Scientific Journal of Riga Technical University
Computer Science. Information Technology and Management Science
2011
Volume 49

Building an Intrusion Detection System for IT Security Based on Data Mining Techniques

Pjotrs Dorogovs[1], Arkady Borisov[2], Andrejs Romanovs[3], [1-3]*Riga Technical University*

***Abstract* – This paper aims to research various data mining techniques applied to solve intrusion detection problems. In general, intrusion detection techniques can be divided into two major categories: misuse detection and anomaly detection. Taking into consideration effectiveness of the anomaly detection technique not only against known types of attacks (like misuse detection does by exploiting signature database) but also against new ones, it has become a topical issue in majority of data and computer security researches.**

The techniques discussed in the paper include the Hidden Markov Model (HMM) method for modelling and evaluating invisible events based on system calls, further development of Stephanie Forrest's idea of the fixed-length audit trail patterns, the principle component analysis based method for anomaly intrusion detection with less computation efforts, algorithm based on k-nearest neighbour method, as well as applying association rule algorithm to audit data.

***Keywords* – Information security, intrusion detection, data mining, association rules**

I. INTRODUCTION

In recent years a vast majority of research activities in the area of anomaly detection have been focused on studying the behaviour of programs and the creation of their profiles based on system call log files. Until now, a simple anomaly detection method based on monitoring system calls initiated by the active and privileged processes is widely used [1]. The profile of normal behaviour is constructed by enumerating all unique, and related fixed length system calls, which are observed in the training data; in turn, previously undetermined sequences are considered abnormal. This approach has been extended by various other methods. It was suggested to utilize data mining approach to study samples of the system calls and construct small set of rules contained in normal data. During the monitoring and detection, the sequences that violate these rules are treated as anomalies [2]. Also Hidden Markov Model (HMM) can be used – a method for modelling and evaluation of invisible events based on system calls [3]. Later, the idea of analyzing patterns of system calls of fixed length has been further developed, by analyzing patterns of system calls, but of variable length [4]. Furthermore, a new method for intrusion detection, based on the method of principal components has been introduced [5]. In addition to those listed above, an algorithm based on the method of K-nearest neighbours is proposed in the paper [6].

II. PROBLEM DESCRIPTION

Profiling the behaviour of the end user is not less important aspect of data protection than the profiling the software activities. This method is effective in detecting internal attacks that constitute one-third of the corporate system security [7]. In information systems based on UNIX or Linux operating system, the sequences of shell commands are easily collectible and analyzable information, thus being the source material for creating profiles of end users. Besides, the collection of such information does not use significant system resources. On the other hand, taking into account the difference between the behaviour of end users, the building of profiles of their activities is a difficult task comparing to building a profile of program behaviour. Hackers can even try adapting their behaviour to fool IDS systems.

III. HIDDEN MARKOV MODELS

A particularly powerful method that uses a fixed number of states is a Hidden Markov model, which is widely used in speech recognition, as well as in the simulation of DNA sequences [8], [9]. The hidden Markov Model (HMM) describes a double stochastic process. HMM states represent some unobservable conditions of the system being modelled. In each state there is a certain probability of creating one of the possible system outputs and a separate probability indicating a possible next state.

Standard HMM has a fixed number of states, so before training the size of the data model has to be chosen. As described in the earlier research works, the only correct way forward is to select a number of states approximately corresponding to the number of unique system calls used by the program [3]. Most of the currently used programs use 40 different system calls; as a consequence, in most cases Hidden Markov Models of the 40 states can be used. The states are completely interconnected, transitions are allowed from any state to any other state. Consequently, probabilities of transition from one state to another and probabilities of occurrence of each system call should be kept. For a program using the S system calls and, therefore, the S state model, that means approximately $2S^2$ values.

In most cases, the transition probabilities can be randomly selected and then trained using the Baum-Welch algorithm, as described in [8]. And yet, in some cases, preliminary information is useful when performing initialization. As an example, system calls for print resources can be mentioned. The main difference between various calls for the same resource is the length of the document sent to the printer.

