Anomaly Intrusion Detection Systems: Handling Temporal Relations between Events

Alexandr Seleznyov, Seppo Puuronen
{alexandr,sepi}@jytko.jyu.fi

Department of Computer Science and Information Systems
University of Jyväskylä
P.O.Box 35, FIN-40351 Jyväskylä
Finland

Abstract

Lately, many approaches have been developed to discover computer abuse. Some of them use data mining techniques to discover anomalous behavior in audit trail, considering this behavior as an intrusive one. This paper discusses a temporal knowledge representation of users' behavior that is used by data mining tools to construct behavior patterns. These are used to decide whether current behavior follows a certain normal pattern or differs from all known users' behavior patterns. The representation uses Allen's temporal interval algebra to describe the temporal relations between events caused by the user. Also we discuss how our representation is used to help in the concept drift when the set of training samples is reduced by removing old data which is no more used for classification.

1. Introduction

Computer systems have become an essential part of critical systems having crucial importance and hence they must have as high tolerance as possible against misuse activities. These activities may be successful due to software bugs, hardware or software failures, or incorrect system administration. Also the amount of successful intrusion incidents has grown lately quite high: even 99% of all major companies have reported at least one major intrusion incident [Spafford 96]. Thus, maintenance of a proper level of security is a very important problem and unlikely to be solved in the near future.

There exist two major categories of intrusion detection systems: *misuse* intrusion detection and *abnormality* intrusion detection [Smaha 93]. The systems of the first category are based on the detection of intrusions

that follow well defined patterns of attack exploiting known system's and application's software vulnerabilities [Kumar 95]. A misuse intrusion detection system has knowledge about poor or unacceptable behavior, which it directly searches for [Smaha 92]. It is difficult for such a system to learn [Kumar 95], hence these kinds of systems are unable to recognize attacks that are not precisely encoded in the system. Misuse detection is harder to automate since it requires applying many rules (as in NIDES [Lunt at al. 89]) or searching for many patterns (as in [Shieh and Gligor 91] and [Kumar and Spafford 94]). Moreover, it is almost impossible to perform a proper testing of such systems due to an insufficient amount of information about the real intrusion cases.

The intrusion detection systems of the second category are based on the detection of the anomalous behavior or the abnormal use of the computer resources [Kumar and Spafford 94]. For example, if a user uses computer only during working hours and only from his office, a connection established using his *userid* during a night from a remote host is absolutely anomalous and, moreover, may be an intrusion. The basic principles of detecting intrusions by identifying "abnormal" behavior are outlined by Anderson [Anderson 80]. Recently, this approach has been extensively developed and many implementations have been suggested [Heberlein at al. 91, Lunt at al. 89, Liepins and Vaccaro 89, Porras and Kemmerer 92]. In these works the classification is based on the sequence of events in time and time relations between events are not taken into account at all or only very little attention is given to it. Sometimes time relations play a crucial role in attempting to classify events, i.e. determining whether an event is a part of anomalous or normal behavior. For instance, if an account has been compromised by an IP spoof attack [Phrack 96], it is easier to recognize it relying on the time relations between events, since the misuse activity appears as a continuation of the normal behavior within a single session. Another aspect that has not received enough attention is the natural change of users' normal behavior when he takes new applications in his use or when his topic of interest changes. Therefore, some of the old data does not accurately reflect the user's behavior any more and should be removed from the training set to handle this *concept drift* [Schlimmer 87].

According to one definition given by Garfinkel and Spafford [Garfinkel and Spafford 91], a secure computer system *can be depended upon to behave as it is expected to.* By analogy with this definition our approach is based on the assumption that the user's behavior includes regularities which can be detected and coded as a number of patterns. The

information derived from these patterns could be used to detect the abnormal behavior and to learn the system.
As can be seen, a system based on the proposed approach forms a last line of defense and is able to recognize intrusions and abuse even if the system has already been compromised. For example, if someone's password is compromised the system is able to elucidate this by observing deviation between the current stream of events and the stored patterns.
In this article we attempt to develop a representation of knowledge about the user's behavior which is later used with data mining tools to analyze the behavior patterns and make decisions with them. This representation uses Allen's temporal intervals' algebra [Allen 83] to describe the knowledge of temporal relations between events. We define the notion of *layer* and *event representation* on each layer in Section 2, *distance consistency* between events in Section 3, and *coefficient of reliability* in Section 4. We discuss how to use our representation in order to help in handling the concept drift and in reducing the set of training samples by removing old data which are not used for classification any more. Finally, at Section 4 we outline future research directions.

