An implementation of artificial neural networks into behavioral analysis system

E V Anashkin and M N Zhukova
Reshetnev Siberian State University of Science and Technology, Krasnoyarsk, Russia

E-mail: a.yegoriy@gmail.com

Abstract. The paper describes the architecture of the behavioural analysis system in development. This system is directed to inner threat countering in computer systems through user behaviour analysis. A part of system’s working algorithms is additionally presented together with the architecture. Also, next question is being decided - possibility of applying artificial neural networks for detection behavioural deviations instead comparing of statistical parameters.

1. Introduction
Machine learning in information security is mostly applied for pattern recognition and anomaly detection purposes [1]. Spam message, malware and network attack detection is hardly a complete list of tasks that can be solved with machine learning methods. Such tasks cannot be solved with a predetermined set of rules due to the fact that intruders continuously modify their toolkit and develop new attack methods.

This paper presents another relevant application of machine and deep learning methods – user behavior analysis. Individual characteristics of user behavior present a challenge for rule-based models of regulation.

External attacks protection is an important and necessary part of an integral security system. However, according to data leaks survey, 90% of all the leaks in Russia (63.5 in the World) involve an internal attacker [2]. Therefore, a shift in focus to methods of inner threat countering is a relevant and reasonable consequence. Application of user action analysis can provide a protection from internal attackers, insiders, compromised account usage, cloud data exfiltration and privileged account capabilities abuse.

Over the past 10 years, there have been a number of researches that touched on application of user behavior analysis in security. In the context of enterprise security, profiling of user behavior is implemented in the process of accessing the server [9]. In the research [10] it is proposed to implement fuzzy logic methods to calculate trust level for a user based on his behavior. Other applications of user behavior analysis are security audit and anomaly detection in databases [11]. This paper proposes to use a one-class support vector machine. Yet another research [13] focuses on protection of data stored in database. It aims to automate the processes of security policy and table access rules correction. To automate these processes, it proposes existing rules assigned to users by security administrator, and access patterns that define user behavior. Next research analyzes variable-length sequences of Unix system commands, that user enters to the terminal, using discrete-time Markov chains [12]. User behavior analysis is also used for malware detection purposes [14]: It allows defining the source of
files, the location that they were launched and how they affected the host’s security level and the precision of detection system. Nevertheless, in spite of the variety of researches related to behavior profiling, there is a shortage of works that would fully cover user behavior in desktop operating systems for workstations. With regard to this issue this paper proposes its own view of a system for profiling user actions in the Windows OS. The paper describes the behavioral analysis system in development and the methods of machine learning used in it.

2. System description
First High-level design of a system and some of its subsystems has been made. Behavioral analysis system might be split into the following subsystems:

- User action accumulation and monitoring subsystem;
- User profile building subsystem;
- Statistical anomaly detection subsystem;
- Heuristic anomaly detection subsystem;
- Deductive analysis subsystem;
- Interagent cooperation subsystem.

2.1. Notification (alarm) subsystem
User action accumulation and monitoring subsystem constantly registers all the user’s actions associated with the events registered by OS. During the process of learning, the subsystem accumulates the data, and during the functioning process its main function is monitoring. The data about actions is saved after the classification of an action as normal or anomalous.

User profile building is not a one-time action that fixes the profile permanently, it is a constant process. User’s behavior can change over time, which leads to the need of profile correction (rebuilding). What is more, profile can be rebuilt with a certain temporal discretization: one-day profile, one-week profile, one-month profile or a profile for a certain date etc. Right after the end of a learning period subsystem must build a base behavior profile. During functioning period, the subsystem must constantly evaluate current user profile.

Statistical anomaly detection subsystem must compare current profile with the base profile to detect statistical deviations.

Heuristic anomaly detection subsystem does not analyze user’s profile; it analyzes combinations of occurring actions. A certain sequence of actions (events) might be an attack marker (malicious actions). Malicious activity must be classified as anomaly.

When statistical anomaly detection subsystem detects an anomaly (deviation), deductive analysis subsystem starts. This subsystem accumulates and analyzes contextual information and concludes whether the deviation is anomalous or not.

Interagent cooperation subsystem’s purpose is to transmit information about the current situation between the adjacent (neighboring) agents. This information forms the situational context.

Notification subsystem generates notifications whenever deductive analysis subsystem concludes that a deviation is an anomaly, or heuristic anomaly detection subsystem finds an attack marker. What at first seems like an easy subsystem gets more complicated with the variability of forms notifications can take (desktop message, email message etc.) and the need to journal every notification.

