Journal of Computers Vol. 32 No. 4, 2021, pp. 201-210
doi:10.53106/199115992021083204016

Research on Insider Threat Detection Method Based on Variational Autoencoding

Zhenjiang Zhang[1*], Lulu Zhao[1], Yang Zhang[1], Hongde Zhou[1], Wei Li[2]

[1] Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education, School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing100044, China
zhangzhenjiang@bjtu.edu.cn, 19125076@bjtu.edu.cn, zhang.yang@bjtu.edu.cn, 17221301@bjtu.edu.cn

[2] Beijing Topsec Co., Ltd. Beijing 100193, China
rd_liwei@topsec.com.cn

Received 1 June 2021; Revised 1 July 2021; Accepted 2 August 2021

Abstract. In recent years, internal attacks have posed a serious threat to the security of individuals, companies and even the country. Machine learning is currently a common method of insider threat detection. However, this technology requires a series of complex feature engineering, which has certain limitations in practical applications. This paper comprehensively considers the user's business operation behavior data and internal psychological data, and establishes an internal threat detection model to analyze their potential associations. The main tasks are as follows: In order to improve the fine-grained features of heterogeneous behavior log data and accurately reflect user behavior attributes, a session-based full feature extraction method is proposed. In this method, combined with a variational autoencoder, a long and short-term memory variational autoencoder (LVE) model is proposed. Taking into account the time characteristics of user behavior, a long and short-term memory network is used in the codec part, that is, input data, generate hidden variables, and then restore output data through hidden variables. The results show that this method improves the recall rate compared with other algorithms. Finally, the main work and improvement prospects are summarized.

Keywords: detection efficiency, insider threat, user behavior analysis, variational autoencoder

1 Introduction

1.1 Research Background and Significance

In the digital age, the complexity of information systems and science and technology makes many organizations vulnerable to the dual pressures of external attacks and internal threats. According to the "2018 U.S. Internet Crime Survey", 25% of cyber attacks are carried out by insiders, and 30% of respondents said that internal attacks are more destructive than external attacks [1]. It can be seen that insider threats are one of the most challenging threats in cyberspace, and they usually cause significant losses to organizations [2].

Since Internet intrusions can break through network boundaries such as firewalls or launch internal attacks by pretending to be internal personnel of the organization, comprehensive defense of the security of information systems has become the focus of research in many universities and enterprises. The focus is on how to identify "may" or "have" anomalies from users authorized by the organization. In addition, in real scenarios, the number of insider threat behaviors is small, and they are scattered among the normal behaviors of users. Looking for abnormal data from massive amounts of data is tantamount to finding a needle in a haystack. The existing internal threat detection algorithms are mainly machine learning, but this method relies heavily on feature engineering, so it is difficult to accurately capture the behavior differences between malicious users and normal users.

* Corresponding Author

Due to the limitations of traditional machine learning algorithms, it is necessary to establish an efficient detection mechanism that can not only monitor user behavior on a large scale, but also quickly judge internal threats by analyzing user habits, so as to reduce the workload of operation and maintenance personnel. Deploying the internal threat detection mechanism in a real scenario can effectively monitor the activity status of internal employees in order to assess the security status of the network in real time.

1.2 Related Works

In this paper, by investigating analysis, internal threat detection methods are divided into the following categories: methods based on statistical models, based on graph clustering, and based on machine learning.

Methods based on statistical models mainly use Hidden Markov Model and Gaussion Mix Model. Rashid et al. [3] analyzed the CMU-CERT data set with the help of a hidden Markov model. First, each item was assigned a symbol, then the files were merged and partitioned according to users, then user behaviors were extracted, and normal behavior baselines were established, and finally used hidden The Markov model recognizes normal behavior, and then uses the model to check the deviation of the test data from the normal behavior.

Graph-based anomaly detection methods are mainly divided into two types: structure-based anomaly detection and community-based anomaly detection. Liu et al. [4] integrated the logs generated by various devices into a heterogeneous graph, counted the five attribute values of each behavior, and set different granular association rules based on these attributes. The essence of graph-based anomaly detection is to extract corresponding features from the original data, build a graph database based on the features, and find objects that do not conform to the distribution of most points/edges/substructures.

Qiaona Hu et al. [5] applied the principal component analysis algorithm to analyze the multi-source log of the enterprise, counted the daily behavior data of the user by counting the frequency of log messages, and then established a model for each user. When the current behavior of the user is completely different from the previous behavior Then mark the current user as a malicious user.

