicting Data Attacks in Distributed Big Data Systems. In *Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2017 IEEE International*. IEEE, 1259–1267.

[3] Matt Bishop and Carrie Gates. 2008. Defining the insider threat. In *Proceedings of the 4th annual workshop on Cyber security and information intelligence research: developing strategies to meet the cyber security and information intelligence challenges ahead*. ACM, 15.

[4] Dawn M Cappelli, Andrew P Moore, and Randall F Trzeciak. 2012. *The CERT guide to insider threats: how to prevent, detect, and respond to information technology crimes (Theft, Sabotage, Fraud)*. Addison-Wesley.

[5] Marcello Cinque, Domenico Cotroneo, and Antonio Pecchia. 2013. Event logs for the analysis of software failures: A rule-based approach. *IEEE Transactions on Software Engineering* 39, 6 (2013), 806–821.

[6] TK Das and P Mohan Kumar. 2013. Big data analytics: A framework for unstructured data analysis. *International Journal of Engineering Science & Technology* 5, 1 (2013), 153.

[7] Min Du and Feifei Li. 2016. Spell: Streaming parsing of system event logs. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 859–864.

[8] Mathias Ekstedt and Teodor Sommestad. 2009. Enterprise architecture models for cyber security analysis. In *Power Systems Conference and Exposition, 2009. PSCE'09. IEEE/PES*. IEEE, 1–6.

[9] Gemalto. 2017. Breach Level Index Report 2017. http://www.breachlevelindex.com/

[10] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with LSTM. (1999).

[11] Pankaj Goel, Aniruddha Datta, and M Sam Mannan. 2017. Application of big data analytics in process safety and risk management. In *Big Data (Big Data), 2017 IEEE International Conference on*. IEEE, 1143–1152.

[12] Ruben Gonzalez-Rubio, Jean Rohmer, and D Terral. 1984. The schuss filter: A processor for non-numerical data processing. *ACM SIGARCH Computer Architecture News* 12, 3 (1984), 64–73.

[13] Hossein Hamooni, Biplob Debnath, Jianwu Xu, Hui Zhang, Guofei Jiang, and Abdullah Mueen. 2016. LogMine: Fast Pattern Recognition for Log Analytics. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1573–1582.

[14] Shilin He, Jieming Zhu, Pinjia He, and Michael R Lyu. 2016. Experience report: system log analysis for anomaly detection. In *Software Reliability Engineering (ISSRE), 2016 IEEE 27th International Symposium on*. IEEE, 207–218.

[15] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications* 13, 4 (1998), 18–28.

[16] Shi-Jinn Horng, Ming-Yang Su, Yuan-Hsin Chen, Tzong-Wann Kao, Rong-Jian Chen, Jui-Lin Lai, and Citra Dwi Perkasa. 2011. A novel intrusion detection system based on hierarchical clustering and support vector machines. *Expert systems with Applications* 38, 1 (2011), 306–313.

[17] Michelle Keeney, Eileen Kowalski, Dawn Cappelli, Andrew Moore, Timothy Shimeall, and Stephanie Rogers. 2005. *Insider threat study: Computer system sabotage in critical infrastructure sectors.* Technical Report. National Threat Assessment Ctr Washington Dc.

[18] Jihyun Kim, Jaehyun Kim, Huong Le Thi Thu, and Howon Kim. 2016. Long short term memory recurrent neural network classifier for intrusion detection. In *Platform Technology and Service (PlatCon), 2016 International Conference on.* IEEE, 1–5.

[19] Chaochun Liu, Huan Sun, Nan Du, Shulong Tan, Hongliang Fei, Wei Fan, Tao Yang, Hao Wu, Yaliang Li, and Chenwei Zhang. 2016. Augmented LSTM Framework to Construct Medical Self-diagnosis Android. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on.* IEEE, 251–260.

[20] Jian-Guang Lou, Qiang Fu, Shengqi Yang, Ye Xu, and Jiang Li. 2010. Mining Invariants from Console Logs for System Problem Detection.. In *USENIX Annual Technical Conference.*

[21] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long short term memory networks for anomaly detection in time series. In *Proceedings.* Presses universitaires de Louvain, 89.