As it has already been mentioned in many researches – HMM training is a very resource-costly process. Calculations for each of the program trace, while performing training, take $O(TS)^2$, where T is the length of the system call (see Table

I**Error! Reference source not found.**), and S is the number of states. In addition, data storage requirements are high enough. "Trellis" of intermediate values that must be stored during calculations for each track requires rapid processing of $T(2S+1)$ of floating point values.

TABLE I

STANDARD DATA SET USED FOR MODELLING SYSTEM CALL BEHAVIOUR

Program	Intrusions	Normal data available		Normal data used for training		Normal data used for testing	
	Number of traces	Number of traces	Number of system calls	Number of traces	Number of system calls	Number of traces	Number of system calls
MITlpr	1001	2703	2926304	415	568733	1645	1553768
UNMlpr	1001	4298	2027468	390	329154	2823	1325670
Named	2	27	9230572	8	677340	12	7690572
Xlock	2	72	16937816	72	778661	1	16000000
Login	9	12	8894	12	8894	-	-
ps	26	24	6144	24	6144	-	-
inetd	31	3	541	3	541	-	-
stide	105	13726	15618237	150	246750	13526	15185927
sendmail	-	71760	44500219	4190	2309419	57775	35578249

Testing is more effective in comparison with the above described training. A standard way to check the HMM is to calculate the probability that it will produce data which differ from the original training set. However, simpler methods can also be used, which (unlike the standard method) are not sensitive to the length of the traces of system calls and better suited for use in online mode. "Reading" traces of a system calls one by one requires tracking of the state transition and output that are required from the HMM to produce that system calls. If the constructed HMM is a good model of the program, then the normal traces should require only standard transitions and outputs, while intrusive traces must include one or more system calls that require unusual state transitions and/or symbol outputs.

At any given time T, there is a list of currently possible states. Choosing just one more probable state for each system call is not the best method of using the constructed HMM; respectively, tracking of all possible ways should be done. Thresholds are set for the probability of "normal" state transitions. Then, if a system call is encountered in the trace which could only have been produced using below-threshold transitions or outputs, it is flagged as a mismatch.

The time to check each system call depends on the size of the model and size of the current list of valid states. This list tends to be very small, if it consists of only "normal" tracks and, on the other hand, contains all the possible states after the discovery of anomalies.

IV. INTRUSION DETECTION USING AUDIT TRAIL PATTERNS

Below a method used to extract and compare patterns, as well as the metrics used to distinguish between normal and abnormal behaviour [4] is described.

A. Extracting Patterns

Algorithm to build a table of fixed-length patterns is very simple. From the sequence of data passed through the analysis module, all the unique subsequences are retrieved (models of a given length *k*). This is achieved by using a sliding window of length *k* for all the input data, followed by writing all occurring subsequences. In this case, the duplicates are ignored. Construction of the pattern table is best illustrated by an example. When $k=3$ and the training set is $ABCCABC$, this table layout is achieved:

$$\{ABC, BCC, CCA, CAB\} \quad (1)$$

It should be noted that ABC pattern appears only once in the table, although it occurs in two positions, namely the first and last.

B. Pattern Matching

Pattern-matching technique is similar to the pattern-generation technique. The *k*-length window is moved through the sequence, recorded during the actual operation of the system. Each item is checked for a *match*, i.e., whether there is a pattern that corresponds to the subsequence in the window.

If the above presented data (pattern table) and simple sequence $ABCCACC$ are taken, for example, three matches – $\{ABC, BCC, CCA\}$, and two mismatches – $\{CAC, ACC\}$ can be observed.

C. Metrics

It should be noted that the length of the analyzed sequences cannot be a source of information about a possible invasion. All events must be analyzed upon arrival to the analysis-unit, besides there is no need to wait until all the events of one

particular process are available for analysis before starting to check for signs of intrusion. That would be problematic, for example, in cases of continuously running processes. In [11], three measures are used to distinguish between normal and abnormal behaviour. However, only the measure described below is independent of the sequence length. Let a and b be two sequences of length k. The expression a_i designates the character at position i. The difference $d(a,b)$ between a and b is defined as

$$d(a,b) = \sum_{i=1}^{k} f_i(a,b), \quad (2)$$

$$\text{where } f_i(a,b) = \begin{cases} 0 \; if \; a_i = b_i \\ 1 \; \textbf{otherwise} \end{cases} \quad (3)$$

