AGENT-BASED ANOMALIES MONITORING IN DISTRIBUTED SYSTEMS

Andrii Shelestov

Abstract. In this paper an agent-based approach for anomalies monitoring in distributed systems such as computer networks, or Grid systems is proposed. This approach envisages on-line and off-line monitoring in order to analyze users' activity. On-line monitoring is carried in real time, and is used to predict user actions. Off-line monitoring is done after the user has ended his work, and is based on the analysis of statistical information obtained during user's work. In both cases neural networks are used in order to predict user actions and to distinguish normal and anomalous user behavior.

Keywords: security, distributed systems, agent approach, neural networks.

ACM Classification Keywords: K.6.5 Security and Protection – Authentication, I.2.6 Learning - Connectionism and neural nets, I.2.11 Distributed Artificial Intelligence - Multiagent systems.

1 Introduction

Nowadays it is practically impossible to imagine different areas of human activity without the use of distributed systems, for example, corporate computer networks, Grid systems [1] for complex scientific problems solving, etc. However, it is evident that the work of many organizations (or set of organizations) considerably depends upon effective use of distributed systems resources and the level of their protection. Many problems, such as data storage, data transfer, information processing automation, complex problems solving are entrusted on them. The security level of information used in distributed systems can vary from private and business to military and state secret. The violation of information confidentiality, integrity and accessibility may have significant and undesirable consequences to its owner. Besides, many sources report that the majority (80%) of information security incidents is perpetrated by insiders (Microsoft Encyclopedia of Security, 2003) [2]. This means that internal computer users constitute the largest threat to the computer systems security.

Unfortunately, traditional methods (such as identification and authentication, access restriction, etc.) are not seemed to solve this problem at all. These rigorous and deterministic approaches possess some drawbacks; among them are low ability of internal malicious users detection, inability to process large amounts of information, low productivity, etc. That is why new approaches for users activity monitoring (including those relying on intelligent methods) are applied.

We may consider so called Personal Security Programs that are used by commercial companies to monitor the activity of their employees. The results of such monitoring can be used to reveal malicious users in the case of information leakage, or to find out whether users use computers for their personal purposes. For example, such programs as PC Spy (www.softdd.com/pcspy/index.htm), Inlook Express (www.jungle-monkey.com), Paparazzi (www.industar.net) allow to capture and save screen images (screenshots) showing exactly what was being viewed by users. All screens can be captured, including Web pages, chat windows, email windows, and anything else shown on the monitor. However, these programs have some disadvantages; among them are high volume of stored information and manual configuration of snapshots frequency.

Another example refers to Intrusion Detection Systems (IDS), particularly anomaly detection in computer systems. Usually, a model of normal user behavior is firstly created, so during monitoring any abnormal activity can be regarded as potential intrusion, or anomaly. Different approaches are applied to the development of anomaly detection systems: statistical methods [3], expert systems [4], finite automata [5], neural networks [6-8], agent-based systems [9], etc.

Generally, the development of monitoring system involves two phases: creation of user behavior model (normal or usual) and system implementation. First phase involves the following steps: data collection and data pre-processing, when useful information about user activity is collected from log-files; data processing, when feature extraction is made to data representation and dimension reduction methods are used to reduce the size of the data; application of different techniques to obtain interesting characteristics of users' behavior; interpretation of

the results. During the implementation phase it should be taken into account the distributed and heterogeneous nature of distributed systems and a great number of users in it. Therefore, it is advisable to provide an autonomous module for each user behavior model developed within the first phase. Moreover, in some cases this module has to move in the system since the user can work on different workstations (computers). Thus, the monitoring system has to be distributed and scalable, it should enable the work with different operating systems and data formats, it should have independent modules to enable autonomy and mobility. To meet these requirements, agent technology represents the most appropriate way [10-11].

