A insider threat detection system based on user and entity behavior analysis

Haowei Liu1,a
1College of Computer and Information Security, Guilin University of Electronic Science and Technology, Majoring in Information Confrontation 541004, Guilin, Guangxi, China
a3118986590@qq.com

Abstract: Under the background of "digital new era", the trend of network environment diversification and personnel technical requirements complexity is becoming more and more intense. After the "Prism Gate" incident was exposed, the public began to think deeply about insider security. At present, most organizations adopt security information and event management (SIEM) security policies and the rules to carry out insider security detection. However, with the surge of insider information data, the number of false alarms and false alarms due to the lack of context increases, which consumes a lot of time and human and material resources. Based on these problems, it is particularly important to develop a new insider safety inspection system and tools. This work proposes to develop an insider threat detection system based on the security strategy of user and entity behavior analysis to realize the detection and analysis of insider threat with high precision. The main work is as follows: This work abandons the traditional SIEM combined rules to determine the anomaly, but adopts the detection strategy of User and Entity Behavior Analysis (UEBA). This work proposes an improved LSTM-GaN insider threat detection algorithm.

1. Introduction

1.1 Insider threat
Insider threat [1] refers to the threat posed to the organization by individuals with legal rights to have an access the internal systems of the organization, including IT breach, intellectual property theft and data fraud. Nowadays, internal threats are having a devastating effect on organizations around the world. The 2018 Insider Threat Report [2] survey shows that 90% of organizations indicate they are vulnerable to insider threats, and main factors include users with excessive access rights, the increasing number of devices that can have access to sensitive data, and the increasing complexity of information technology. In a 12-month period in 2020, 53 percent confirmed that their organization had been attacked from within. According to a U.S. Department of Justice survey, 74% of corporate cybertheft is committed by insiders. And of all cyber incidents reported by 36,000 US enterprises, 40% involved insider attacks [3].

1.2 User and entity behavior analysis
The traditional approach built on features, rules and manual analysis existed a blind spot of security visibility, so it could not automatically adapt to the escape of the attacker, and the policy upgrade often takes months, which has a serious lag effect, and it is even completely impossible to detect the unknown attack. In the case of this dilemma, the security industry turns to the new security paradigm on big data.
drive, security analysis and machine learning, so as to make up for the shortcomings of traditional security. User and entity behavior analysis (UEBA) is a typical new security paradigm using machine learning [4]. In behavioral analysis, a large number of basic data analysis techniques such as statistical analysis and time series analysis are adopted, as well as advanced analysis techniques such as unsupervised learning [5], supervised learning [6] and deep learning [7]. Through machine learning technology, the subtle points that human beings cannot perceive and recognize can be captured from behavioral data, and the behavior patterns and features that do not meet expectations hidden under the surface could be seized. Meanwhile, the behavior analysis driven by machine learning tends to avoid the difficulty and ineffectiveness of manual threshold setting.

2. Design and implementation

2.1 System architecture

As shown in Figure 1, through analysis and modeling of five types of behaviors of users and entities (PC login/log out, U disk access/disconnect, web page access, email communication and file access), portraits are constructed to achieve baseline. The baseline is used to predict the future behavior, and the deviation value is obtained by comparing with the subsequent behavior data, and the deviation value is employed as the indicator of the degree of abnormal behavior. In the UI detection interface, when the comprehensive index reaches a certain level, the person is regarded as abnormal, and the system alarms.

![System architecture](image)

Figure 1. System architecture

The model training is shown in Figure 2. This study investigates the log-in and log-out of equipment, web page access, use of hard disk drive, email sending, file access and 18-month LDAP data inside the company system. After that, improved LSTM-GAN algorithm [9] developed by the research is employed to train the data set, and the remote host (cloud server) GPU is used for model training. Traffic baselines are generated and predictions are made for the future.
2.2 Algorithm design

The main innovation is that after the GAN auto-encoder, another encoder is added, forming a "encode-decode-encode" structure, which evaluates whether the input behavior is abnormal behavior by comparing the output of the second encoder with the output of the first encoder. The training process of the network also bring in the idea of confrontation training, that is, a discriminator is adopted to judge whether the reconstruction behavior output by the decoder stems from the original behavior or the reconstruction behavior.