During pattern matching, for each subsequence u the minimum distance $d_{\min(u)}$ between u and the entries in the pattern table is determined:

$$d_{\min(u)} = \min\{d(u,p) \forall \; patterns \; p\} \quad (4)$$

To detect an attack, at least one of the subsequences generated by the very attack, should be classified as an anomaly. In terms of the above measure, there is at least one subsequence u for which

$$d_{\min(u)} > 0 \quad (5)$$

It is assumed that the higher the value $d_{\min}$, the greater the likelihood that the sequence is actually generated by the invasion. In practice, the maximum observed value $d_{\min}$ is used as an example of the invasion, because it represents a strong signal of the anomalous behaviour of the system. Anomaly signal, S_A, is defined as:

$$S_A = \max\{d_{\min(u)} \forall \; subsequences \; u\} \quad (6)$$

Ideally, the value S_A, which is higher than 0 can be considered a sign of invasion. However, as shown by experimental results, full compliance cannot always be achieved [11]. Thus, the threshold is defined so that only the sequences with S_A above this threshold are considered suspicious.

V. K-Nearest Neighbours

Instead of analyzing the local ordering of system calls, data on the frequency of system calls can be used to characterize program behaviour. Using the metaphor of text processing, each system call is treated as a "word" in a long document, and a set of system calls generated by each process is seen as a "document". This analogy enables us to use the full range of well-developed methods of text processing [10] for the problem of intrusion detection. One such method is the method of K-nearest neighbour classification [6].

By analogy with text categorization, each process initially appears as a vector where each entry is a system call during the process. Ranking techniques, such as the frequency weighting and $tf \bullet idf$, are used to determine the values of vector elements. To classify a new process as normal or intrusive, kNN classifier computes the similarity between the new process and each instance of training data, and uses the class labels of *K* closest neighbours to define the class of the new process. This approach contributes to the underlying assumption that the processes belonging to one class will be collected in a single cluster in the vector space.

Some significant advantages of using text classification techniques to detect intrusions should be identified. First, the size of system calls dictionary is very limited. In a widely used DARPA data set, less than 100 different system calls are presented, while a typical text classification problem can handle over 15,000 unique words [10]. Thus, the dimension of the process vector is greatly reduced, and there is no need to use dimensionality reduction methods. Secondly, the problem of intrusion detection can be considered a binary classification problem, which makes adapting text categorization methods very simple.

First for training purposes intrusion detection is implemented, based solely on the normal behaviour of the program. In order to ensure processing of all possible normal processes of the program, a large amount of training data must be used.

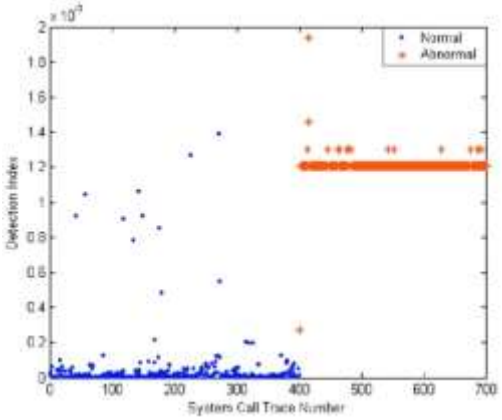

Fig. 1. Experimental results for system call data MITlpr

After processing the training data of normal behaviour, the method of text categorization can easily be adapted to detect anomalies. It is necessary to scan the audit test data to extract all system call sequences for each new process. Afterwards, it is necessary to calculate the similarity of the new process and every process from a training set of data. If at least in one case the value of similarity equals 1, that means that the new process system call and a system call of a process from training data sets match perfectly, then this new process can be immediately classified as "normal". Otherwise, ratings of calculated similarities are sorted, and k-nearest neighbours are selected to determine the "normality" of the new process.

VI. Principle Component Analysis

Assuming that the observed dataset is divided into m blocks either by a fixed length (for example, divided consecutively by 100 in the command data) or by an appointed scheme (for example, separated by processes in the system call data) and there are n unique elements in the available dataset, in this case, the analyzed dataset can be represented by m vectors, each of which contains n observations. Then a $n \times m$ matrix X can be build, where each element X_{ij} stands for the frequency of i -th, individual element occurs in the j -th block.