In this paper we present an agent-based approach for anomalies monitoring in distributed systems. This approach envisages on-line and off-line monitoring that enables the detection of anomalies and irregularities in users' behavior. On-line monitoring is carried in real time, and is used to predict user actions. For this purpose, we use feed-forward neural networks [12]. Off-line monitoring is done after the user has ended his work, and is based on the analysis of statistical information obtained during user's work. We use neural network as classifier to distinguish normal and anomalous user behavior. The use of on-line and off-line monitoring allows one to reflect both dynamical and statistical features of user's activity. Considering system implementation, we use Java programming language and Aglets Software Development Kit (ASDK) for the development of mobile agents.

2 Agent Paradigm

The main point about agents is that they are autonomous, i.e. capable of acting independently. An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors [10]. That is, the agent can be characterized by the following set:

$$<S, Prog, Eff, Arch, P, A, G, E> \quad (1)$$

where E defines the environment where agent works; S — sensors through which it perceives information from environment; Eff — effectors through which agent can act on environment; P — what kind of information agent can perceive from its sensors; A — what kind of actions agent can make using its effectors; Prog (program) Prog: P–>A — defines agent's response to its percepts; G — goal the agent trying to reach; Arch — agent's architecture.

Main agents' properties are the following ones [13]: autonomy, reactivity (provides an ongoing interaction with its environment, and responds to changes that occur in it), proactiveness (means goal directed behavior of agent), social ability (ability to interact with other agents via some kind of agent-communication language, and perhaps co-operate with others), mobility (the ability of an agent to move around an electronic network), rationality (agent will act in order to achieve its goals), learning/adaptation (agents improve performance over time).

In this paper, software agents will be used for implementation of intelligent security system. In general, they represent computer programs and act in computer systems. Thus, according to (1) for software agent we have — E=computer system, Arch=program code, S and Eff represent some functions (or, in general case, programs) through which agent can interact with environment.

3 System Architecture and Functionality

The proposed intelligent security system for users' activity monitoring in distributed systems consists of the following components (Fig. 1):

— On-line User Agent that provides on-line monitoring,

— Off-line User Agent that provides off-line monitoring,

— Controller Agent that manages other agents,

— Database.

On-line User Agent. This agent is functioning in real time with aim to detect anomalies and irregularities in computer users' activity. It predicts user actions on the basis of previous ones. For this purpose a neural network is used. The output of the neural network is compared to real actions performed by user. If the relative number of correctly predicted actions larger than specified threshold, then it can be assumed that the user behavior is normal. Otherwise, it is abnormal. Additionally, this agent collects information about user's activity and stores it in

database. This type of agents should be constructed for different operating systems used in computer system (e.g. Win2K/XP, Win98, FreeBSD).

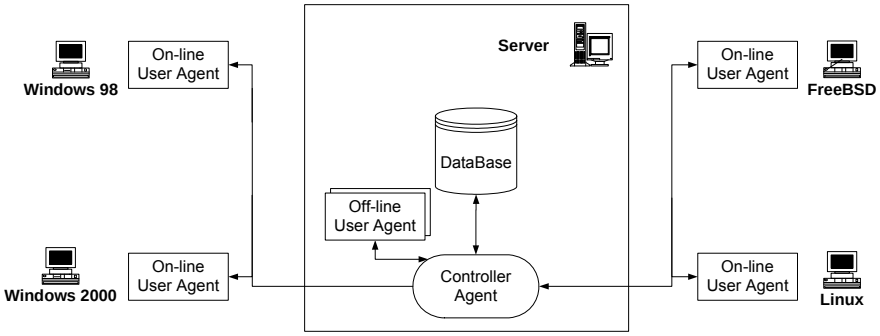


Fig. 1. System architecture

Off-line User Agent. This agent works off-line (i.e. after the user has ended his work) and tries to detect anomalies in the user activity on the basis of statistical parameters (user signature). The following set of characteristics about user behavior were taken as user signature: the set of processes (number of processes started by user), results of on-line agent functioning (number of correctly predicted processes by On-line User Agent), user login host (the set of hosts from which user logs on), user session time (the session duration for the user), user activity time (the time of user session starting). For each user its own Off-line User Agent is created based on feed-forward neural network. The network is trained in order to distinguish normal and abnormal user behavior.