2. Basic concepts

Traditionally, in anomaly detection, user profiles have been built basing on different characteristics, such as consumed resources, command count, typing rate, command sequences, etc. In these cases information analysis has been made using system log files, command traces, and audit trails [Lane and Brodley 98]. In our approach we store and analyze information basing on its context. We present information as temporal intervals and apply Allen's algebra [Allen83] to discover relations between temporal intervals and to store them for further classification.
The term *event*[1] is widely used within the temporal database area giving it different meanings. For an operation system's point of view it is a single line in the audit trail; for example it can be a *request for connection establishment* between the server and the user's workstation. On the other hand, from the user point of view it is a much more complicated procedure - we call it *action*. For example - *checking mail*,

[1] In this paper we attempt to use all definitions and terms in accordance with [Jensen at al. 98] and [Allen and Ferguson 94].

text editing, etc. For an operation system each of these actions includes several events. C*hecking mail*, for instance, includes *establishing connection, user authentication, commands execution, data transfer*, and *closing connection*. We define a notion of *layer* which reflects the level of information abstraction (information about relations between temporal intervals) and *event* on each layer in order to clarify our usage of event concept. We use an underlying assumption that a person's interaction with a computer consists of different *activities* that he performs in order to achieve his goals. These activities consist of *actions*. Each action causes series of *events* in the operation system. Each user performs similar activities which are expressed by repeated sets of actions and which differ on a per-user basis. This gives the possibility to differentiate an intruder from a valid user.

Layer is a concept generalization level of relations between occurrences, each of which represents a single event on a different layer. At the lowest i*nstant layer* all occurrences are represented as a time point (instant) on an underlying time axis. A single occurrence on this layer is called an *event*. It is equivalent to a single line in the audit trail.

An event is described by a single instant relative to a particular user. We add to this description the name of the event with an integer index that defines the number of the event relative to the user and also a place from which the user caused this event. Also, the last field is reserved for the event's particular information, which differs according to the type of event: it may be the name of the user, the name of the computer, an error code, etc. In other words, it is auxiliary information. Thus the whole description of the event is as follows:

Event: <<Name>, <Index>, <Place>, <Absolute time>, <other information>>.

One example can be a mail check using POP3 protocol:

<<Popper>, <2364>, <computer_name>, < Jan 15 16:54:54>, <user_name, 0 0 88 90232>>.

In the *interval or action layer* level the events (i.e. actions) with their relations are described. The action is considered as a temporal interval, as for example: LOGIN-LOGOUT.

A *relation* defines a temporal relation between two events as one of Allen's point temporal relations [Allen 83] and has a *Name*. Considering the following model of *time*: time is linear, and time points can be identified with the rationales under the usual ordering <. The difference of any two-time points is likewise a rational number [Kautz and Ladkin

L91]. There are three basic relations that are used to represent relations of events: < - "less" relation, = - "equal" relation, and > - "greater" relation.

Thus Action: $<< Event_i >, < Event_j >, <Name>>$, where $Name \in \{?,?,?\}$.

A *principal relation* is a relation between two events *i* and *j* with the same name, i.e.: $LOGIN_i ? LOGIN_j$. A principal relation has to be *consistent.*

The most complicated level is the *activity layer,* which is represented by actions (temporal intervals or moments) and relations between them. Because the actions are extended in time, different actions may overlap in time and interact. One LOGIN-LOGOUT temporal interval, for instance, includes dozens of mail check intervals. A single occurrence on this level we call an *activity*.

Relation between two actions (temporal intervals) is defined as one of Allen's interval temporal relations [Allen 83] and it has a *Name*.