Using the available training set of data vectors $x_1, x_2, \dots, x_m$, average vector μ and each mean-adjusted vector can be computed. So as m eigenvalue-eigenvector pairs $(\lambda_1, u_1), (\lambda_2, u_2), \dots, (\lambda_m, u_m)$ of the sample covariance matrix of vectors [11], [12] can be calculated. Several eigenvectors $u_1, u_2, \dots, u_k (k \ll m)$, forming the $n \times k$ matrix U, which can be used to represent the distribution of the initial data, are chosen empirically.

Given a test data vector t, it can be projected onto the k -dimensional subspace according to the rules [13]

$$y = U^T(t - \mu) \tag{7}$$

The distance between the test data vector and its projection onto the subspace is simply the distance between the mean-adjusted input data vector

$$\Phi = t - \mu \tag{8}$$

and

$$\Phi_f = Uy \tag{9}$$

If the test data vector is normal (does not contain any signs of intrusion), the vector and its projection will be very similar, and the distance between them is very small and close to zero [14]. Based on this property, a normal program and user behaviour are profiled for anomaly detection. In principle, the following three methods for measuring the distance (or in this case, similarity) between two vectors can be used for comparison of the experimental results - Euclidean distance, Cosine distance and Signal-to-Noise (SNR) measure.

Euclidean distance, Cosine distance and the SNR measure for the anomaly detection purposes can be represented as follows:

$$\varepsilon_1 = \|\Phi - \Phi_f\|^2 \tag{10}$$

$$\varepsilon_2 = \frac{\Phi^T \Phi_f}{\|\Phi\| \|\Phi_f\|} \tag{11}$$

$$\varepsilon_3 = 10 \log\left(\frac{\|\Phi\|^2}{\|\Phi - \Phi_f\|^2}\right) \tag{12}$$

In the procedure of anomaly detection, ε_1, ε_2 and ε_3 are considered to be *detection indices*. If either ε_1, ε_2 is below or ε_3 is above a predetermined threshold, test data t will be classified as normal, otherwise as anomalous.

Using PCA for intrusion detection, good testing results can be achieved. Figure 1 shows the experimental results using squared Euclidean distance measure with the former 200 traces of data for training and other 700 traces for testing. It is observed that abnormal data can be easily distinguished from normal data.

VII. Association Rules

All the above-described data mining methods are widely studied and discussed in many articles and scientific experiments. In turn, a method for association rule construction being one of the most popular approaches among various data mining methods is relatively rarely used in anomaly detection sphere.

Initially, association rule induction method has been proposed to be used for the so-called problem of market basket analysis, which aims to find patterns in behaviour of supermarket consumers. In particular, Boolean associative rules aim to identify sets of products (items) which are often bought together. Discovered rules may, in particular, tell us that people who buy butter and milk will also buy bread.

Construction of association rules is aimed at finding meaningful connections between the elements of a given set of data. Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of products, called elements. Let D be the set of transactions, where each transaction T – is a set of elements of I, where $T \subseteq I$. Each transaction represents a binary vector, where $t[k] = 1$ if i_k element is present in the transaction, otherwise $t[k] = 0$. We say that transaction T contains X, a set of elements of I, if $X \subset T$. Implication $X \Rightarrow Y$ is called an associative rule, where $X \subset I$, $Y \subset I$, $X \cap Y = \emptyset$. Rule $X \Rightarrow Y$ has support S, if $S\%$ of transactions in D, contain $X \cup Y$, $supp(X \Rightarrow Y) = supp(X \cup Y)$. The reliability of the rules indicates the probability that X follows from Y. Rule $X \Rightarrow Y$ is valid with confidence C, if $C\%$ of transactions in D, containing X, also contains Y,

$$conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}. \tag{13}$$

In other words, the purpose of analysis is to identify the following dependencies: if a certain set of elements of X appears in the transaction, then on the basis of this we can conclude that a different set of Y is also to appear in this transaction. Establishing such relationships enables us to find very simple and intuitive rules.

