User and Entity Behavior Analysis under Urban Big Data

ZHIHONG TIAN, Cyberspace Institute of Advanced Technology, GuangZhou University, GuangZhou, China
CHAOCHAO LUO, Venustech established Active Defense Laboratory, Beijing, China
HUI LU, SHEN SU, and YANBIN SUN, Cyberspace Institute of Advanced Technology, GuangZhou University, GuangZhou, China
MAN ZHANG, Peng Cheng Laboratory, Shenzhen, China

16

Recently, the urban network infrastructure has undergone a rapid expansion that is increasingly generating a large quantity of data and transforming our cities into smart cities. However, serious security problems arise with this development with more and more smart devices collecting private information under smart city scenario. In this article, we investigate the task of detecting insiders' anomalous behaviors to prevent urban big data leakage. Specifically, we characterize a user's daily activities from four perspectives and use several deep learning algorithms (long short-term memory (LSTM) and convolutional LSTM (convLSTM)) to calculate deviations between realistic actions and normalcy of daily behaviors and use multilayer perceptron (MLP) to identify abnormal behaviors according to those deviations. To evaluate the proposed multimodel-based system (MBS), we conducted experiments on the CERT (United States Computer Emergency Readiness Team) dataset. The experimental results show that our proposed MBS has a remarkable ability to learn the normal pattern of users' daily activities and detect anomalous behaviors.

CCS Concepts: • **Security and privacy;** • **Computing methodologies** → *Supervised learning by classification*; *Anomaly detection*;

Additional Key Words and Phrases: UEBA, anomaly detection, deep learning, security

ACM Reference format:
Zhihong Tian, Chaochao Luo, Hui Lu, Shen Su, Yanbin Sun, and Man Zhang. 2020. User and Entity Behavior Analysis under Urban Big Data. *ACM/IMS Trans. Data Sci.* 1, 3, Article 16 (September 2020), 19 pages.
https://doi.org/10.1145/3374749

This research is supported by the Guangdong Province Key Research and Development Plan (Grant No. 2019B010137004) the National Key Research and Development Plan (Grant No. 2018YFB0803504) and the National Natural Science Foundation of China (U1636215, 61871140, 61872100, 61572153), and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2019).
Author's addresses: Z. Tian, H. Lu (corresponding author), S. Su, and Y. Sun, Cyberspace Institute of Advanced Technology, GuangZhou University, No. 230, West Waihuan Road, GuangZhou 510006, China; emails: {tianzhihong, luhui, sushen, sunyanbin}@gzhu.edu.cn; C. Luo, ADLab of Venustech, ZhongGuanCun Software Park, No. 8 Dongbeiwang West Road of Haidian District, Beijing 100193, China; email: luochaochaoforaffair@gmail.com; M. Zhang, Peng Cheng Laboratory, No. 2 Xingke First Street of NanShan District, Shenzhen, 518040, China; email: zhangman@pcl.ac.cn.

2577-3224/2020/09-ART16 $15.00
https://doi.org/10.1145/3374749

1 INTRODUCTION

Cloud computing and big data techniques have been used to solve various problems and have transformed our cities into smart cities. Smart cities have effectively promoted the intensive development of regional or industrial information infrastructure, while the sharing of such intensive basic resources complicates the network environment and creates more risks of private data loss, stolen identity certifications, financial fraud, and other security problems. For instance, smart medical treatment is an important part of smart cities, and medical data collected from patients are widely used to help doctors to treat patients effectively, but these medical data are private and are easy to be leaked. Hence, there is an urgent need to prevent leakage of private data under urban big data scenarios. In urban big data security, threats from organization insiders are much more threatening; threats originating from the outside are prevented by intrusion detection systems (IDS), firewalls, and other security systems, while organization insiders can perform malicious acts much more easily and can cause worse damage because they are positioned internally and are much closer to private data or private servers. Most security experts agree that threats from insiders are the most difficult to prevent. It was reported that 53 percent of organizations encountered an attack from insiders, 27 percent thought that attacks from insiders were more frequent, and 90 percent admitted that they were unable to defend against insider threats. Furthermore, just last year, a report from the Ponemon Institute studied cyberattack cases for over 237 companies in six countries around the world and found that insider threat was the most expensive attack, costing companies an average of $167,890 annually, and this cost is likely to increase in the future [1].

There are three primary perpetrators from inside an organization: insiders who do not intend to cause damage, attackers who pose as insiders, and insiders who cause damage deliberately. For the first scenario, employees incurred some extraordinary efforts due to mistakes at work and caused losses. For the second, usually, when some employees' personal accounts are leaked, the attacker can log into the company's internal system after obtaining the account and then use the employee's authority to steal sensitive data or spread malicious files. For the last perpetrator, some employees who have been punished by the company or are dissatisfied with the company may retaliate by some malicious act. Before these users intentionally or unintentionally cause damage, they will exhibit an abnormal behavior pattern, such as sudden and frequent remote logins, sudden and frequent removable drive usage, or sudden and frequent sensitive file access, which deviates from the normal behavior trajectory of the user and is used to detect abnormal behaviors from insiders by researchers.

Similarly, we use the deviation between normal and abnormal behaviors. In this article, we propose a multimodel-based system (MBS) for anomaly detection of user behaviors through data analysis and can be used in big data, which includes more features. Specifically, in the MBS, we analyze data to characterize users' daily activities and habits and determine whether a user is performing a threatening operation from three perspectives:

Feature deviation is defined by the deviations between features to be detected and features predicted by the pretrained model by histories.

Sequence deviation is the deviation between the action sequence to be detected and the normal action sequence predicted by the pretrained model.

Role deviation describes the degree of deviation between role features to be detected and the role features calculated based on all features of users that are in the same group.

In this work, we make the following contributions. First, we comprehensively outline users' daily activities from four perspectives and utilize a multimodel to detect anomalous behaviors. Next, to the best our knowledge, this is the first study to use convolutional long short-term memory (convLSTM) to detect abnormal behaviors. Finally, we propose an MBS for anomaly detection of user behaviors through data analysis and evaluate it on the CERT dataset.

This article is organized as follows. Section 2 presents a brief review of related work. In Section 3, we describe the MBS workflow and the methodology details. The experimental settings and evaluation results are provided in Section 4. In Section 5, we conclude the article and discuss our future work.

2 RELATED WORK

Studies in Refs [2–11] have focused on the development of smart cities and the Internet of Things, and the work described in Refs [12–18] concentrated on solving the Internet of Things security problems. However, attacks from insiders have not been well researched in recent years while becoming one of the most dangerous threats. Detecting these threats is quite challenging because it is difficult to detect anomalous behaviors from daily activities, and malicious users with the technical ability to leverage these services often have sufficient knowledge and expertise to manage (and conceal) unauthorized activities. General literature reviews of abnormal behavior detection and guidelines for preventing attacks are provided by Refs [19] and [20].