In summary, the main insider threat detection methods based on machine learning in recent years [6] can be summarized as follows:

(1) Data collection: Behavior logs generated by different activities of users are the main source of data, such as e-mail exchanges, network access records, etc.

(2) Data preprocessing: construct feature vectors from the data collected in the previous step at different time granularities.

(3) Based on the feature vector constructed in the previous step, different machine learning algorithms are used to analyze the data.

(4) Determine whether the user is an internal threat user based on the results of data analysis.

1.3 Research Content and Main Work

Compared with the traditional attack methods, the existing internal threats have the characteristics of high-risk and concealment [7]. It can be seen that the detection of internal threats faces new challenges of cross-domain and cross-time nodes. Therefore, the research content of this article is as follows:

The feature extraction method is very important for the detection of internal threats. The existing feature extraction method divides a fixed time window, and uses the frequency of user behavior in the fixed time window as the corresponding feature value. Internal attacks often last for a period of time, and it is easy to miss abnormal behaviors using fixed time windows to extract features. Based on this, the user login-logout behavior is regarded as a session. This article proposes a session-based full feature extraction method, which can more accurately represent all related events related to user activities such as devices and files.

The above feature extraction method is combined with the variational autoencoder to construct the LVE model. In order to extract the deep features of the time series data, the LSTM network is used to learn user behavior in the encoding and decoding part, and the potential space of the generated data is compared with the potential space of the input data through this method. Complete the internal threat detection.

2 Insider threat Related Theories

2.1 Overview of Insider Threat Detection

Among the different definitions of insider threats, what everyone generally agrees with is the definition of insider threats in the "Guiding Documents on Insider Threats" issued by CERT [8] in 2012: insider threats are defined as internal threats by individuals who have or have authorized access to organizational assets. A malicious or unintentional way, using its authority to make a behavior that has a negative impact on the organization. Insider threat models can be roughly divided into the following types:

(1) Subject-based model

In response to internal threats, researchers hope to obtain different behavior extraction patterns based on different models. The CMO model proposed by Wood B et al. [9] and the SKRAM model proposed by Parker DB et al. [10] are two internal threat models that appeared earlier. This model considers the factors of successful internal attacks from the perspective of internal attackers. Subjective factors include motivation, resources, and skill literacy, while objective factors include poor organizational defenses and lack of effective supervision. The CMO model is based on the subjective aspects of the attacker. The model defines three conditions: insider threat users, tools that can implement insider threats, and an environment that provides insider threats. An attacker can successfully implement an attack requires three subjective elements: Capability, Motivation, and Opportunity. Only when these three elements are met, internal users can successfully implement the attack.

(2) Object-based model

Role-Based Access Control is an extension proposed to deal with the limitations of RBAC in terms of internal threats [11]. CRBM not only inherits the advantages of RBAC's scalable management, least privilege and separation of duties, but also provides a scalable and reusable mechanism.

(3) Models based on artificial intelligence

Traditional insider threat detection is mainly based on signatures or rules, but with the rise of artificial intelligence, deep learning has gradually become the mainstream technology for insider threat detection.

In general, the general flow of the artificial intelligence model is shown in Fig. 1 below.