Association rule mining algorithms are designed to find all the rules of *XY*, furthermore support and confidence of these rules should be above some certain threshold, called, respectively, the minimum support (*minsupport*) and the minimum confidence (*minconfidence*).

Association rule mining problem is divided into two subtasks:

- Mining of all itemsets that satisfy the *minsupport* threshold. Such itemsets are called frequent.
- Generation of rules from the itemsets mined according to the previous point with reliability, meeting the threshold *minconfidence*.

One of the first introduced algorithms to effectively solve this class of problems is the APriori algorithm. The algorithm works according to the above mentioned two stages – the first step is to find frequent itemsets, and the second one – to mine rules. The number of items in a set is called the size of the set, and a set consisting of *k* elements – *k*-cell collection.

In the first step of the algorithm, one-element frequent sets are computed. In order to do this, it is necessary to compute a support value for all elements in the data base.

The following steps will consist of two parts: the generation of potentially frequent itemsets (called candidates) and computing of support for candidates.

The above-described algorithm can be represented in the following pseudo-code:

```
F1 = {frequent one-cell element set }
for (k=2; Fk-1 <> ∅; k++)
{
 Ck = Apriorigen(Fk-1) // generation of the
 candidates
 For all transaction t ∈T
   {
   Ct = subset(Ck, t) // removal of redundant rules
   For all candidates c ∈ Ct
   c.count ++
   }
 Fk = { c ∈ Ck  | c.count >= minsupport} //
candidate
 selection
}
Result ∪ Fk
```

The described association rule induction algorithm can be used for two phase (learning and detection phases) intrusion detection system. Taking into account that during the learning phase the system learns and analyzes raw data provided and then, on the basis of these rules, creates a security model that will subsequently be used for a real intrusion detection, it can be concluded that the construction of proper rules of normal behaviour of the system is a key factor for an intrusion detection system.

VIII. CONCLUSIONS

Such data mining techniques applied to the problem of intrusion detection have been examined, as Hidden Markov models, audit trail pattern extraction, k-nearest neighbour method, principle component analysis and the construction of association rules. All the above mentioned techniques have both advantages and disadvantages regarding the stated task of intrusion detection.

Markov models have a fixed number of states, so, firstly, it is necessary to decide on the size of the data model before learning process starts, and, secondly, taking into account the fact that the states are completely interconnected, transitions are allowed from any state to any other state and, consequently, probabilities of transition from one state to another and probabilities of occurrence of each system call should be kept. For a program using the S system calls and, therefore, the S state model, that means approximately $2S^2$ values.

In case of an audit trail pattern extraction method, it must be emphasized that the anomaly signal, S_A is defined as:

$$S_A = \max\{d_{\min(u)} \forall\ subsequences\ u\} \qquad (14)$$

Ideally, the value S_A, which is higher than 0 can be considered a sign of invasion. However, as shown by experimental results, full compliance cannot always be achieved [12]. Thus, the threshold is defined so that only the sequences with S_A above this threshold are considered suspicious.

The k-nearest neighbour data mining technique is one of the most applicable to the solution of the intrusion detection problem, taking into account that this method allows using the full range of well-developed methods of text processing [10]. As mentioned above, the method for association rule induction being one of the most popular approaches among various data mining methods is relatively rarely used in anomaly detection sphere. It is absolutely necessary to mention such important advantages of this algorithm to solve the problem as the simplicity of algorithm usage in transactional databases, as well as the very essence of the construction of association rules – support and confidence of these rules must be above some certain threshold, called, respectively, the minimum support (*minsupport*) and minimum confidence (*minconfidence*). Thus, this method shows a good (in this case – low) level of false detection of anomalies.

REFERENCES

[1] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. *A sense of self for Unix processes*. Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy, Los Alamos, CA, pp.120-128, 1996.

[2] W. Lee and S. Stolfo. Data Mining Approaches for Intrusion Detection. In Proceedings of the 7th USENIX Security Symposium, Usenix Association, pp.79-94, January 1998.

[3] C. Warrender, S. Forrest and B. Pearlmutter, Detecting Intrusions Using System Calls: Alternative Data Models, Proceedings of 1999 IEEE Symposium on Security and Privacy, pp.133-145, 1999.