In an attempt to identify anomalous behaviors, several typical studies have been central to researching anomalous user behavior detection. In Ref. [21], the authors focused on detecting anomalous behaviors with a novel ontology for detecting anomalous user behaviors from inside a banking domain database system in the financial sector. In their novel ontology, taxonomy was defined first; then, it was used to identify relationships between those basic defined ontology classes. The resulting structure is a domain ontology mapped onto the suggested upper merged ontology (SUMO), friend of a friend (FOAF), and finance ontologies to make their work integrable to the systems that use these ontologies and to create a broad knowledge base [21]. Experimental results have proven that their proposed model can realistically and systematically evaluate anomalous behaviors from inside organizations. Another typical method for anomalous behavior detection was proposed by Ref. [22]. In this work, the authors proposed a particularly designed tripwire grammar for anomalous behavior detection. First, anomalous actions were defined as two classes: actions that violate a policy that is specifically crafted to describe behaviors that are highly likely to be of concern if they are exhibited and behaviors that follow a pattern of a known insider threat attack [22]. Then, those actions are defined as tripwires within a system and implemented to detect anomalous behaviors.

Machine learning and deep learning techniques have made considerable progress in many fields. Machine learning and deep learning are increasingly applied to detect anomalous behaviors from inside organizations. Reference [23] extended existing works and proposed a novel approach to detect insider anomalous behaviors with hidden Markov models (HMMs). In this work, they used all normal behavior data and took advantage of HMMs to model a user's normal behaviors and learn what constitutes normal behavior. Specifically, every user's model was trained by their own history records to predict the next state. Whether a user will be identified as anomalous or normal mainly depends on the deviation between the prediction and the real record. Similarly, Ref. [24] also used HMMs and extended the work in Ref. [23]. In particular, Ref. [24] applied supervised learning and unsupervised learning to detect anomalous behaviors. In their work, they extracted two kinds of features, sequential data and numerical data, from log data, organization structure data, and users' information provided by organizations in the first stage and then used machine learning algorithms to detect abnormal behaviors. Specifically, an HMM was utilized to learn a user's habit of behaviors from sequential data, a decision tree algorithm (DT) was applied to use numerical data, and a self-organizing map (SOM) was used to combine the HMM and DT to determine whether a user has done something suspicious.

A body of research has focused on detecting anomalous behaviors in some interesting ways. References [25] and [26] conducted multimodel neurophysiological assessments to learn how users' brains act when the user is performing abnormal and normal activities. In particular, they

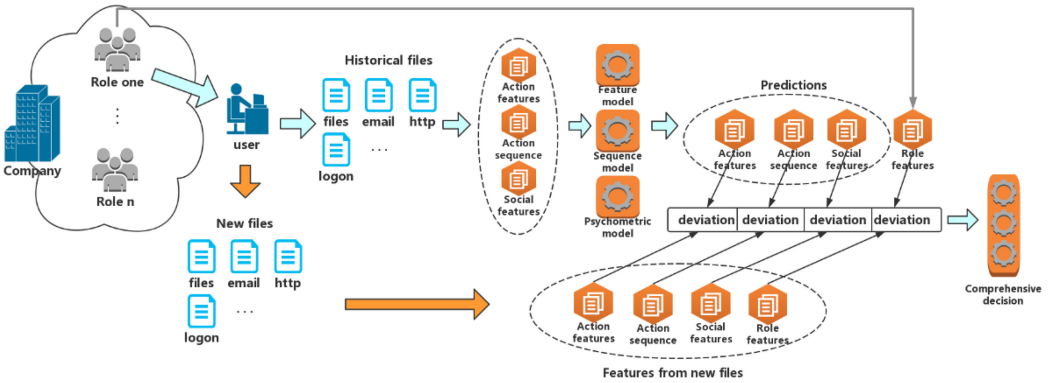

Fig. 1. Multimodel-based system (MBS) workflow.

focused on using electroencephalogram (EEG) signals that arise from the user's brain activities and eye tracking, which can capture spontaneous responses that are unfiltered by the conscious mind [25]. They extracted features from those signals and used them to train classifiers, such as support vector machine (SVM), k-nearest neighbor (KNN), and random forest (RF), to distinguish between anomalous and benign activities. Both experimental results showed that EEG signals can reveal valuable information about a user's malicious intent and can be used as an effective indicator in designing real-time insider threat monitoring and detection frameworks [25].

In our work, we aim to detect users' abnormal behaviors by analyzing data from four perspectives: action sequence, action features, social features, and role features. Similar to Refs [23] and [24], we use deep learning algorithms. Specifically, we use LSTM to learn the sequence of a user's behavior and ConvLSTM to learn the feature of a user's behavior. Finally, MBS performs anomaly detection based on the results from the two models above and role feature deviation.

3 METHODOLOGY

The principal of our work is to use deep learning models to analyze users' data through three perspectives to detect abnormal behaviors from insiders. We now describe the MBS workflow and explain every part of the MBS in detail.

3.1 MBS Design

Figure 1 illustrates the MBS workflow. To achieve anomaly detection, three primary models, a model for action features, a model for action sequence, and a model for role features, are designed in an MBS. On the left side of the workflow, workers in organizations or companies are grouped according to their roles, such as technicians, human resource representatives, salesmen, engineers, and managers. We designed this system by considering that users who are in the same group always have the same jobs, and role features can be extracted from these groups through their daily jobs, behaviors, and other data to justify detection to some extent. For instance, workers in the human resources sector are inclined to read resumes, send emails, and make phone calls throughout their working hours; if one employee seldom sends emails and makes phone calls but suddenly accesses some files that he or she never accessed before and begins frequently using removable drives when there are no major events in the company, this employee's behavior is suspicious, and security officers should be vigilant.

Historical data are needed to extract features and train models for every user. Typically, these data collected from inside an organization or company can be divided into three categories:

(i) users' actions and operational information on their own computers, (ii) all logs on servers and internal systems, and (iii) organization's structure and logs of major events and punishment. To derive useful information from these data, three types of features composed of action features, action sequence, and social features, are considered. Then, three models are devised to learn what kind of features constitute normal behaviors through the features extracted from historical data and make predictions for the next state. These features to be detected deviate from the predictions that match a normal distribution when a perpetrator's anomalous behavior occurs. Finally, to successfully detect abnormal behaviors through these deviations, four types of deviation are input into the pretrained comprehensive decision model.