By analogy with events the *principal relation* between actions is the relation between two actions *i* and *j* with the same name. Each particular action has a different consistency condition, which is defined by the developer. For example for LOGIN-LOGOUT actions it is: if $Name \in \{before, after, meets, met\ by\}$. Indeed, in the real world the consistency conditions are more complicated.

One of the important functions of the intrusion detection system is the *consistency control* of all the principal relations and a warning if an inconsistency is discovered.

Activity Relation: $<< Action_i >, < Action_j >, <Name>>$, where:

$Name \in \{before, after, meets, met\ by, during, includes, overlaps, overlapped\ by, starts, started\ by, finishes, finished\ by, equals\}$.

According to [Allen 83]:

1. given any interval, there exists another interval related to it by each of the thirteen relationships;
2. the relationships are mutually exclusive;
3. the relations have a transitive behavior, e.g. if *A* is "before" *B*, and *B* "meets" *C* then *A* is "before" *C*.

Also, it is possible to specify interval relations using relations between endpoints of these intervals [Hirsch 96]. For example: "equals", "before", "after", "during", "includes", etc.

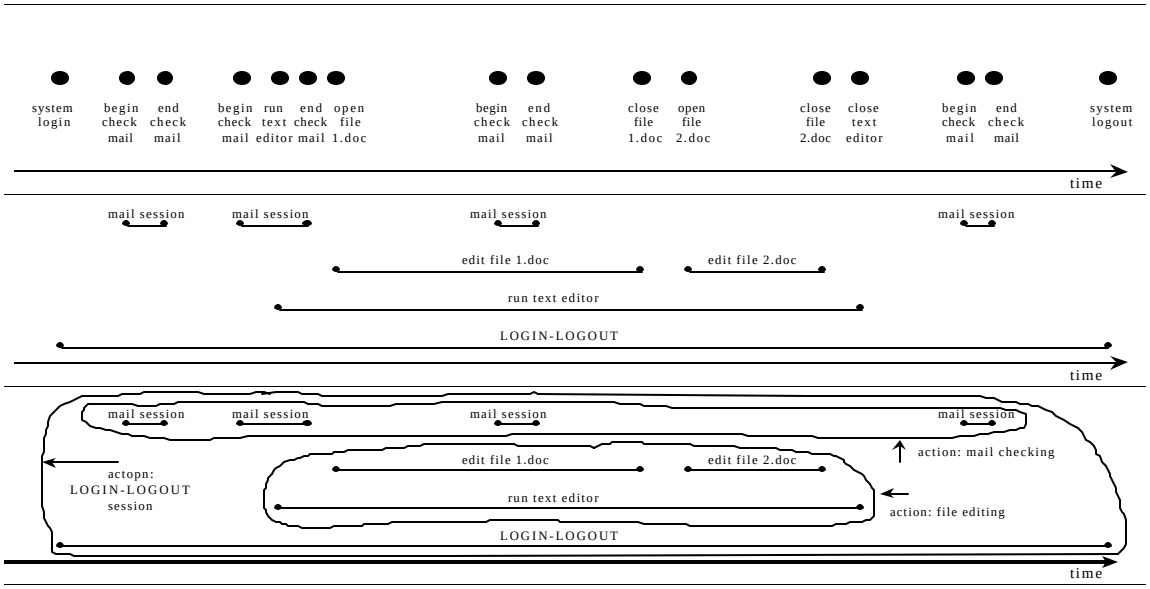

Figure 1. Layer structure of a simple user session.

The layer structure of a simple user session is shown in Figure 1. These three layers are the way by which agents classify certain patterns of change. No one is more correct than other, although some may be more informative for certain circumstances [Allen and Ferguson 94].