As shown in Figure 3, the whole network could be divided into three sub-modules: the auto-encoder module \((G_E, G_D)\), Encoder module \((E)\) and Discriminator module \((D)\). Initially, the input behavior \(x\) gets the eigenvector \(\hat{z}\) in the potential space through the encoder part \(G_E\) in the auto-encoder module, and then the reconstruction behavior \(\hat{x}\) gets through the decoder part \(G_D\). Next, the reconstructed behavior \(\hat{x}\) enters the encoder module \(E\) and obtains the reconstructed eigenvector \(\hat{z}\). Finally, the reconstructed behavior \(\hat{x}\) and the input behavior \(x\) are input into the discriminant \(D\) for judgment.

The first sub-network, shown in Figure 4, is an LSTM Auto Encoder, which is employed to reconstruct the behavior of the input. The Decoder [10] of the auto-encoder is almost identical to the generation network of the LSTM, mapping from a dimensional vector to a 3-channel behavior. The encoder part (Encoder1) of the auto-encoder is the inverse process of the encoder, which maps from a 3-channel behavior feature to a dimensional vector.

The second sub-network is a coding network (Encoder2) that compresses the reconstructed behavior of the first sub-network into a dimensional vector (Bottleneck2). Encoder2 takes the same structure as Encoder1, but the parameters are not identical. This structure abandoned the majority of anomaly detection method based on the encoder which compares the differences between the original behavior and reconstruction to infer abnormal way, and adopted a new method by comparing the original behavior and rebuild behavior in the higher layer abstract space difference to infer abnormal way. An additional layer of abstraction contributes to improve the ability of resisting noise greatly, and learn more robust anomaly detection models.
The third sub-network is a discriminant network (D-NET), which is adopted to distinguish the original behavior and the reconstruction behavior (the behavior generated by G-NET), that is, the original behavior is judged to be true, and the reconstruction behavior is judged to be false. Its structure is identical as the decoding network of the first sub-network. The introduction of D-NET is to recommend the idea of confrontation training, aiming at learning better G-NET.

3. Testing and analysis

3.1 Data preprocessing and feature extraction

| file        | Feature | Feature description                                                                 |
|-------------|---------|--------------------------------------------------------------------------------------|
| http.csv    | url     | The url accessed by the user                                                          |
|             | activity| User behavior on the site                                                             |
|             | hour    | Time spent by users visiting the web (in hours)                                       |
| file.csv    | activity| The type of action a user takes on a file on removable media                          |
|             | hour    | The time that the user operates on the file (in hours)                                |
| email.csv   | to      | Send email                                                                            |
|             | from    | The source of the received email                                                     |
|             | size    | The size of the email.                                                                |
|             | hour    | Time of operation                                                                    |

Data set features are described, as shown in Table 1.

3.2 Loss function

Figure 5(a) Generator loss function curve  Figure 5(b) discriminant loss function curve
The loss function test curves of GAN generator and discriminator are shown in Fig.5 (a) and Fig.5 (b).

### 3.3 Model test

During the whole training process, the input behavior is normal behavior, and the input behavior during the test has normal behavior and abnormal behavior. When the input behavior is abnormal behavior, the encoder $G_E$ encodes its information to obtain the characteristic vector $z$. However, since the network is trained only under the condition of normal behavior, the decoder $G_D$ does not have the ability to reconstruct the abnormal behavior $\tilde{x}$, and then the reconstructed behavior $\tilde{x}$ will lose the information of the abnormal part. When encoded again by the encoder $E$, the reconstructed feature vector obtained naturally does not contain abnormal information, consequently there existed obvious difference between $\tilde{z}$ and $z$, and the absolute value of this difference will be calculated to get the abnormal score of $\mathcal{A}(\tilde{x})$.

$$\mathcal{A}(\tilde{x}) = \|G_E(\tilde{x}) - E(G(x))\|$$  \hspace{1cm} (1)