3.2 Feature extraction

The goal of feature extraction is to identify relative information from log files (e.g., login and logout files, network traffic packets, file access logs) and convert them into a normalized representation from which deep learning algorithms can detect deviations when anomalous behavior occurs. Suitable features play a significant role in modeling a user's normal behavior and capturing deviations that are indicative of abnormal behaviors and can describe the threat profiles of suspicious users. Hence, it is useful to identify different actions a user can take (logging on, accessing a file, sending an email, etc.) that allow their behaviors to be modeled based on how insiders act and use them as our features [23]. Therefore, we extract action features, action sequences, social features, and role features through data that are aggregated from different sources. Action features, where every instance is converted into a fixed-length vector, and action sequences are the most widely used in user behavior anomaly detection, while role features and social features are missing dimensions of users' anomalous behavior detection.

- **Action features** are numerical features that are extracted to represent a user's features for his or her daily activities for each time period (day or week). There are two types of features that we should consider: which activities and what kind of features for each activity. The daily activities may differ for every user due to different sectors in which they participate. Hence, we analyze users' behaviors based on roles. The daily activities mostly depend on the files we can obtain, such as log activities from login and logout logs, thumb drive uses from connect and disconnect events, and network activities from network traffic packets. It should be noted that features are defined by experts based on the characteristics of every activity, and these features imply a user's daily work and working habits.
- **Action sequences** are sequential data that summarize the sequence of behaviors for each time period. We count every record of all activities of a user over a period of time first, and then we sort all records according to their time recordings and obtain a sequence of actions in a time series. For instance, a user logs onto a computer first, and then he or she browses three web pages, uses a thumb drive, sends an email, and finally logs out. The action sequence is {log on, web, web, web, drive connect, drive disconnect, email, log out}. Obviously, users perform different activities in different orders, and the action sequence is a feature of the user's working habits as well.
- **Social features** are a missing dimension in user behavior anomaly detection. This type of feature includes users' social media, topics they are interested in, and major events in the organization that are more comprehensive than other features. Social features are capable of identifying anomalous behaviors to some extent. They can also provide a reference for analyzing the causes of abnormal behavior because the user's abnormal behavior is often related to his or her social activities. For instance, some users may perform some malicious acts to retaliate after a dispute between colleagues or after a penalty has been assessed.

—**Role features** are statistical features from all colleagues in the same group. Obviously, colleagues in the same group have similar works and their action features are largely similar to each other. For example, colleagues in human resources will send emails frequently, browse the internal websites for recruitment frequently, read documents for resumes, and do other actions related to their roles during one day. Role features are mainly used to describe the common working characteristics of all users under this role. Different from action features, which are for individual users, role features are for all users. In this article, we define the average of action features selected from all colleagues under the same role as role features.

3.3 Deep learning strategy

The MBS is devised to employ several kinds of deep learning techniques to control scale and detect whether these entries, such login and logout files, network traffic packets, or file access logs, show suspicious behaviors. It is a general assumption that observable changes in these files (e.g., changes in frequencies of sensitive file access and thumb drive uses) are indicative of anomalous behaviors that may cause data leakage.

To learn what constitutes normal behaviors and detect abnormal behaviors, we take advantage of historical records to predict the next state and detect abnormal behaviors through the deviation between predictions and files to be detected. Most techniques chosen in MBS are prediction algorithms that have been widely used in natural language processing (NLP) or computer vision for a significant amount of time. It should be noted that the social model is not discussed in this work because of experimental limitations.

The MBS is a novel approach to anomaly detection with deep learning algorithms and provides a new combination of deep learning algorithms that are discussed in the following section. To the best of our knowledge, we are the first to apply convLSTM to detect anomalous behaviors of insiders.

3.3.1 LSTM. LSTM is a supervised method proposed by Ref. [27] that has been widely applied in NLP and speech recognition. Its capability of predicting the next state based on the current state sequence makes it the most widely used technique in various regression domain problems and has achieved remarkable success. Similarly, we take advantage of its capacity. In particular, we train LSTMs to learn the normal action sequences and predict the action sequences of the next state based on histories. The deviation between the true action sequence and the prediction is an indicator of anomaly detection. In our experiments, we use N days of action sequences to predict the action sequence of the next state, and the length of every user's action sequence is different. For example, a user's action can be represented as {log on, web, web, web, drive connect, drive disconnect, ..., email, log out} and input into LSTMs. As illustrated in Figure 2, LSTMs with two layers are unrolled; every LSTM unit has an input x_i in the first layer and $h_{1,i}$ in the second layer, and two outputs, hidden state h and cell output c, which are input into the LSTM unit in the next state. $h_{1,i}$ and $c_{j,i}$ are the hidden state output and cell output of the LSTM unit in the j_{th} layer at time i. x_i is an action sequence of one day, and N is the number of days used to predict the action sequence of the next state. Particularly, the model is composed of two LSTM layers with 100, and 160 units separately, a "tanh" activation layer after every LSTM layer, a Dense layer with 37 units, and a "relu" activation layer as illustrated in Table 2.

3.3.2 ConvLSTM. ConvLSTM is an evolved LSTM that was proposed by Ref. [28] in 2015 and has achieved considerable success in the tasks of video frame prediction. In contrast to LSTM, convLSTM not only has the capability of predicting the power of LSTM but also has the ability to describe the local CNN features. There are two reasons why we choose convLSTM: (i) the daily behavioral characteristics of the user can reflect the user's daily work and behavior habits, and we

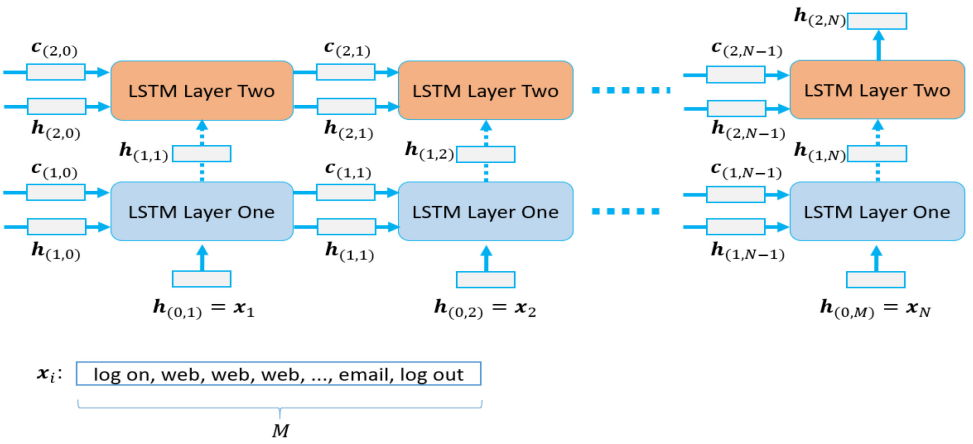

Fig. 2. Unrolled LSTMs with two layers.