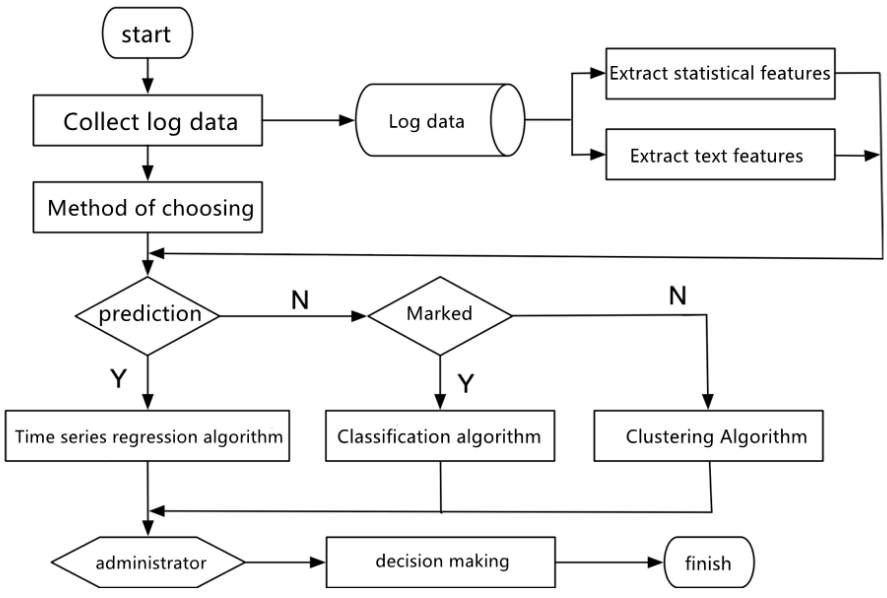

Fig. 1. General model and process based on artificial intelligence

As can be seen from the above figure, the internal threat detection method based on the artificial intelligence model is to establish a user's normal behavior pattern model, extract features and train the model according to different methods. The main steps are as follows:

(1) Data preparation: Collect log data on various devices.

(2) Feature extraction: The two commonly used features in log data are statistical features and text features. Statistical features refer to the number of occurrences of statistical events within a divided time

window. The text features are transformed into digital form through text processing technologies such as word2vec and TF-IDF.

(3) Selection method: choose a suitable deep learning method and build a model.

(4) Detection: Input the data to be detected and get the result.

(5) Decision-making: The administrator further analyzes based on the results.

The data set is essential for the research of insider threat detection. Currently, there is no open and comprehensive real data set that can be directly used for insider threat detection. The existing public data sets are usually artificially operated (simulating malicious behavior) to generate attack sessions to simulate internal threats. Common internal threat data sets include RUU data set [12], TWOS data set [13], CMU-CERT data set [14] and so on.

2.2 Insider Threat Detection Algorithm

Auto Encoder is a deep learning neural network model proposed by Rumlhart et al. [15] for semi-supervised and unsupervised learning. The model learns the hidden features of the original data through the encoder, and reconstructs the newly learned features into the same data as the original data through the decoder. The model of the autoencoder is shown in Fig. 2:

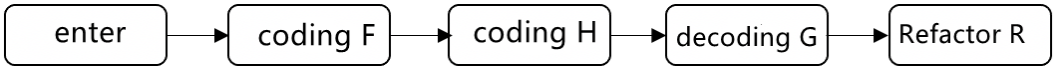

Fig. 2. Basic structure of autoencoder

To break through the limitations of autoencoders, Kingma et al. proposed Variational Auto Encoder in 2014. Different from the autoencoder, the variational autoencoder mainly consists of two parts: modeling probability distribution (encoder) and conditional generation model (decoder).

The variational autoencoder model can be divided into two parts:

The coding network of the variational autoencoder uses the variational method to estimate the approximate posterior distribution $q(z;\phi)$, The input of the coding network is the original data *x*, and the output is the variational distribution obtained by sampling the hidden variables $q_\phi(z \mid x)$.

The entire network structure of the variational autoencoder is shown in Fig. 3 below. The solid line represents the network calculation operation, and the dashed line represents the sampling operation.

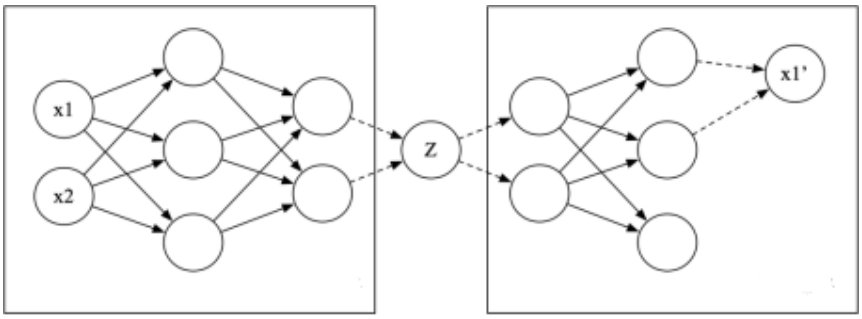

Fig. 3. Variational autoencoder network structure

In order to determine the difference between the coding network model and the true posterior distribution, the variational autoencoder uses KL divergence to measure the similarity between the two, so it is necessary to find a suitable KL divergence that can minimize the two distributions, which is:

$$\Phi,\theta = \arg\min_{\phi,\theta} D_{KL}(q_\phi(z \mid x) \,||\, P_\theta(z \mid x)) \tag{1}$$