[4] A. Wespi, M. Dacier and H. Debar. Intrusion Detection Using Variable-Length Audit Trail Patterns. In Debar, H., Me, etc editors, Proceedings of the Third International Workshop on the Recent Advances in Intrusion Detection (RAID'2000), No. 1907 in LNCS.

[5] W. Wang, X. Guan and Xiangliang Zhang. A Novel Intrusion Detection Method Based on Principle Component Analysis in Computer Security. In Proceedings of the International IEEE Symposium on Neural Networks, Dalian, China. Lecture Notes in Computer Science, Vol. 3174, pp. 657-662, Aug 2004.

[6] Y.H. Liao and V. R. Vemuri, Use of K-nearest Neighbor Classifier for Intrusion Detection. Computer & Security, vol. 21, No 5, pp. 439-448, 2002.

[7] Computer Security Institute, CSI/FBI computer crime and security survey results quantify financial losses, Computer Security Alert 181 (1998).

[8] L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. Proceedings of the IEEE, 77(2):257–286, 1989.

[9] L. R. Rabiner and B. H. Juang. An introduction to Hidden Markov Models. IEEE ASSP Magazine, pages 4–16, January 1986.

[10] K. Aas and L. Eikvil, Text Categorisation: A Survey, http://citeseer.nj.nec.com/aas99text.html, 1999

[11] Steven A. Hofmeyr, Stephanie Forrest, and Anil Somayaji. Intrusion detection using sequences of system calls. Journal of Computer Security, 6(3):151–180, 1998.
[12] Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. 2nd Edn. China Machine Press, Beijing (2004) 568-570
[13] Jolliffe, I.T.: Principal Component Analysis. 2nd Edn. Springer-Verlag, New York (2002)
[14] Turk, M., Pentland, A.: Eigenfaces for Recognition. Journal of Cognitive Neuroscience. Vol. 3, No. 1 (1991) 71-86

Pjotrs Dorogovs is a Doctoral student at the Department of Modelling and Simulation, Riga Technical University (Latvia). He received Bachelor degree in Information Technology from Riga Technical University in 2005. He obtained Master degree in IT project management (Mg.sc.ing.) from Riga Technical University in 2008. His research interests are IT security and IT governance. Since 2006 he is the Head of National Schengen Information System Unit of the Information Centre of the Ministry of the Interior of the Republic of Latvia. Since 2006 he has been participating in monthly Large-scale IT system management forum taking place mostly in Brussels organized by the European Parliament. He is a member of IEEE. He participated in several international scientific conferences and research projects with scientific publications in the field of ICT. E-mail: pjotrs.dorogovs@inbox.lv

Arkady Borisov is a Professor of Computer Science at the Faculty of Computer Science and Information Technology, Riga Technical University (Latvia). He received his Doctoral degree in Technical Cybernetics from Riga Polytechnic Institute in 1970 and Dr.hab.sci.comp. degree in Technical Cybernetics from Taganrog State Radio-Engineering University in 1986.

His research areas include artificial intelligence, decision support systems, fuzzy set theory and its applications and artificial neural systems. He has more than 210 publications in the area.
E-mail: arkadijs.borisovs@cs.rtu.lv

Andrejs Romanovs is an Associate Professor at the Department of Modelling and Simulation, Riga Technical University (Latvia). In 1993 he graduated from the University of Latvia as an Engineer-Economist in Electronic Data Processing (ing.oec.). He received a Master degree in Management Information Systems (MBA) in 1995, Doctoral degree in Telematics and Logistics (Dr.sc.ing.) in 2006. His professional interests include modelling of management information systems, IT governance, logistics information technologies and electronic commerce, as well as education in these areas.
Andrejs Romanovs has more than 20 years of practical experience in development of more than 50 data processing and management information systems in Latvia and abroad for state institutions and private business as an IT project manager and system analyst. He is a member of IEEE and the Latvian Simulation Society. He is an author of 30 scientific publications and two textbooks in the field of information technology. He participated in more than 20 international scientific conferences, as well as in seven research projects.
E-mail: andrejs.romanovs@rtu.lv