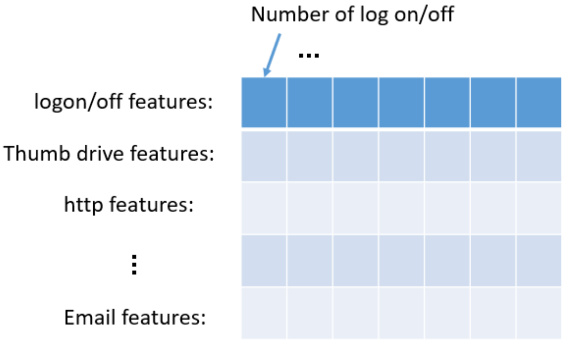

Fig. 3. Feature map of action features.

can use the predictive characteristics of LSTM for prediction, similar to action sequences; (ii) for different activities, we select different features to learn a user's daily behavior characteristics, but there are some potential connections between the features of different activities, and we can use CNN to extract these connections. In this work, action features are represented with a feature map that is delineated in Figure 3. As presented in Figure 4, the convLSTM extends traditional LSTM and uses convolutional structures in both the input-to-state and state-state conversions to replace simple calculating structures in traditional LSTM delineated in Figure 2. H_i and C_i are hidden state output and cell output, which are both 3D tensors whose last two dimensions are spatial dimensions (rows and columns) at time i. The input X_i is a feature map of action features at time i. Likewise, we use N days of feature maps to predict the feature map of the next state and take the deviation between the real feature map and prediction as a significant indicator in detecting abnormal behaviors. In this work, the model includes three ConvLSTM layers with filters 24, 128, and 48 separately; a "tanh" activation layer after each ConvLSTM layer; a maxpooling layer with kernel of 3×3; a dropout layer with rate of 0.5; a Dense layer with 48 units; and a "relu" activation layer as described in Table 3.

3.3.3 MLP. The comprehensive decision model is the final step in the MBS. In this section, we take advantage of MLP to combine the results of the multiple deep learning models above to perform anomaly detection. MLP is a kind of neural network that consists of an input layer, hidden

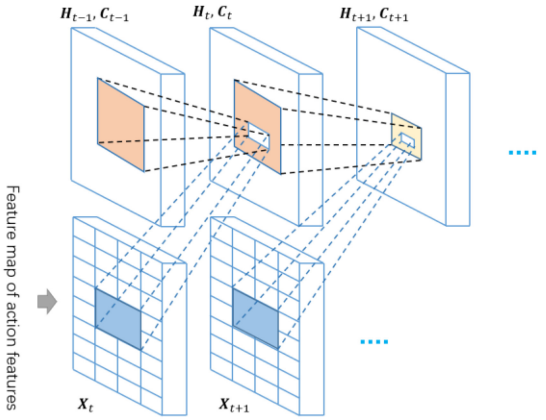

Fig. 4. Calculating structures in convLSTM.

layers, and an output layer and is often used to solve nonlinear problems. In the MBS, we need to use historical data to train the MLP to learn the previous relationship of these features. Then, the MLP determines whether there are abnormal behaviors according to deviations that include action sequence, action features, and role features, similar to performing a classification task.

4 EXPERIMENTS AND RESULTS

To evaluate the MBS, we conducted experiments on publicly available Computer Emergency Response Team (CERT) [29] datasets. We run all experiments in an environment with an Intel Core i7-6900k, 4 Kingston DDR4 16G, keras 2.2.2, 2 CUDA-enabled MSI GTX 1080Ti and Ubuntu 16.04. In our experiments, the social model is not discussed because of experimental limitations, and we considered action features, action sequences, and role features. Moreover, we defined the weighted deviation degree (WDD) to measure the deviations between the real features and prediction. For this case, changes in some features may not be indicative of abnormal behaviors. Therefore, we defined the WDD, which weighs the squared error linearly according to a weighted value. The WDD can be formulated as:

$$\mathrm{WDD} = \frac{1}{|V|} \sum_{\forall\, y \in V} w(y - \hat{y})^2, \tag{1}$$

where V is the set of all features in the real feature map, y is a single feature belonging to V, $\hat{y}$ is the same feature as y but belongs to the predicted feature map, and w is a specially designed value according the feature y.

4.1 Cert Dataset

The CERT insider threat dataset is a publicly available dataset for research, development, and testing of insider threat mitigation approaches [30]. The CERT dataset was generated with various models including topic models, behavior models, and psychometric models by the CERT at Carnegie Mellon University (CMU) and has been the most popular dataset for insider threat detection. On the one hand, we used CERT dataset because of the lack of dataset for insider threat detection. On the other hand, this system is applicable to big data, which includes more various features. The dataset simulates a company that has more than 1,000 employees and generates several kinds of logs (log on.csv, email.csv, device.csv, http.csv, file.csv, psychometric.csv) to characterize users' daily activities consisting of normal behaviors and anomalous activities in one day. In particular, logon.csv records all users' log-on or log-off activities with details, email.csv consists

Table 1. Comprehensive Features

Action features
Weekday log on/log off (users logged on or logged off during working time)
After-working log on (users logged on or logged off beyond working time),
Weekend log on (users logged on or logged off during the weekend)
Online time (the online time)
Num. device (number of thumb drives used)
Files exe copy (users copied exe files to thumb drives)
Files jpg copy (users copied jpg files to thumb drives)
Files txt/doc/pdf copy (users copied txt/doc/pdf files to thumb drives)
Files zip copy (users copied zip files to thumb drives);
Num. emails sending (number of emails sent)
Internal email sends (users sent emails by using company emails)
Num. Internal email receive (number of receivers' emails that are company emails)
Num external email receive (number of receivers' emails that are other emails)
Size of emails (the size of emails), Num. attachments (the number of attachments)
Num. websites (times of visiting websites)
Num. career sites (number of visits to job websites)
Num. news sites (number of visits to news websites)
Num. tech sites (number of visits to techniques websites)
Action sequence
Types of actions include log on, log off, http, device connect, device disconnect, email.
Role features
The average of features selected from action features of all employees in this group. These features include Weekday log on/log off, After-working log on, Weekend log on, Online time, Num. emails sending, Internal email send, Num. Internal email receive, Size of emails, Num. websites, and Num. tech sites.

of records of sending emails, http.csv records web activities, file.csv is composed of records of file accesses, and psychometric.csv provides a "Big Five personality traits score" for every user. Besides this, several files are provided for every user's anomalous records composed of these logs mentioned above, and a file records each user's employee name, user id, email, role, and team.

In our experiments, we use release 4.2 of the dataset to evaluate our system. According to the dataset description, release 4.2 contains a significantly greater number of insider threat incidents than other releases, which makes it possible to test the proposed detection systems against a more diverse set of scenarios [24].