In the generative network, according to the joint probability distribution $P_\theta(x \mid z) = P_\theta(z)P_\theta(x \mid z)$, The probability density of the generated sample can be calculated according to the neural network, because:

$$\begin{aligned} L(\theta,\phi;X) &= E_{q_\phi(z|x)}[-lbq_\phi(z\mid x)+lbP_\theta(x,z)] \\ &= -D_{KL}(q_\phi(z\mid x)\mid P_\theta(z))+E_{q_\phi(z|x)}lbP_\theta(x\mid z) \end{aligned} \tag{2}$$

So the optimization goal of the generating network is transformed into:

$$\Phi,\theta=\arg\max_{\phi,\theta} L(\theta,\phi;X) \tag{3}$$

Therefore, the optimization goal of the variational autoencoder is to find a set of network parameters that can maximize L. This paper uses reparameterization method to transform the random sampling relationship betweenzand into a deterministic functional relationship. By introducing a random variable ε with a distribution of $p(\varepsilon)$, the second term can be written as:

$$E_{z\sim q_\phi(z|x)}[\log p_\theta(x\mid z)]=E_{\varepsilon\sim p(\varepsilon)}[\log p_\theta(x\mid g(\phi,\varepsilon))] \tag{4}$$

Reparameterize according to the following formula:

$$z=\mu+\sigma\bullet\varepsilon \tag{5}$$

After re-parameterization, the network structure of the variational autoencoder is shown in Fig. 4:

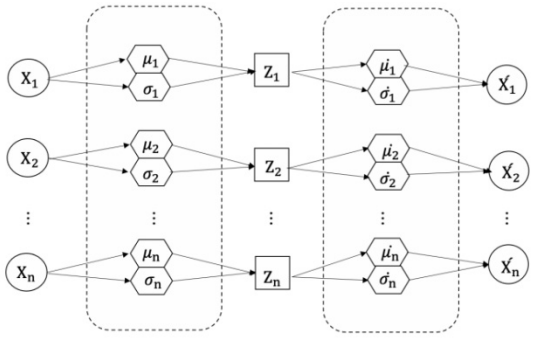

Fig. 4. Network structure of variational autoencoder

As shown in the figure above, by introducing auxiliary parameters, the relationship between hidden variables and parameters changes from a sampling relationship to a definite relationship, so that each item of the objective function can be calculated, and the model result is also stable.

Variational autoencoders are now widely used in tasks such as image classification, natural language processing, and face recognition. It is these applications that make the development prospects of variational autoencoders broader.

3 LVE-based Internal Threat Detection Algorithm

The long and short-term memory network has a memory function, so it can process long series of data. Variational autoencoders also have better advantages in non-stationary time series reconstruction. Input the abnormal data into the trained model. Since the abnormal data does not conform to the normal data distribution, the reconstruction loss value deviates from the normal data loss distribution. At the same time, VAE is a bottleneck model, which can denoise the original data to a certain extent. The LVE model is shown in Fig. 5 below.

The model consists of three parts. The first module is data preprocessing. This part integrates unprocessed multi-source data and performs data cleaning, coding, integration and other steps into data features that can accurately reflect user behavior; the second module After the data is preprocessed, the processed data features are input to the Encoder, and the encoder part uses the LSTM model to process the data; the third module is to decode the obtained hidden variables into new data after resampling the data, through comparison The difference between vectors is judged abnormally.