For the entire test set $\tilde{D}$ can get an exception score set $S=${ $s_i$: $A(\tilde{x}_i)$, $\tilde{x}_i \in \tilde{D}$ }, The whole abnormal score set is normalized.

$$S'_i = \frac{s_{i} - \text{min}(S)}{\text{max}(S) - \text{min}(S)}$$  \hspace{1cm} (2)

The final normalized exception score set $S'$ would be obtained $S'$.

| Step length | 10  | 20  | 40  | 80  |
|-------------|-----|-----|-----|-----|
| hidden layers | 10  | 85.11% | 90.57% | 89.18% | 86.83% |
|             | 2   | 87.28% | 91.14% | 90.35% | 90.53% |
|             | 3   | 88.57% | 90.11% | 91.05% | 90.40% |
|             | 4   | 84.05% | 88.40% | 88.31% | 87.36% |

The experimental results are shown in Table 2. When the step size of hidden layer is set to 80 or 64, the model does not perform significantly better in one of the steps. However, when the step size is 20, the model always performs better and stably, winning or losing each other with the model of 40 steps. Combining the influence of the two parameters on the accuracy and considering the memory overhead, this study selects two hidden layers for the following experiment.

| Hidden layer size | accuracy |
|------------------|----------|
| 64               | 89.38%   |
| 128              | 91.14%   |
| 256              | 90.98%   |
| 512              | 89.27%   |

As shown in Table 3, the table shows the relationship between the size of the hidden layer and the accuracy. It could be seen that between 64 and 128, the accuracy goes up. After 128, the accuracy gradually decreases. The highest accuracy rate was 91.14%.
The relationship between the number of epochs and accuracy is shown in Figure 6. In deep learning, an epoch refers to the whole process of all training samples completing a forward propagation operation and a back propagation operation. Therefore, the number of data samples participating in the training of an epoch is the sum of training samples.

The ROC curve (AUC=0.92) is shown in Figure 7.

4. Conclusion
Extensibility has been considered in the development of this study, that is, the loose coupling method is adopted to design each module. After changing the data acquisition method, threat detection could be carried out according to different situations, such as expanding for other environments such as confidential enterprises or financial institutions. The next work will also provide exception detection for DDoS, XSS, or lateral movement attacks. For the further development of this work, for the normal operation and maintenance of the server, it can provide a more efficient and cost saving detection method for the security team of the government or enterprises.

References
[1]Hunker J, Probst C w. Insiders and Insider Threats: An Overview of Definitions and Mitigation Techniques[J]. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 2011,2(1):4-27.
[2]Institute of Security, China Academy of Information and Communications Technology. User entity behavior analysis technology(UEBA)Blue paper(2020)[R]. Hangzhou. Security Research
Institute, China Academy of Information and Communications Technology, 2020.

[3] Tian Yuan. Research on Internal Threat Detection Method Based on Deep Learning[D]. Beijing University of Technology, 2019.

[4] Cyber security Ideas and Crowd Research Partners. Insider threat 2018[R], 2018.

[5] Machine Learning: Investigators from Dalhousie University Release New Data on Machine Learning (Analyzing Data Granularity Levels for Insider Threat Detection Using Machine Learning)[J]. Journal of Engineering, 2020.

[6] LeCun Y, Bengio Y, Hinton GE. Deep Learning. Nature[J]. 2015, 521(7553): 436-444.

[7] Schmidhuber J. Deep Learning in Neural Networks: An Overview. Neural networks[J]. 2015, 61: 85-117.

[8] Bengio Y, Courville A, Vincent P. Represent action Learning: A Review and New Perspectives[C]. IEEE TPAMI, 2013, 35(8): 1798-1828.

[9] Yu Y, Canales S. Conditional LSTM-GAN for Melody Generation from Lyrics[J]. 2019.

[10] Jingxian Yang, Shuai Zhang, Yue Xiang, Jichun Liu, Junyong Liu, Xiaoyan Han, Fei Teng. LSTM auto-encoder based representative scenario generation method for hybrid hydro-PV power system[J]. IET Generation, Transmission & Distribution, 2020, 14(24).