In our experiments, user files under the role of "ProductionLineWorker" are employed for testing. The comprehensive features are presented in Table 1.

4.2 Evaluation results

In this section, we describe the results of LSTM, convLSTM, and the MBS on the CERT dataset. It is noteworthy that we use four-day features to predict features of the fifth day in both LSTM and convLSTM and calculate the deviations between real features and predictions. In addition, the files employed to train LSTMs and convLSTM are all benign so that the models can learn the normalcy of daily activities. To evaluate MBS, we used the approach proposed in Ref. [24] as a baseline and compared its results with the MBS results.

Table 2. Details of LSTM Model

Layers	Parameters
Input	Dim = 128
Reshape	Dim = (4,32)
LSTM	Units = 100
Activation	Function = tanh
LSTM	Units = 160
Activation	Function = tanh
Dense	Dim = 32
Activation	Function = relu

Table 3. Details of ConvLSTM Model

Layers	Parameters
Input	Dim = 92
Reshape	Dim = (4,6,8,1)
ConvLSTM	Filters = 24, kernel size = (2,3)
Activation	Function = elu
ConvLSTM	Filters = 128, kernel size = (2,3)
Activation	Function = tanh
ConvLSTM	Filters = 48, kernel size = (2,3)
Activation	Function = tanh
maxpooling	Pool size = (3,3)
Dense	Dim = 48
Activation	Function = relu

4.2.1 Action Sequence Results. As described in Section 3, LSTM neural networks are trained to learn the normal pattern of the action sequence of the user. In our experiment, we trained each user separately because every user has his or her own characteristics. Due to the time and data limitation, we use action sequences to train the LSTMs every 5 days, where the first 4 days are used as known data to predict the data of the fifth day, and the real data are compared to the LSTM predictions for error calculation and optimization.

Figure 5 shows the LSTM training process, which is designed to learn action sequences for a user. It is obvious that before epochs = 40, binary_accuracy maintains a stable value of approximately 0.65, and then immediately remains stable at approximately 0.95, while the loss value also falls smoothly. It appears that the user's action sequence habits have been accurately learned, but this is not the case. We found that it is not normal for there to be a large error between the test data and the training data. This result indicated that the LSTMs did not meet our expectations, although their accuracy and loss indicate that they had. After analysis, we found that the LSTMs were actually overfitting. We calculated the training data and test data losses and presented them in Figure 6. There is an obvious boundary between the first 160 days of data for training and the last 80 days of data for testing, and the training data losses are between 0 and 2, while the test data losses are between 2 and 10. Therefore, we modified and retrained the LSTMs, and the training process is shown in Figure 7. We can clearly see that the binary_accuracy and WDD loss tend to be stable before epochs = 40 and have distinct fluctuations when the epochs approach 40. This finding indicates that the LSTMs learn the user's daily behavior sequence before the epochs exceed

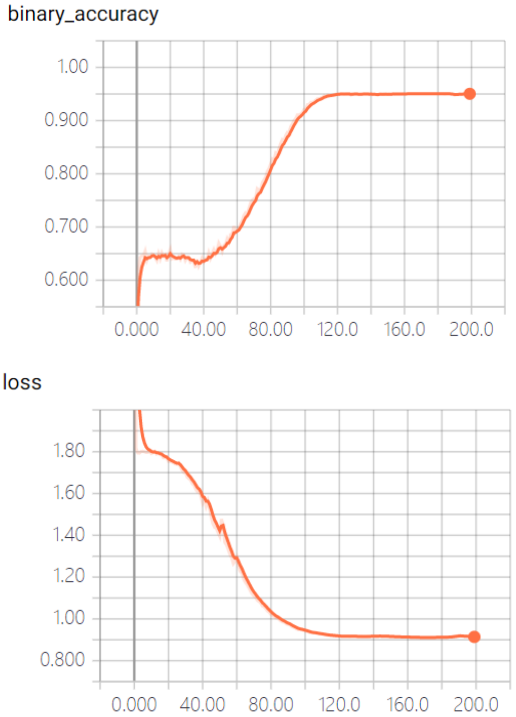

Fig. 5. Training process of overfitting.

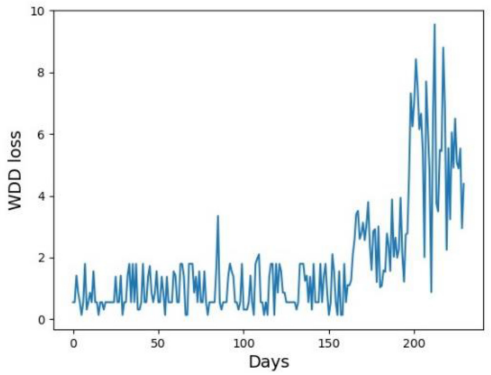

Fig. 6. WDD under overfitting.

40 and overfit after the epochs exceed 40. Similarly, we calculated the MDD loss of the training data and test data. As shown in Figure 8, we find that the first 40 days of test data have similar distributions to the first 160 training data, with losses ranging from 0 to 4. In addition, we find that there is a sharp change in the losses of the 200th day because the user had some abnormal behaviors near the 200th day, which causes the abnormal deviation between the predictions and real sequences.

According to the acquired results and analysis above, it is obvious that LSTMs utilized to learn a normal pattern of action sequences can be trained effectively with few data, and it is useful in the real world. As discussed above, the model in Figure 7 successfully learned the normal pattern

Fig. 7. Training process of pretrained LSTM.

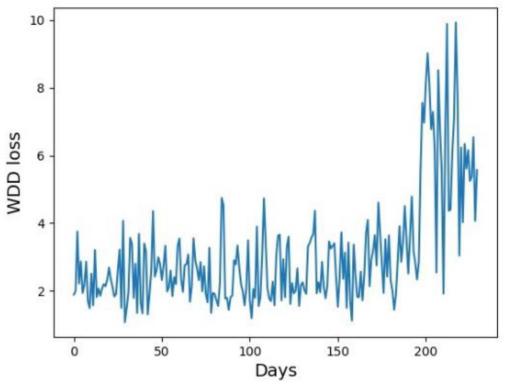

Fig. 8. WDD under pretrained LSTM.

of action sequences with flawed metrics of 0.555 in binary_accuracy and 3.2 in loss. Because the length and types of user's daily behavior are changing, they are within a certain range. Consequently, the action sequence is indicative of detecting anomalous behaviors, and the LSTMs show the capability of learning a normal pattern of action sequences with action sequences.