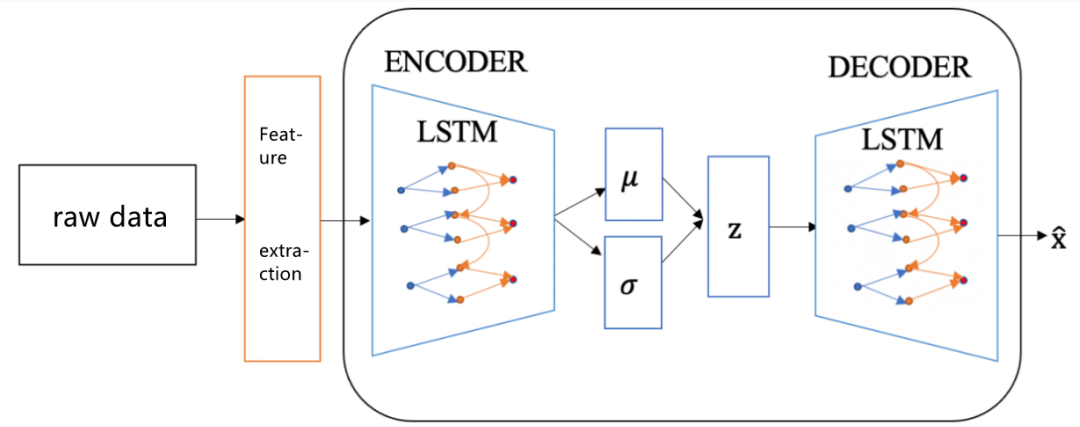

Fig. 5. Structure of LVE

For each sample of the input LVE, the probability decoder will find its probability distribution, and use their mean and variance to estimate the reconstruction data generated under the distribution. At the same time, because the model is an unsupervised learning method, there is no need for manually labeled labels in the data set, and the imbalance of positive and negative samples is allowed in the data.

In the training process, since LVE uses normal samples to train the model, it can make full use of the data context to optimize the encoder and generator of the model. According to the model, the LVE loss function mainly contains two parts, and the entire loss function is:

$$L = w_{enc} L_{enc} + w_{KL} L_{KL} \tag{6}$$

The neural network can learn normal behavior patterns after multiple iterations. When a new user's behavior is used as input, it is only necessary to send the data into the trained model to determine whether it is abnormal or not. The specific flow chart of this method is shown in Fig. 6 below.

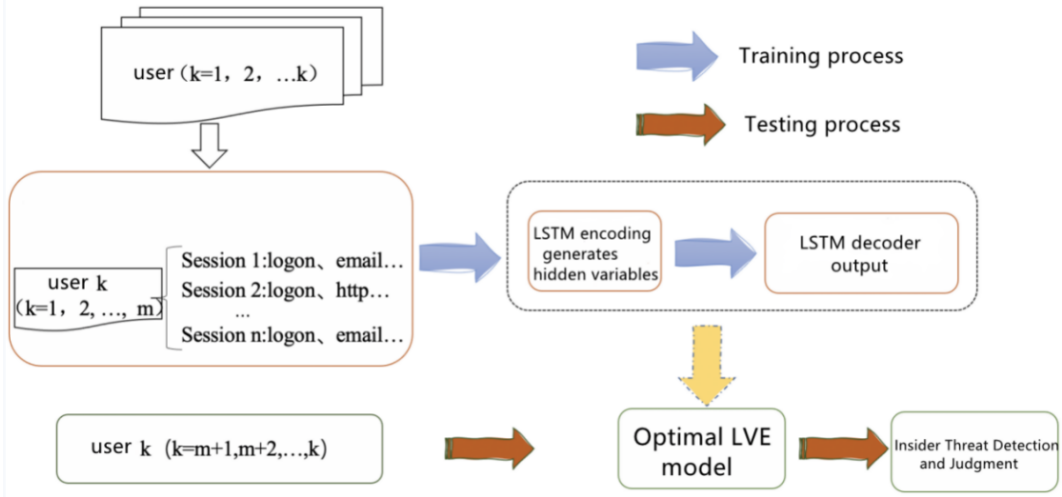

Fig. 6. Insider threat detection process of LVE

The LVE insider threat detection shown in Fig. 3-2 specifically includes 4 steps:

(1) Feature extraction: The 7 sub-data sources of CMU-CERT are combined to form a multi-domain audit log vector. First, the 16GB multi-domain audit log vector is preprocessed, and the processed data features are extracted into a 128-dimensional feature vector.

(2) Learning effective features based on the maximum training weight of the LVE encoder. The LVE encoder part is composed of three layers of LSTM, takes user data features as the input pre-training model of the LVE encoder, and calculates the mean and variance of the output of the last layer of LSTM network.

(3) The hidden variable obtained after resampling the output result of the LSTM encoder is used as the input of the LVE decoder, and then decoded into the output data similar to the input by the three-layer

LSTM decoder. The overall loss value formed by the error between the output data and the original input data and the KL divergence is back-propagated to adjust the parameters.

(4) Testing and testing stage. The test data set is also subjected to the feature extraction process shown in stage (1), and each piece of data in the test data is input into the trained LVE model, and the judgment is made according to the output result of the model.