Controller Agent. This agent is responsible for overall system functioning, agents initializing and coordination, and interaction with database.

Database. Contains data that is needed for system functioning.

When the user logs on (that is, begins his work on computer), Controller Agent creates corresponding On-line User Agent and initializes it. On-line User Agent gets data about specified user from a database and moves to the computer where the user works. During the user's session, this agent monitors user's activity by predicting his actions (using neural network) and comparing them to real ones. If the relative number of correctly predicted actions larger than specified threshold, then it can be assumed that user behavior is normal and corresponds to the previously built model. Otherwise, user behavior is assumed to be abnormal. In the case of anomaly detection On-line User Agent informs Controller Agent about suspicious activity. When user finishes his work, On-line User Agent is destroyed.

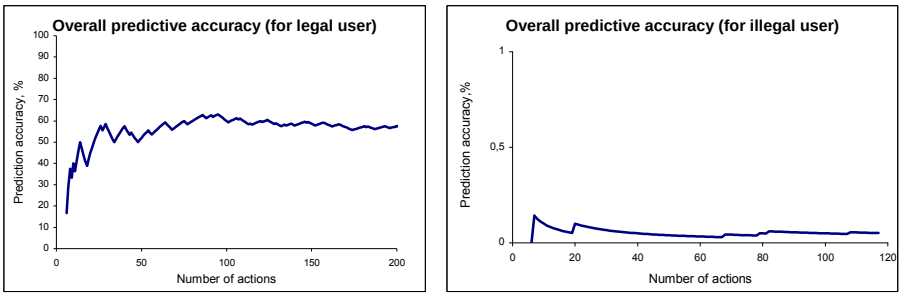
At the end of the day (when the system load is low) Controller Agent initializes Off-line User Agent. On the basis of data obtained from On-line User Agent it detects if the user activity was normal or abnormal. In the case of abnormal activity (i.e. it had anomalies) Off-line User Agent informs Controller Agent about it.

4 Description of Experiments

Different experiments were run to demonstrate the efficiency of both On-line User Agent and Off-line User Agent. Since both types of agents are based on the use of neural networks data needed for neural network training were obtained during real work of users in the Space Research Institute NASU-NSAU. For this purpose special software was developed to get data about users' activity.

For On-line User Agent log files were transformed into format suitable for neural network. That is, for each user an alphabet of actions (processes) was created, and each action was assigned an identifier (decimal number). For neural network input a binary coding was applied (7 bits for each command). Feed-forward neural network trained by means of error back-propagation algorithm [12] was used in order to predict user action on the basis of 5 previous ones. Thus, the dimension of input data space for neural network was 35. In turn, for output data decimal coding was applied, and the dimension of output data space was 1. As to neural network architecture, we used neural network with 3 layers: input layer with 35 neurons, hidden layer with 35 neurons, and output layer with 1 neuron.

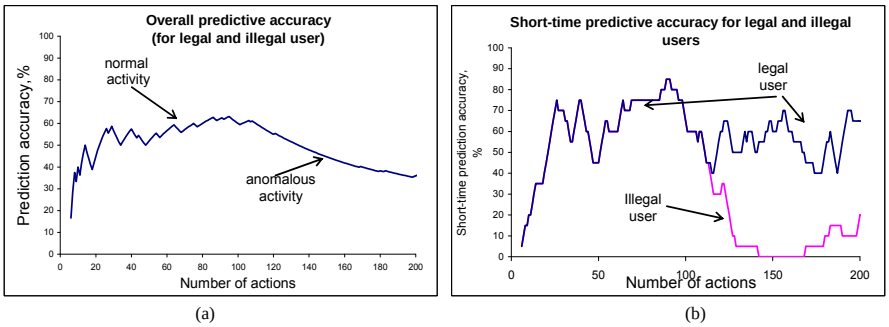
Then all data were randomly mixed and divided into training and test sets (70% for training and 30% for testing). Results of neural network work on test data showed that overall predictive accuracy (that is, the number of correctly predicted commands divided by total number) for different users varied from 33% to 59% (an example of overall predictive accuracy variations within number of user actions is depicted on Fig. 2,a). But the main point in constructing On-line User Agents is to ensure that they differ for different users. That is, the efficiency of On-line User Agent should be viewed not in the term of absolute value of the predictive accuracy for the user, but relative to other users. In order to demonstrate that the neural network was able to distinguish one user from another we run so called cross experiments. Two types of cross experiments were implemented. First one consisted in the following: the data obtained during the work of one user (name him illegal user) were put to neural network that was trained for another (legal user). In such a case, overall predictive accuracy of neural network hardly exceeded 5% (on Fig. 2,b it is shown an example where overall predictive accuracy was 0,05%). That is, the overall predictive accuracy decreased, at least, six times for illegal user. Such experiment modeled the situation when illegal user logged on and begun to work under the account of another user.