4.2.2 Action Feature Results. As discussed in Section 3.2, the user activity features are diverse, and there are potential connections between these various features. We are supposed to learn

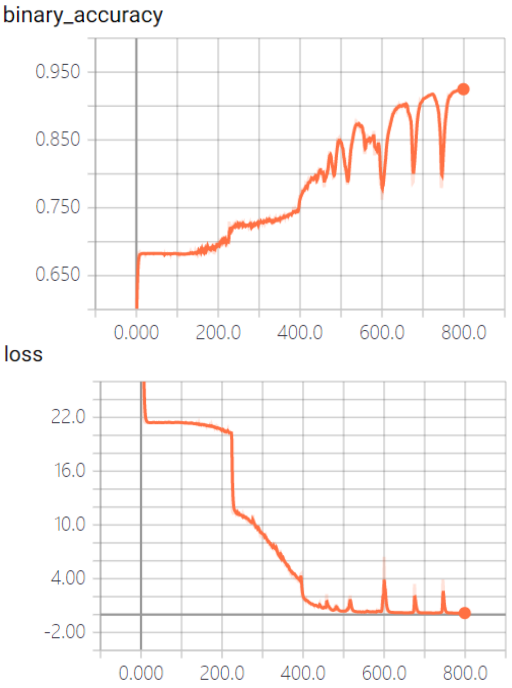

Fig. 9. Training process of overfitting.

not only the actions features of the user's daily behaviors but also the potential links between various action features, while the time and space characteristics of convLSTM are necessary for handling the user's action features. Similarly, we also utilize normal user feature maps to train the convLSTM every five days. The first four days of feature maps are used to predict the fifth day of data, and the error between the real fifth day of data and the predictions from convLSTM are calculated and optimized in the training phase. It should be noted that the defined feature maps of all users under the same role are the same, the convLSTM model is also the same, and all users share the same convLSTM model while different users save their own parameters.

Figure 9 records the process of training convLSTM for the same user above. In the loss diagram in Figure 9, it can be seen that the loss value tends to be stable at approximately 20 before epochs = 200. After the epochs exceed 200, the loss falls sharply, and then the loss stabilizes at approximately 0. Then, the accuracy tends to stabilize before epochs = 200, almost immediately after exceeding 200, and then fluctuates around 0.93. Generally, such training results are excellent results in other tasks that use neural networks. However, in our task, the results are terrible when we calculate the deviation of the training data and test data. We encountered the same problems in LSTMs—overfitting.

The overfitting in convLSTM is more obvious, and we can clearly see the difference in deviations (MSE and WDD) between the first 160 days of data for training and the last 80 days of data for testing, as shown in Figures 10 and 14. Clearly, the convLSTM did not learn the features of different behaviors. Hence, we conducted new experiments and presented the training results in Figure 11. We can speculate from the figure that the model in the state with loss = 72 and acc = 0.718 will perform as we expected. Afterward, we calculated the deviations (MSE and WDD) of the training data and test data and delineated them in Figures 12 and 13. As illustrated in Figures 12 and 13, the deviations of the first 200 days are almost in the same interval (0–1.5), and the remaining deviations

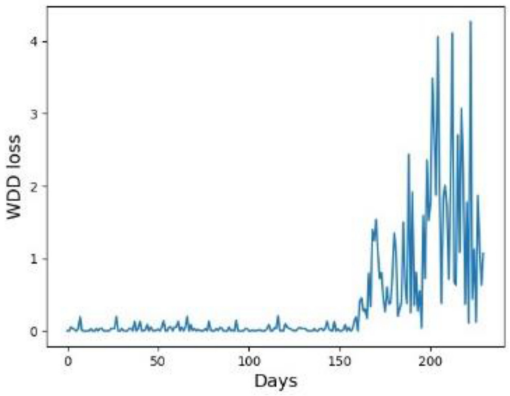

Fig. 10. WDD under overfitting.

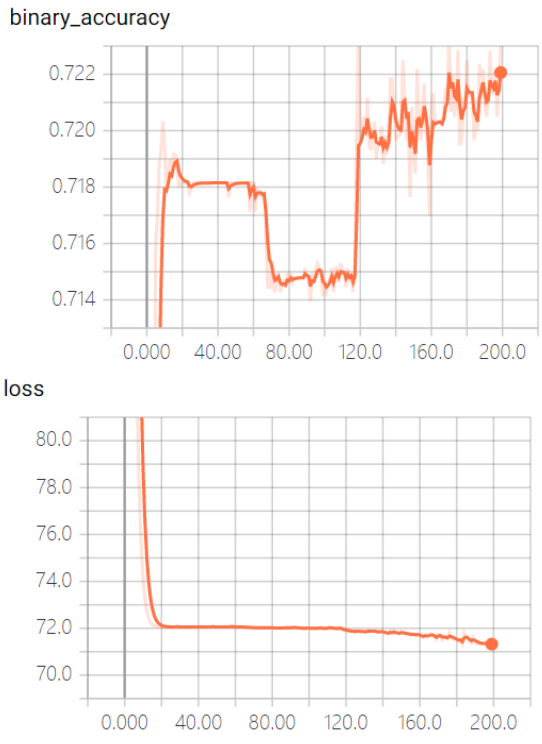

Fig. 11. Training process of pretrained convLSTM.

have sharp increases that are significantly different from the previous deviation. In addition, in Figure 13, the number of abnormal deviations that indicate abnormal behaviors after 200 days is only 2, but there are more than two malicious behaviors, and the MSE loss failed to detect them. In Figure 12, we can see more abnormal deviations, which corresponds with the actual situation.

4.2.3 Role Feature Results. In this work, we used the partial action features of all users under the same role to perform the mean value and then calculated the WDD value between the user's daily role features and the standard role features that were calculated; then, we presented the results in

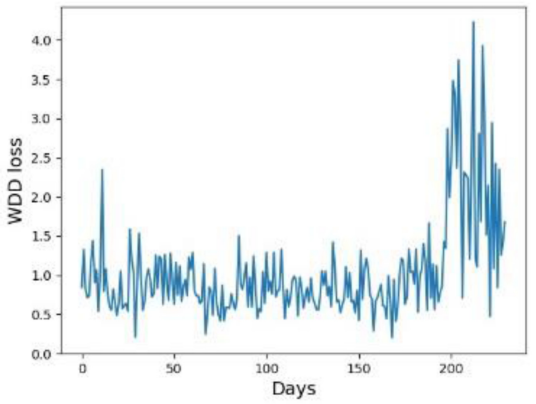

Fig. 12. WDD under pretrained convLSTM.