4 Simulation Process and Result Analysis

4.1 Data Preprocessing

We introduced the 4.2 version of the CMU-CERT data set to evaluate the effectiveness of the LVE internal threat detection model. It is an internal threat data set generated by researchers from Carnegie Mellon University through artificial threat scenarios.

The original data in the CMU-CERTv4.2 internal threat data set is the behavior log of internal users, containing data from various sub-data sources, but it cannot be directly used as the input of the model, so further processing of the original data set is required.

(1) Data cleaning: In order to maintain the purity of the data, it is necessary to clean the CMU-CERT data set, modify the error information, and directly delete the duplicates for the redundant data.

(2) Data integration: Data integration is the process of merging different forms of data collected in an organizational environment containing various data sources for data storage.

(3) Data coding: After data cleaning and integration of the original data, the data can be divided into two categories: numerical data and categorical data.

4.2 Feature Extraction

Through the feature extraction of the collected data and the user context model, a data vector suitable for the input of the deep learning model is obtained. First, aggregate data from different sources according to user id based on given aggregation conditions, and then perform feature extraction on the aggregated data as shown in Fig. 7 below to generate a fixed-length vector that summarizes user operations.

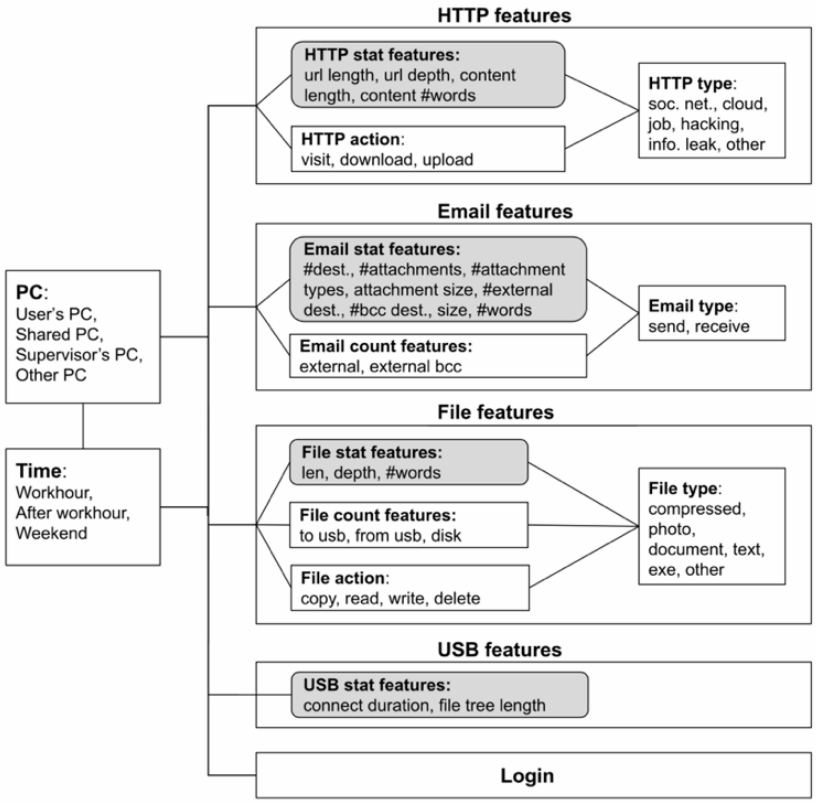

Fig. 7. Illustration of the feature extraction process

The overall process of feature extraction is shown in Table.1. In the process of feature extraction, the user's activities during the two days on weekends are not excluded, and the company's working hours are set from 7:30 in the morning to 5:30 in the evening. If the user connects to the driveable device during non-working hours, the behavior may be considered abnormal. The user is only allowed to work on the host assigned by the organization. If the user performs any operation on other hosts, the user and the host id does not match and the behavior is determined to be abnormal.