Pjotrs Dorogovs, Arkādijs Borisovs, Andrejs Romanovs. Nesankcionētas piekļuves sistēmas izveidošana, izmantojot intelektuālās datu ieguves metodes
Darba mērķis ir izpētīt dažādas datu ieguves metodes, kas tiek izmantotas nesankcionētas piekļuves problēmas risināšanai. Kopumā nesankcionētas piekļuves paņēmienus var iedalīt divās nozīmīgās kategorijās – ļaunprātīga izmantošanas un anomāliju noteikšana. Ņemot vērā anomāliju noteikšanas metodes efektivitāti ne tikai pret zināma veida uzbrukumiem (piemēram, izmantojot parakstu datu bāzi), bet arī pret jauniem veidiem, tā ir kļuvusi par aktuālāko tēmu lielākajā daļā datu un informācijas tehnoloģiju drošības pētījumos.
Darbā tiek izpētītas tādas datu ieguves metodes kā Markova apslēptā modeļa (Hidden Markov Model (HMM)) metode apslēpto notikumu, kas balstās uz sistēmas izsaukumiem, modelēšanai un novērtēšanai , tālāka Stefanijas Foresteres idejas izstrāde par fiksētā garuma Auditācijas pierakstu analīzi, metode anomāliju noteikšanai, kas balstās uz galveno komponentu analīzi, K-tuvāko kaimiņu metode, kā arī asociatīvo noteikumu metodes piemērošana auditācijas datiem.
Visas izpētītās datu ieguves metodes ir pietiekami labi piemērotas nesankcionētas piekļuves problēmas risināšanai, tomēr to reālā pielietošanas sfēra lielā mērā ir atkarīga no konkrētiem apstākļiem. Piemēram, paslēpto Markova modeļu izmantošanai ir jārēķinās ar augstām skaitļošanas prasībām modeļa izveidošanai, kas dažos gadījumos var būt nepieņemami. Savukārt, ņemot vērā K-tuvāko kaimiņu metodes piemērotību teksta atpazīšanai, tā arī ir salīdzinoši viegli pielietojama minētās problēmas risināšanai. Turpmākie pētījumi apskatītajā nesankcionētas piekļuves sfērā tiek novirzīti Asociatīvo noteikumu indukcijas metodes pielietošanai, ņemot vērā to salīdzinoši vieglo pielietojumu transakciju datu bāzēs, turklāt šī metode demonstrē zemu kļūdaino atklājumu līmeni.

Пётр Дорогов, Аркадий Борисов, Андрей Романов. Построение системы обнаружения несанкционированных вторжений с использованием методов интеллектуального анализа данных
Данная работа посвящена исследованию различных методов интеллектуального анализа данных, используемых для решения проблем обнаружения несанкционированных вторжений. В общем случае обнаружения вторжений можно разделить на две основные категории – выявление злоупотреблений и обнаружение аномалий. Принимая во внимание эффективность не только для определения известных видов атак (например, обнаружение злоупотреблений реализуется, используя сигнатурную базу данных), но также и против новых, обнаружение аномалий стало актуальным вопросом в большинстве исследований по безопасности данных и компьютерной безопасности.
Методы, обсуждаемые в работе, включают в себя: моделирование невидимых событий на основе системных вызовов, используя скрытые Марковские модели; дальнейшее развитие идеи Стефании Форрест по анализу аудитных данных фиксированной длины; обнаружение аномалий, используя метод главных компонентов; алгоритм, основанный на анализе K-ближайших соседей, а также построение ассоциативных правил для аудитных данных.
Все изученные методы анализа данных достаточно хорошо применимы для решения проблемы несанкционированного доступа, но реальная сфера их применения существенно зависит от конкретных обстоятельств. Например, применяя скрытые Марковские модели, необходимо учитывать высокие вычислительные требования при создании модели, что в некоторых случаях может оказаться неприемлемым. С другой стороны, учитывая высокую пригодность метода K-ближайших соседей для распознавания текста, он оказывается относительно простым в использовании для решения поставленной проблемы. Планируется, что дальнейшие исследования в рамках решения проблемы несанкционированного доступа будут посвящены методу индукции ассоциативных правил, принимая во внимание относительную легкость его применения в транзакционных базах данных; кроме того, этот метод демонстрирует низкий уровень ложных детекций.