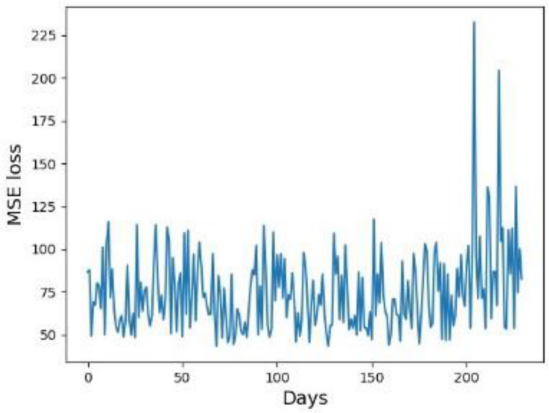

Fig. 13. MSE under pretrained convLSTM.

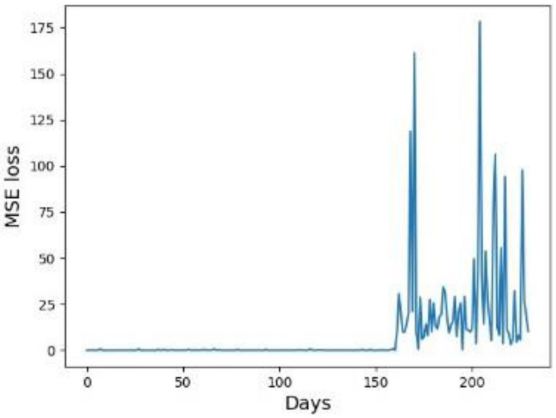

Fig. 14. MSE under overfitting.

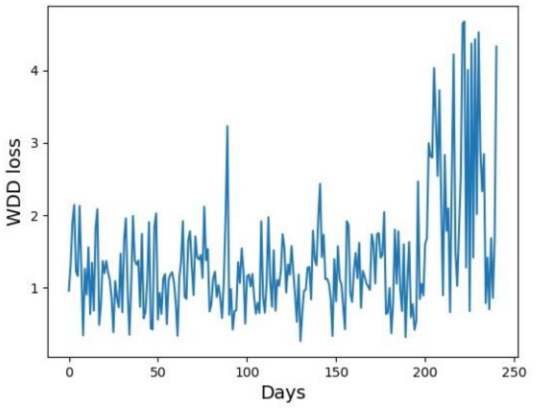

Fig. 15. WDD of Role features.

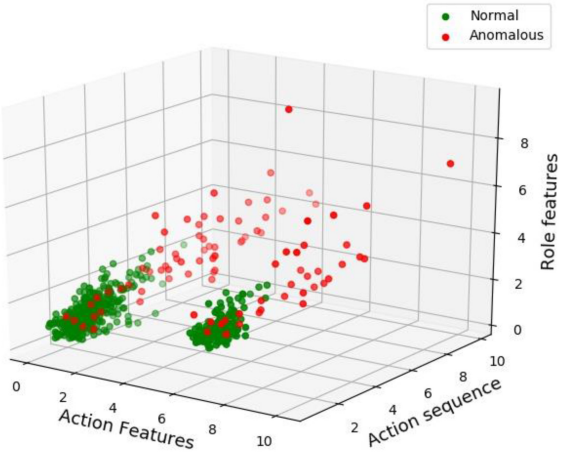

Fig. 16. Distributions of deviations of normal and anomalous data.

Figure 15. As shown in Figure 15, it can be seen that in the first 200 days, the deviation between the user's daily features and the standard role features fluctuated within the range of 0–2, but 200 days later, the user started to exhibit some suspicious behavior, and there was a significant difference. This indicates that the role features can reflect whether the user's daily behavior conforms to normalcy to some extent and can be indicative of abnormal behavior detection.

4.2.4 MBS Results. In our proposed MBS, we aim to determine whether or not users have malicious behaviors from four perspectives (social factors are not discussed in this work). We obtained three different degrees of deviation from the three perspectives above and delineated parts of them in Figure 16. In Figure 16, it is clear that normal points and anomalous points are separable. The three features reflect the abnormal behavior of users to some extent, although there are false positives and false negatives. For the task of raising alarms more accurately when abnormal behaviors occur, we use MLP to learn the relationship between the three deviations. Specifically, we concatenated the three types of deviations and used them to train the MLP to learn these potential links. Finally, the MLP determines whether the user has abnormal behaviors on a certain day.

Table 4. Experimental Metrics of the MBS and Baseline

Model	Accuracy	TPR	FPR	Precision
MBS	0.97	**0.98844**	0.14815	0.97714
SUS	0.94	0.96532	0.2222	0.96532

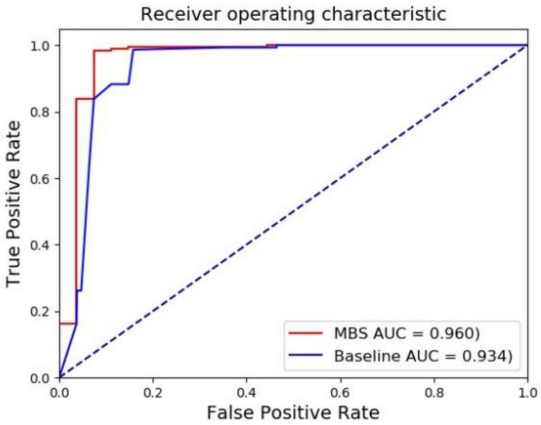

Fig. 17. ROC curves of the MBS and baseline.

Table 4 and Figure 17 show the experimental metrics of MBS and SUS (Supervised and Unsupervised learning detection System) in Ref. [24], and it is obvious that the MBS achieves promising results and is superior to the baseline model in all metrics. In the ROC curve, the SUS and MBS have similar trajectories, but MBS is slightly better than SUS. Specifically, the area under the curve (AUC) value of the MBS reaches 0.96, which is a very powerful illustration of the effectiveness of MBS.

5 CONCLUSION

In this article, we examined the problem of detecting abnormal behaviors from insiders by considering four perspectives: action features, action sequence, social features, and role features. Our main contribution is the development and evaluation of a novel system that takes advantage of multiple models to learn a user's normal pattern of behaviors to identify anomalous behaviors.

From the results presented above, we can clearly see that every type of feature has the capability of identifying abnormal behaviors. However, there are false positives and false negatives when using every single feature to identify deviations. To detect abnormal behaviors more accurately, we use the MLP to perform a comprehensive decision by taking advantage of deviations from every type of feature. Consequently, experimental results of the MBS show its promising ability to detect abnormal behaviors from insiders.

5.1 Limitations

In the exploration of our work on detecting abnormal behaviors from insiders, there are some limitations that may limit the effectiveness of our proposed approach in some scenarios. In the following, we will enumerate two of the most important limitations and give some advice to address them.

Previous data are anomalous: The most significant limitation is that we made an assumption that the previous data used to train the models to learn normalcy of a user's daily activities are represented in the initial data. This assumption will lead to the failure to detect abnormal behaviors when a user launches attacks from the very beginning. This approach is based on users' normalcy of activities, and another method is needed to address this problem in that scenario.