Table 1. Core steps of feature extraction of CMU-CERTv4.2

Overall feature extraction process
enter: CMU-CERTv4.2data
Output: 128-dimensional user features extracted according to user sessions
1. Combine data from sources by week,
2. Convert each action from DataByWeek to numerical data, stored in NumDataByWeek //Convert each activity in DataByWeek into numeric data and store it in NumDataByWeek
3. Extract to sessionr4.2.csv //Extract user behavior characteristics and save

Design the core steps of feature extraction for file files, email files, and http files, encode the file file type, and then generate the user's numerical features for one week. Finally, after extracting the full features of the CMU-CERTv4.2 internal threat data set, a total of 128-dimensional features. In this simulation, the description of the experimental environment is as follows:

System environment: macOS operating system

Hardware configuration: 2.6 GHz Intel Core i7 16G memory 256G hard disk

Experimental framework: Tensorflow deep learning framework

Development language: Python

Dependent libraries: numpy, matplotlib, sklearn, etc.

4.3 Evaluation Index

Among the evaluation methods of machine learning models, the most popular is the evaluation method based on confusion matrix. Its diagonal indicates the number of correct predictions.

Accuracy: In all data, the proportion of correctly predicted samples is calculated as follows.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{7}$$

Accuracy: The prediction result is the proportion of positive examples that are indeed positive examples. The calculation formula is as follows.

$$\Pr ecision = TP/(TP + FP) \tag{8}$$

Recall rate: the proportion of the samples whose real positive cases are predicted to be positive cases. The calculation formula is as follows.

$$Recall = TP/(TP + FN) \tag{9}$$

The precision rate and the recall rate contradict each other, and the compromise between the recall rate and the accuracy rate is used as an evaluation index to reflect the robustness of the model. The calculation formula is shown below.

$$F_1 - score = (2 \times \mathrm{Re} call \times \Pr ecision)/(\mathrm{Re} call + \Pr ecision) \tag{10}$$

Literature defines the daily budget (cumulative recall rate) in the article and uses it as an evaluation indicator. The daily budget is the sum of the recall rates of all budgets (including k). In this article, 20 is used as the increment, then $CR(k) = R(10) + R(30) + R(50) + \cdots + R(k)$.

4.4 Result Analysis

In the CMU-CERTv4.2 data set, 1,000 users generated 32,770,227 events in 502 days (73 weeks). Among these original incidents, 7,323 incidents were abnormal. Then the original data is processed into 470,611 feature vectors that can accurately reflect user behavior.

In this paper, the Adam optimizer is used to minimize the loss function. On the one hand, the loss function measures the reconstruction error of the sample through cross entropy, and on the other hand, measures the difference between the distribution of the hidden variable generated by the posterior probability and the standard Gaussian distribution through the KL divergence. The learning rate is set to 0.01 and all weight parameters are initialized to 0.0001.

As shown in Fig. 8, taking the daily budget as an evaluation indicator and comparing the results with the OCSVM algorithm, it can be seen that the LVE internal threat detection model proposed in this paper is significantly better than the traditional OCSVM algorithm. As shown in Table.2, this paper selects the classic machine learning algorithm and the LVE algorithm to compare the pros and cons.

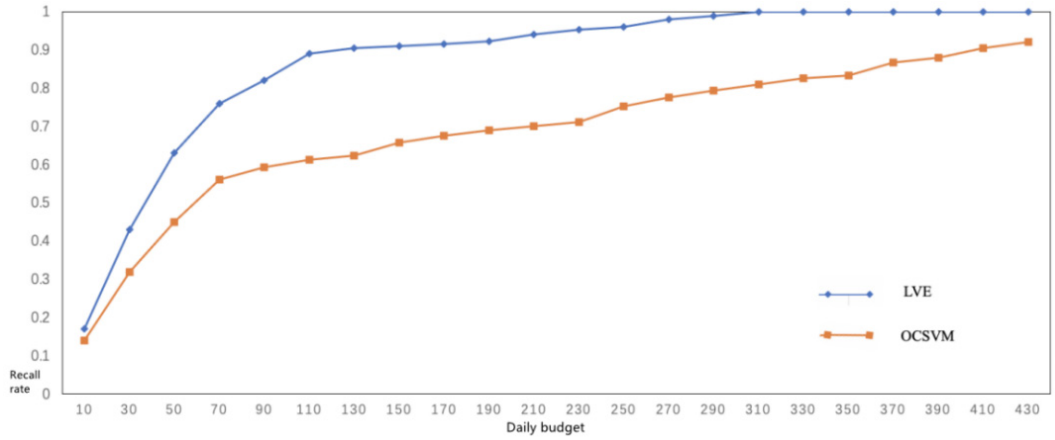

Fig. 8. Contrast of daily budget

Table 2. The recall rate when daily budget is 100

model	OCSVM	LVE
Recall rate	0.6123	0.8914

As shown in Table 4-2, it may be due to the timing characteristics in the data. The single-class support vector machine and the isolated forest algorithm are inferior to the LVE algorithm proposed in this article, but this also reflects the deep learning model in processing user data. Advantage.