2.2. Base algorithm
Base algorithm of behavioral analysis system consists of two consequent stages: learning period and functioning period. In the duration of learning period, the system collects statistical data about user’s actions. After the end of the learning period user’s base behavioral profile is formed from the accumulated statistical data. After this the system switches to functioning mode. Functioning mode’s algorithm is shown on figure 1.
The showed algorithm involves statistical detection of deviations from base profile, heuristic analysis of action combinations, and also deductive contextual analysis. In cases, when the algorithm of deductive contextual analysis cannot exactly classify occurred action, expert's knowledges can be required. Deductive rules are created and added to relevant database after expert's help or when the algorithm makes its own decision. A set of predefined rules can be added before system startup.

3. **Contextual information**

Classification of detected deviations, potentially dangerous or not, must be based not only on behavioral statistics, but also on additional contextual information. Otherwise, such system will be burdened with frequent false alarms.

The six types of contexts were determined:

- Personal context;
- Temporal context;
- Problem’s source (causative) context;
- Security context;
Personal context is formed from the statistical data about user’s actions. It contains a data about user’s normal workflow and a list of programs that user needs to do his work etc.

Temporal context is based on time and date of the current events. For example, if a user’s action occurs after the end of his working hours, the action is determined as anomalous. From the research [15] it’s worth to point out that user’s first access to a system, the time of his first authentication, is widely used as an indicator of malicious activity.

Causative context must address the question “Why this action was made? What is the purpose?”

Security context is formed from information taken from other sources that are dealing with information security management. For example, information security facilities. AVZ notification, the result of a scan made by vulnerability scanner etc. can be classified as context forming information.

Group context is a combination (averaging) of personal statistics of users belonging to one group.

Situational context is formed from information transmitted from other adjacent agents about detected deviations in user’s actions.

Registration of all received contexts must be handled by deductive analysis subsystem’s algorithm. Figure 2 illustrates approximate scheme of the algorithm that processes such action as user launching previously unknown program (a program that has never been registered in user’s profile).

Figure 2. An example of deductive contextual analysis.

For semantic comparison It is proposed to use either one of two solutions: ontology system or word2vec toolkit [7], based on vector representation of words and artificial neural networks [8].

Following the first step, a practical implementation of the first subsystem – user action accumulation and monitoring subsystem has been made. To implement the subsystem meant to develop an agent program that would function as a background process.

It was proposed to split the accumulated data about user actions into different layers. For example, D. Dasgupta in his research [16] on Immuno-inspired autonomic system for cyber defense proposes a multilayer profiling that consists of:
• Application layer that monitors all deviations from user behavioral framework;
• System layer that monitors the resources in use;
• Process layer that monitors all illegitimate processes and process priority violations;
• Packet layer that monitors network traffic (amount and size of network packages, its source field and protocol type).

Regarding the research, it can be concluded that there has to be a clear distinction between objects and subjects of profiling. As such, in the process of user profiling there is no need to take into consideration such system characteristics as CPU time, memory, priorities etc as they belong to another profiling subject – process. Table 1 contains all the layers and their attributes that were selected to be monitored.

Table 1. The analyzed levels and attributes.

| Layer name                                  | Attributes                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------|
| Application layer                           | Date, time, username (ID), PC name, process name.                          |
| System working schedule layer               | Date, time, username (ID), PC name, action, address, action result (success or not) |
| External data storage device interaction layer | Date, time, device name, device ID                                         |
| File system layer                           | Date, time username (ID), PC name, filename (full path), action (FS operation). |
| Network activity layer                      | Date, time, source IP, destination IP, protocol, source port, destination port, packet size |

4. Artificial Neural network application

The most important requirements for the algorithms applicable for user action analysis are the ability to self-learn and to adapt [3]. Bioinspired algorithms (models) meet these requirements, for example, artificial neural networks and artificial immune networks.

Exabeam experts present user behavioral profile as a set histograms built from historical data (statistics) [5]. A hypothesis has been proposed that it is possible to replace a process of building a profile statistical model with artificial neural network that accepts as an input encoded data about user actions. Artificial neural network’s application for detection of suspicious activity can be found in the research [17]. In it authors use 4 defining attributes:

• The system in use;
• Location;
• Date and time;
• Session duration.

There are more data layers and attributes in this research. Artificial neural network implementation can also be found in UEBA (User and Entity Behavior Analytics) class solution made by Niara (renamed Aruba), where user’s actions are encoded into a picture where event intensity is defined by color shade [6]. These pictures are processed by neural network with the purpose of anomaly detection. However, picture processing is more resource-intense than text processing. Therefore, it is decided to test the hypothesis using text data.

The application layer data about user’s actions for one-day period was used to test the hypothesis. If an artificial neural network will be well-learned on this small amount of input data then this method will be used in the future.

A single record about user’s actions on application layer contains following entries:
• Account name;
• Date (day, month, year);
• Time (hours, minutes, seconds);
• PC identifier;
• Executable file full name.