[22] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association.*

[23] Andriy Miranskyy, Abdelwahab Hamou-Lhadj, Enzo Cialini, and Alf Larsson. 2016. Operational-log analysis for big data systems: Challenges and solutions. *IEEE Software* 33, 2 (2016), 52–59.

[24] Anvardh Nanduri and Lance Sherry. 2016. Anomaly detection in aircraft data using Recurrent Neural Networks (RNN). In *Integrated Communications Navigation and Surveillance (ICNS), 2016.* IEEE, 5C2–1.

[25] Alina Oprea, Zhou Li, Ting-Fang Yen, Sang H Chin, and Sumayah Alrwais. 2015. Detection of early-stage enterprise infection by mining large-scale log data. In *Dependable Systems and Networks (DSN), 2015 45th Annual IEEE/IFIP International Conference on.* IEEE, 45–56.

[26] Marisa Reddy Randazzo, Michelle Keeney, Eileen Kowalski, Dawn M Cappelli, and Andrew P Moore. 2005. Insider threat study: Illicit cyber activity in the banking and finance sector. (2005).

[27] Tabish Rashid, Ioannis Agrafiotis, and Jason RC Nurse. 2016. A new take on detecting insider threats: exploring the use of hidden markov models. In *Proceedings of the 2016 International Workshop on Managing Insider Security Threats.* ACM, 47–56.

[28] Sudip Roy, Arnd Christian König, Igor Dvorkin, and Manish Kumar. 2015. Perfaugur: Robust diagnostics for performance anomalies in cloud services. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on.* IEEE, 1167–1178.

[29] S Rasoul Safavian and David Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics* 21, 3 (1991), 660–674.

[30] Haşim Sak, Andrew Senior, and Françoise Beaufays. 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association.*

[31] Eric D Shaw, Lynn F Fischer, and Andrée E Rose. 2009. *Insider risk evaluation and audit.* Technical Report. Defense Personnel Security Research Center Monterey CA.

[32] Robin Sommer and Vern Paxson. 2010. Outside the closed world: On using machine learning for network intrusion detection. In *Security and Privacy (SP), 2010 IEEE Symposium on.* IEEE, 305–316.

[33] Ralf C Staudemeyer. 2015. Applying long short-term memory recurrent neural networks to intrusion detection. *South African Computer Journal* 56, 1 (2015), 136–154.

[34] Trishita Tiwari, Ata Turk, Alina Oprea, Katzalin Olcoz, and Ayse K Coskun. 2017. User-profile-based analytics for detecting cloud security breaches. In *Big Data (Big Data), 2017 IEEE International Conference on.* IEEE, 4529–4535.

[35] Aaron Tuor, Samuel Kaplan, Brian Hutchinson, Nicole Nichols, and Sean Robinson. 2017. Deep learning for unsupervised insider threat detection in structured cybersecurity data streams. *arXiv preprint arXiv:1710.00811* (2017).

[36] Imtiaz Ullah and Qusay H Mahmoud. 2017. A filter-based feature selection model for anomaly-based intrusion detection systems. In *Big Data (Big Data), 2017 IEEE International Conference on.* IEEE, 2151–2159.

[37] Svante Wold, Kim Esbensen, and Paul Geladi. 1987. Principal component analysis. *Chemometrics and intelligent laboratory systems* 2, 1-3 (1987), 37–52.

[38] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael Jordan. 2009. Online system problem detection by mining patterns of console logs. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on.* IEEE, 588–597.

[39] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. 2009. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles.* ACM, 117–132.

[40] Ting-Fang Yen, Alina Oprea, Kaan Onarlioglu, Todd Leetham, William Robertson, Ari Juels, and Engin Kirda. 2013. Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks. In *Proceedings of the 29th Annual Computer Security Applications Conference.* ACM, 199–208.

[41] Fangfang Yuan, Yanan Cao, Yanmin Shang, Yanbing Liu, Jianlong Tan, and Binxing Fang. 2018. Insider Threat Detection with Deep Neural Network. In *International Conference on Computational Science.* Springer, 43–54.