5 Summary and Outlook

Internal attacks caused by improper operation of internal users can pose a serious threat to security. Considering its particularity, related technologies for external threat detection cannot be directly applied to internal threat detection. In addition, there are so many types of insider threat behaviors that it is impossible to explicitly model every threat behavior. In recent years, deep learning has been widely used in many fields, providing new ideas for insider threat detection technology. LSTM is a network structure suitable for processing time series data, and at the same time, VAE is an emerging data generation model, so this paper proposes an internal threat detection algorithm based on the LVE model. The simulation results show that the detection rate of this algorithm is higher, and it is more suitable for real scenes. The specific research work and results are as follows:

A global fine-grained feature extraction method is proposed to improve the efficiency of internal threat detection by increasing the number of examples extracted. In this method, a variational autoencoder is combined to propose an LVE internal threat detection algorithm. That is, the latent vector of the original data is generated by the LSTM encoder, so that the model retains the characteristics of the original data to the greatest extent, and the LSTM decoder is used to reconstruct the original features to optimize the LVE model. The simulation results show that compared with the traditional isolated forest method, this algorithm has a higher recall rate.

In the future, further research can be conducted in the following directions:

The representativeness and validity of the data set are critical to the model. Although the internal threat detection method proposed in this article has a certain effect, this article only uses the commonly used

CMU-CERT internal threat data set detection, and hopes to use the real data collected from the enterprise to detect the effectiveness of the LVE algorithm.

Acknowledgements

The research of the authors was supported by Industrial Internet innovation and development project of MIIT, China.

References

[1] R.E. Morgan, G. Kena, Criminal victimization, 2018, Bureau of Justice Statistics, 2019, NCJ 253043.

[2] D. Wang, Data breach: the unbearable weight of security, Collection, 2018, 3.

[3] T. Rashid, I. Agrafiotis, J.R.C. Nurse, A new take on detecting insider threats: exploring the use of hidden markov models, in: Proc. the 8th ACM CCS International Workshop on Managing Insider Security Threats, 2016.

[4] Q. Hu, B. Tang, D. Lin, Anomalous User Activity Detection in Enterprise Multi-Source Logs, in: Proc. 2017 17th Ieee International Conference on Data Mining Workshops, 2017.

[5] L. Liu, O. De Vel, C. Chen, J. Zhang, Anomaly-based insider threat detection using deep autoencoders, in: Proc. 2018 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE, 2018.

[6] M. Warkentin, R. Willison, A.C. Johnston, The Role of Perceptions of Organizational Injustice and Techniques of Neutralization in Forming Computer Abuse Intentions, in: Proc. 17th Americas Conference on Information Systems (AMCIS), 2011.

[7] D.M. Cappelli, A.P. Moore, R.F. Trzeciak, The CERT guide to insider threats: how to prevent, detect, and respond to information technology crimes (Theft, Sabotage, Fraud), Addison-Wesley, 2012.

[8] B. Wood, An insider threat model for adversary simulation, SRI International, Research on Mitigating the Insider Threat to Information Systems 2(2000) 1-3.

[9] D.B. Parker, Fighting computer crime: A new framework for protecting information, John Wiley & Sons, Inc., 1998.

[10] J.S. Park, S.M. Ho, Composite role-based monitoring (CRBM) for countering insider threats, in: Proc. Intelligence and Security Informatics, 2004.

[11] J. Shetty, J. Adibi, The Enron email dataset database schema and brief statistical report, Information sciences institute technical report, University of Southern California, 2004.

[12] J. Glasser, B. Lindauer, Bridging the gap: A pragmatic approach to generating insider threat data, in: Proc. 2013 IEEE Security and Privacy Workshops. IEEE, 2013.

[13] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, nature 323(6088)(1986) 533-536.

[14] D.P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv: 1312.6114, 2013.

[15] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, S. Robinson, Deep learning for unsupervised insider threat detection in structured cybersecurity data streams, arXiv preprint arXiv: 1710.00811, 2017.