Given that this data is categorical and a part of it is in text form, there is a need to transform it. To transform the data 3 different approaches are used:

• Ordinal encoding with normalization [0;1];
• One-hot encoding;
• Binary encoding.

The difficulty of using artificial neural networks comes from the fact that encoded data represents only one class – legitimate user while neural networks need at least 2. Two solutions were selected to remove this limitation:

• Data sample generation for the 2nd class;
• Use of “auto encoder” neural network architecture [4].

To generate examples of anomalous activity regulated parameters are introduced. They are used to set activity windows. For example, there is a 30-minute window. If an action occurs half an hour before or after than usual then this action is still considered normal. When action happens earlier or later than half an hour, the action is considered anomalous.

Deciding artificial neural network’s architecture is a nontrivial task, especially if it needs to be optimal. This question has been researched for a long time. It concerns such problems as defining the number of hidden layers [18] and the number of neurons [19][20] in them for best results. For example, research [20] presents a formula that allows to calculate the number of neurons for hidden layer that depends not only on the size of input and output layers but also on the amount of training examples. However, at this point this research is not aimed at deciding the optimal network architecture. And the amount of training examples is hard to predict before the end of statistics accumulation (each user does different amount of actions). Instead this work was aimed at testing the hypothesis that neural network can be used for detecting behavior deviations. Therefore, the decision on the number of neurons in the hidden layer was taken at random, most often doubling the amount layer after layer. Artificial neural network of each architecture was started 10 times. After that, average values were calculated for each network architecture. A separate sample was made to rate the precision of networks. Rel.U was used as an activation function [21].

Table 2 contains experiment results with classification neural networks, where LS – learning sample, TS - test sample. In the architecture column numbers define the number of neurons and a “-“ (hyphen) separates different layers. The first layer is the input layer and the last layer is the output layer.

Table 2. The results of the experiment with classifying ANN.

| Neural network architecture | Number of learning epochs | 100 | 1000 |
|-----------------------------|---------------------------|-----|------|
|                             |                           | LS  | Precision | LS  | Precision | LS  | Precision | LS  | Precision | TS  | Deviation |
| 5-5-5-1                     | Ordinal encoding + input data normalization | 0.545820 | 0.721440 | 0.4 | 0.111620 | 0.964300 | 0.4 |
| 5-5-5-1                     |                           | 0.563140 | 0.714300 | 0.3 | 0.155140 | 0.957160 | 0.3 |
Table 3 contains experiment results with auto encoder architecture. To classify a network, an average deviation of output data from input data has to be calculated. Afterwards the result number is compared with coefficient k. A deviation that is bigger than k is an anomaly (intruder action), less or equal than k – legitimate user.

| Neural network architecture | 100 Learning epochs | 1000 Learning epochs |
|-----------------------------|---------------------|----------------------|
|                            | LS deviation | TS deviation | LS deviation | TS deviation |
|                            | k=0.2 | k=0.1 | k=0.05 | k=0.2 | k=0.1 | k=0.05 |
| Ordinal encoding + input data normalization | | | | | | |
| 5-5-3-5                    | 0.01620 | 0.3 | 0.2 | 0.5 | 0.00390 | 0.3 | 0.3 | 0.6 |
| 5-20-10-5                  | 0.00700 | 0.3 | 0.2 | 0.5 | 0.00034 | 0.4 | 0.4 | 0.2 |
| 5-50-25-25-50-5            | 0.00410 | 0.3 | 0.3 | 0.3 | 0.00027 | 0.4 | 0.3 | 0.2 |
| One-hot input data encoding | | | | | | |
| 126-126-63-126             | 0.06650 | 0.3 | 0.5 | 0.5 | 0.01070 | 0.5 | 0.3 | 0.4 |
| 126-252-126-63-30-15-30-63-126 | 0.06720 | 0.3 | 0.4 | 0.5 | 0.01070 | 0.5 | 0.3 | 0.4 |
| 126-20-10-126              | 0.07330 | 0.3 | 0.5 | 0.5 | 0.06870 | 0.3 | 0.5 | 0.5 |
| Binary input data encoding | | | | | | |
| 22-44-22-11-22             | 0.27150 | 0.5 | 0.5 | 0.5 | 0.03620 | 0.3 | 0.3 | 0.3 |
| 22-22-11-12-22             | 0.27450 | 0.5 | 0.5 | 0.5 | 0.03100 | 0.3 | 0.3 | 0.4 |

Table 3. The results of the experiment with ANN type "autoencoder".

Figure 3. Taught neural network’s precision rate on test sample.
Despite the small amount of learning examples some neural network architectures demonstrated high rate of precision. Classification neural networks showed higher precision compared to “auto encoders” and other methods of anomaly detection. This fact is demonstrated in figure 3. The shortcoming of classifying neural networks is the need to generate anomalous activity examples for a learning sample.