(a) (b)

Fig. 2. Overall predictive accuracy for: (a) legal user; (b) illegal user

The second type of cross experiments was carried out by inserting the data of illegal user into the data of legal one. This experiment modeled the situation when an intruder begun to work under the account of another user already logged on. In a such case, the overall predictive accuracy begun to decrease constantly, as shown on Fig.3,a. Another measure that can be used to distinguish normal and anomalous user activity is a short-time predictive accuracy. To estimate the short-time predictive accuracy we took into considerations only last actions performed by the user but not all (for example, twenty last actions). Variations of short-time predictive accuracy for both legal and illegal user are shown on Fig. 3,b. From figure it is evident that the short-time predictive accuracy for illegal user begun to decrease.



(a) (b)

Fig. 3. Predictive accuracy for: (a) overall; (b) short-time

Therefore, experimental results showed the ability of neural networks to distinguish confidently normal and abnormal (anomalous) user behavior.

As with On-line User Agent, all data needed for Off-line User Agent were obtained from the log files. Then the data were encoded, divided into training and test sets, and input to neural network. Results of neural network work on test data gave 80% accuracy of correct user behavior classification. That is, experiments showed that

Off-line User Agent was able to distinguish normal and abnormal (anomalous) user behavior. Additionally, Off-line User Agent can be used to verify the work of On-line User Agent.

5 System Implementation

The proposed agent-based system was implemented using mobile agents. Java language and Aglets Software Development Kit (ASDK) were chosen, respectively, as programming language and environment for mobile agents development. Java offers the set of unique features that allows one to simplify the development of multi-agent systems. The following properties of Java should be mentioned: platform independence; secure code execution; dynamic class loading; multithreading programming; object serialization.

ASDK is a free-ware software, provided by IBM. It enables the development of mobile agents that are called aglets (http://sourceforge.net/projects/aglets/). The following properties of ASDK could be mentioned: the use of special MASIF (Mobile Agent System Interoperability Facility) standard which allows various agent systems to interoperate; the use of ATP (Agent Transfer Protocol) protocol that represents a simple application-level protocol designed to transmit an agent in an agent-system-independent manner; mobility of agents; the use of Java security policy (JDK keytool).

In general, aglets are Java objects that can move from one host on the network to another. That is, an aglet that is run on one host can suddenly halt execution, dispatch to a remote host, and start executing again. When the aglet moves, it takes along its program code as well as the states of all the objects it is carrying. A built-in special security mechanism makes it safe to host untrusted aglets.

Proposed intelligent security system was implemented based on client/server architecture. Server side represented a special platform which was used for the creation of agents and its hosting (all agents used in the system are initiated on server side), for database interaction, requests redirection. Client side is responsible for agent functioning on user computers. Among its functions are support of agents hosting and information logging about user activity. Additionally, special user interface was developed that shows information about user logged on, operating system that is used, client platform parameters, and information about On-line User Agent work.