No anomalous data for the MLP: As discussed in Section 3, the MLP is utilized to learn the potential connections between these features and detect anomalous behaviors. The key point is that anomalous data are needed for training the MLP, while anomalous data are difficult to obtain in realistic scenarios. A possible solution is to conduct several simulations and collect data in realistic scenarios.

5.2 Future research

Future work on this topic will focus on, but is not limited to, the following directions. Primarily, our proposed approach of detecting abnormal behaviors is based on the normalcy of users' daily activities, and finding a method to address the first limitation above is significant for this topic. Second, we considered four types of features to characterize a user's daily activities and identified anomalous behaviors with deviations from different features, but we did not conduct experiments for social features due to the experimental limitation. Exploring social features and experimenting on a more comprehensive dataset will be our main direction in the future to characterize users' behaviors more accurately. Finally, anomalous data, which are hard to collect, are needed to train the MLP, and we want to find another method without this limitation to replace the MLP.

REFERENCES

[1] Y. Hashem, H. Takabi, R. Dantu, and R. Nielsen. 2017, October. A multi-modal neuro-physiological study of malicious insider threats. In *Proceedings of the 2017 International Workshop on Managing Insider Security Threats* (pp. 33–44). ACM.

[2] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang. 2015, October. A hierarchical distributed fog computing architecture for big data analysis in smart cities. In *Proceedings of the ASE Big Data & Social Informatics* 2015 (p. 28). ACM.

[3] Z. Tian, W. Shi, Y. Wang, C. Zhu, X. Du, S. Su, Y. Sun, and N. Guizani. 2019. Real time lateral movement detection based on evidence reasoning network for edge computing environment. *IEEE Transactions on Industrial Informatics* 15, 7 (2019), 4285–4294.

[4] E. Al Nuaimi, H. Al Neyadi, N. Mohamed, and J. Al-Jaroodi. 2015. Applications of big data to smart cities. *Journal of Internet Services and Applications*, 6(1), 25.

[5] Z. Tian, X. Gao, S. Su, J. Qiu, X. Du, and M. Guizani. 2019. Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory. *IEEE Transactions on Vehicular Technology* 68, 6 (2019), 5971–5980.

[6] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. 2014. Internet of Things for smart cities. *IEEE Internet of Things Journal* 1, 1 (2014), 22–32.

[7] Z. Tian, S. Su, W. Shi, X. Du, M. Guizani, and X. Yu. 2019. A data-driven method for future internet route decision modeling. *Future Generation Computer Systems*. 95 (2019), 212–220.

[8] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira. 2011, May. Smart cities and the future internet: Towards cooperation frameworks for open innovation. In *The Future Internet Assembly* (pp. 431–446). Springer, Berlin.

[9] J. Qiu, L. Du, D. Zhang, S. Su, and Z. Tian. 2019. Nei-TTE: Intelligent traffic time estimation based on fine-grained time derivation of road segments for smart city. *IEEE Transactions on Industrial Informatics*. 2019. DOI: 10.1109/TII.2019.2943906

[10] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7), 1645–1660.

[11] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton. 2009. Smart objects as building blocks for the Internet of Things. *IEEE Internet Computing*, 14(1), 44–51.

[12] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian. 2019, April. Toward a comprehensive insight into the eclipse attacks of tor hidden services, *IEEE Internet of Things Journal* 6, 2 (April 2019), 1584–1593.

[13] J. Pan and Z. Yang. 2018, March. Cybersecurity challenges and opportunities in the new edge computing+ IoT world. In *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization* (pp. 29–32). ACM.

[14] Y. Xiao, V. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway. 2007, Sept. A survey of key management schemes in wireless sensor networks, *Journal of Computer Communications* 30, 11–12, (Sept. 2007), 2314–2341.

[15] Y. Han, S. Etigowni, H. Liu, S. Zonouz, and A. Petropulu. 2017, 2020, October. Watch me, but don't touch me! Contactless control flow monitoring via electromagnetic emanations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1095–1108). ACM.

[16] Z. Tian, M. Li, M. Qiu, Y. Sun, and S. Su. 2019. Block-DEF: A secure digital evidence framework using blockchain, *Information Sciences*. 491(2019) 151–165.

[17] R. Doshi, N. Apthorpe, and N. Feamster. 2018, May. Machine learning DDoS detection for consumer Internet of Things devices. In *2018 IEEE Security and Privacy Workshops (SPW)* (pp. 29–35). IEEE.

[18] X. Du and H. H. Chen. 2008, Aug. Security in wireless sensor networks, *IEEE Wireless Communications Magazine*, 15, 4 (Aug. 2008), pp. 60–66

[19] The CERT Insider Threat Center. 2016. Common sense guide to mitigating insider threats, fifth edition, *CERT, SRI, Carnegie Mellon University, Tech.* Rep. CMU/SEI-2015-TR-010, 2016.

[20] National Cybersecurity and Communications Integration Center. 2014. Combating the insider threat, The Us Department of homeland security, Tech. Rep., 2014.

[21] G. Kul and S. Upadhyaya. 2015, October. A preliminary cyber ontology for insider threats in the financial sector. In *Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats* (pp. 75–78). ACM.

[22] I. Agrafiotis, A. Erola, M. Goldsmith, and S. Creese. 2016, October. A tripwire grammar for insider threat detection. In *Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats* (pp. 105–108). ACM.

[23] T. Rashid, I. Agrafiotis, and J. R. Nurse. 2016, October. A new take on detecting insider threats: exploring the use of hidden markov models. In *Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats* (pp. 47–56). ACM.

[24] D. C. Le and A. N. Zincir-Heywood. 2018, May. Evaluating insider threat detection workflow using supervised and unsupervised learning. In *2018 IEEE Security and Privacy Workshops (SPW)* (pp. 270–275). IEEE.

[25] Y. Hashem, H. Takabi, R. Dantu, and R. Nielsen. 2017, October. A multi-modal neuro-physiological study of malicious insider threats. In *Proceedings of the 2017 International Workshop on Managing Insider Security Threats* (pp. 33–44). ACM.

[26] Y. Hashem, H. Takabi, M. GhasemiGol, and R. Dantu. 2015, October. Towards insider threat detection using psychophysiological signals. In *Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats* (pp. 71–74). ACM.

[27] S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory[J]. Neural Computation 9(8) (1997), 1735–1780.

[28] S. H. I. Xingjian, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems* (pp. 802–810).

[29] https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=508099.

[30] J. Glasser and B. Lindauer. 2013, May. Bridging the gap: A pragmatic approach to generating insider threat data. In *2013 IEEE Security and Privacy Workshops* (pp. 98–104). IEEE.

Received June 2019; revised September 2019; accepted November 2019