5. Conclusion and future work
This research presents the results of workstation user behavioral analysis system’s design process. Instead of user profile building it is proposed to use artificial neural networks. As a learning sample it is proposed to use user’s actions. Proposed solution shows high precision rates but it needs to be taught with examples of anomalous activity. In case of availability of means to automate the generation process of such examples this solution is acceptable. Otherwise, the need of manual tuning of additional parameters renders this solution useless.

The question that remains is a way all data layers could be united into one. This research describes the implementation of artificial neural network for application layer only. But according to the design there are 4 more layers to be concerned with. The question remains whether the system consists of five neural networks responsible for each layer, or it is possible to develop a universal artificial neural network architecture that is capable of accepting and processing data of all five data layers. The answer to the question is to be answered in future researches.

In future researches it is planned focus more on deductive analysis subsystem and its practical implementation. Therefore, it is planned to test two different solutions (approaches): ontology and word2sec toolkit.

References
[1] Chio C and Freeman D 2018 Machine Learning and Security (Sebastopol, O'Reilly Media) pp 12-3
[2] InfoWatch. Utechki dannyh. Rossiya. 2018 god Retrieved from: https://www.infowatch.ru/resources/analytics/reports/russia2018
[3] Anashkin E 2019 To the question of algorithms choice in tasks of user behavior analysis Aktual’nye problemy aviantii i kosmonavtiki
[4] Hawkins S, He H., Williams G and Baxter R 2002 Outlier Detection Using Replicator Neural Networks Data Warehousing and Knowledge Discovery pp 113-23
[5] Lin D and Kladko L 2017 User Behavior Analytics for Cyber Security and Its Implementation In Scala Conf. ScalaUA2017
[6] Wang J 2017 Deep Learning in Security—An Empirical Example in User and Entity Behavior Analytics (UEBA) Conf. Spark Summit 2017
[7] Wor2Vec Retrieved from: https://code.google.com/archive/p/word2vec/
[8] Mikolov T, Chen K, Corrado G and Dean J 2013 Efficient Estimation of Word Representations in Vector Space Conf. Proceedings of the International Conference on Learning Representations
[9] Shashanka M, Shen M and Wang J 2016 User and Entity Behavior Analytics for Enterprise Security IEEE International Conference on Big Data (Big Data) pp 1867-74
[10] Alruwaythi M and Nygard K 2019 Fuzzy logic Approach Based on User behavior Trust in Cloud Security Conf. IEEE International Conference on Electro Information Technology (EIT)
[11] Li Y and Zhang T 2016 Anomaly Detection of User Behavior for Database Security Audit Based on OCSVM 3rd International Conference on Information Science and Control Engineering pp 214-9
[12] Xi X, Shu-tao X, Xin-guang T and Qi-bin Z 2011 Anomaly detection of user behavior based on
DTMC with states of variable-length sequences The Journal of China Universities of Posts and Telecommunications 18(6) 106-15

[13] Ghazinour K and Ghayoumi M 2015 An Autonomous Model to Enforce Security Policies Based on User’s Behavior Conf. 14th International Conference on Computer and Information Science (ICIS) (Las-Vegas)

[14] Yang F, Wu J, Tang S and Zhang H 2013 Dynamic Knowledge Repository-based Security Auxiliary System of User behavior Conf. IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber,Physical and Social Computing

[15] Tang B, Hu Q and Lin D 2017 Reducing False Positives Of User-to-Entity First-Access Alerts for User Behavior Analytics Conf. IEEE International Conference on Data Mining Workshops pp 804-11

[16] Dasgupta D 2007 Immuno-inspired autonomic system for cyber defense Information security technical report vol. 12 pp 235–41

[17] Ussath M, Jaeger D, Cheng F and Meinel C 2017 Identifying Suspicious User Behavior with Neural Networks Conf. IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud) (New-York)

[18] Thomas A, Petridis M, Walters S, Ghayouss S and Morgan R 2017 Two Hidden Layers are Usually Better than One Engineering Applications of Neural Networks: 18th International Conference (EANN 2017) pp 279-90

[19] Lawrence S, Giles C and Tsoi A 1996 What Size Neural Network Gives Optimal Generalization? Convergence Properties of Backpropagation Technical Reports from UMIACS (Maryland)

[20] Eliseeff A and Paugam-Moisy H 1996 Size of multilayer networks for exact learning: analytic approach Advances in Neural Information Processing Systems vol. 9 pp 162-8

[21] Glorot X, Bordes A and Bengio Y 2011 Deep Sparse Rectifier Neural Networks Conf. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (Ft. Lauderdale)