6 Conclusions

The proposed system takes advantages of both intelligent methods for monitoring of user activity and multi-agent approach. To reflect both dynamical and statistical parameters of user behavior on-line and off-line monitoring is done. The use of neural network provides adaptive and robust approach for the analysis and generalization of data obtained during user activity. The use of multi-agent approach is motivated by the system functioning in heterogeneous environment, and by processing data in different operating systems.

Acknowledgments

This research is supported by INTAS-CNES-NSAU project “Data Fusion Grid Infrastructure”, Ref. No 06-1000024-9154..

Bibliography

[1] Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. Int. J. Supercomputer Applications 15(3) (2001).

[2] Tulloch, M.: Microsoft Encyclopedia of Security. Redmond, Washington: Microsoft Press (2003) 414 p.

[3] Javitz, H., Valdes, A.: The SRI IDES statistical anomaly detector. In: Proc. IEEE Symp. on Research in Security and Privacy (1991) 316–326.

[4] Dowell, C., Ramstedt, P.: The ComputerWatch data reduction tool. In: Proc. 13th National Computer Security Conf. (1990) 99–108.

[5] Kussul, N., Sokolov, A.: Adaptive Anomaly Detection of Computer System User's Behavior Applying Markovian Chains with Variable Memory Length. Part I. Adaptive Model of Markovian Chains with Variable Memory Length. J. of Automation and Information Sciences Vol. 35 Issue 6 (2003).

[6] Ryan, J., Lin M-J., Miikkulainen, R.: Intrusion Detection with Neural Networks. In: Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press (1998) 943–949.

[7] Reznik, A, Kussul, N., Sokolov, A.: Identification of user activity using neural networks. Cybernetics and computer techniques, vol. 123 (1999) 70–79. (in Russian)

[8] Kussul, N., et al. : Multi-Agent Security System based on Neural Network Model of User's Behavior. Int. J. on Information Theories and Applications Vol. 10 Num. 2 (2003) 184–188.

[9] Gorodetski, V., et al.: Agent-based model of Computer Network Security System: A Case Study. In: Proc. of the International Workshop 'Mathematical Methods, Models and Architectures for Computer Network Security', Lecture Notes in Computer Science, Vol. 2052. Springer Verlag (2001) 39-50.

[10] Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Upper Saddle River NJ: Prentice Hall (1995).

[11] Luck, M., McBurney, P., Preist, C.: Agent Technology: Enabling Next Generation Computing. AgentLink (2003).

[12] Haykin S.: Neural Networks: a comprehensive foundation. Upper Saddle River, New Jersey: Prentice Hall (1999).

[13] Wooldridge, M.: An Introduction to Multi-agent Systems. Chichester, England: John Wiley & Sons (2002).

Author's Information

Andrii Yu. Shelestov – PhD, Senior Researcher, Department of Space Information Technologies and Systems, Space Research Institute of NASU-NSAU, Glushkov Ave 40, Kyiv-187, 03650 Ukraine, e-mail: inform@ikd.kiev.ua.

USING THE AGGLOMERATIVE METHOD OF HIERARCHICAL CLUSTERING AS A DATA MINING TOOL IN CAPITAL MARKET[1]

Vera Marinova–Boncheva

Abstract: The purpose of this paper is to explain the notion of clustering and a concrete clustering method-agglomerative hierarchical clustering algorithm. It shows how a data mining method like clustering can be applied to the analysis of stocks, traded on the Bulgarian Stock Exchange in order to identify similar temporal behavior of the traded stocks. This problem is solved with the aid of a data mining tool that is called XLMiner™ for Microsoft Excel Office.

Keywords: Data Mining, Knowledge Discovery, Agglomerative Hierarchical Clustering.

ACM Classification Keywords: I.5.3 Clustering

Introduction

Data mining, the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems. Data mining tools can answer business questions that traditionally are time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations. Data mining consists of analysis of sets of supervised data with the aim of finding unexpected dependencies or to be generalized in a new way that is understandable and useful for owners of the data. There is a great deal of data mining techniques but we differentiate two of them like classification and clustering as supervised and unsupervised learning from data. [2]

[1] This work was supported by the Ministry of Education and Science under Project № MU-MI-1601/2006