3. Consistency of relations

One group of attacks take advantage of the ability to forge (or "spoof") an IP address which is sent along with every IP packet even when it is not actually used for routing. This makes it possible for a user to pretend that he is his neighbor when sending packets to a server. While he will not see any transferred data of his neighbor, he is still able to take advantage of the situation. For example, he can pretend to be his neighbor on a command channel and ask the server to open a second channel to his own address. He still does not see any data of the first channel, but the second channel is wide open for him to be exploited.
The situation above creates an inconsistency in the log files, which can be checked without spending many resources. There exist at least two different kinds of consistency: 1) *information consistency* which requires that every action has to begin with some event and end with a corresponding one, and 2) *distance consistency* which requires that

when we consider the actions as temporal intervals, every pair of intervals has distance that has to satisfy predefined conditions.
Checking for the information consistency is a trivial task and so we concentrate on the distance consistency here. We base the distance consistency evaluation between two actions on the IP addresses (i.e. approximate spatial places) of the source packets of the actions.
We introduce two notions: *temporal distance between places* and *coefficient of mobility*.
Temporal distance between two places $t(place_i, place_j)$ is the minimal time needed by any user to change place from the first place to the second one. For simplicity we include to the profile description of each user a matrix which defines these minimal times between different *time zones*. For example, in the department case they may be: department, university, city, country, and world.

The diagonal elements of the matrix- $t(time\_zone_i, time\_zone_i)$ define the minimum time needed to change IP address within the same time zone. This time depends on: 1) the local physical network structure which sets constraints on every user of the network and 2) the user related characteristics of the IP address change, as for example how the user moves from one IP address to the other one.
The *coefficient of mobility* shows the probability that the next session (i.e. login) of a user will occur from a different IP address than the previous one. The profile of each user contains a matrix of coefficients $p(time\_zone_i, time\_zone_j)$, which are statistically created and constantly updated for every user. The diagonal elements of the coefficient matrix $p(time\_zone_i, time\_zone_i)$ define the probabilities of changing IP addresses within the same time zone and the non-diagonal elements $p(time\_zone_i, time\_zone_j)$ define the probabilities of changing IP addresses between two time zones. The coefficient mobility matrix is mostly defined by each particular user behavior. Consider, for instance, a simple university example:

?? A university researcher participates conferences around the world and he uses his account occasionally from different countries and continents.
?? A system engineer does not travel so much around the world, but he maintains the network and the computers of the department staff using his account from many different computers in the department.
?? A department secretary maintains a secure student's credit unit database using his computer mostly from one location - his office.

Let the *temporal distance* between two events be $t(event_i, event_j) ? event_i ? event_j$. The relation between two events i and j is *distance consistent* if:

$$? (event_i ? event_j), t(event_i, event_j) ? t(time\_zone_i, time\_zone_j)$$

Let the *temporal distance* between two actions be $t(action_i, action_j) ? action\_begin_j ? action\_end_i$. The relation between two *certain kind* of actions i and j is *distance consistent* if:

$$? (action_i ? action_j), t(action_i, action_j) ? t(time\_zone_i, time\_zone_j).$$

The above distance consistency definition of actions requires that the two actions are of certain kind. As can be seen in Figure 1 the actions generally are active during some temporal intervals, which may overlap. Thus the application of the distance consistency between actions requires considerations of actions and the local arrangements. Some of these situations are more obvious than the other ones. For example, if two users are logged in from different places during the same time interval under the same login name, then we can suspect that one of them is an intruder. (Of course, even this conclusion requires that the user never forgets to logout connection or that there exist some automatic logout system.) Some other situations are more complicated and hidden and they should be revealed during the learning process.

4. Coefficient of reliability and concept drift

One of the main objectives of abnormal behavior detection systems is not only to expose the abnormal behavior of users, but also sometimes to choose between two contradictory cases. For example if two audit records showing that there exist two sessions from different IP addresses at the same time under the same login name, the task of the system is not only to detect this, but also to determine which login is abnormal (may be they both are).

Let the *coefficient of reliability of an action* be a real number from the closed interval from 0 to 1. We assume that these coefficients are initialized for each user taking into account the validity of each action. The more doubt a system has about an action nature, the smaller value is assigned to the coefficient and vice versa. The value of a coefficient for a new action is calculated taking into account the previous value of the

coefficient, the user's mobility, and the temporal distances of time zones and actions. The exact formulas are still under research.
We suggest implementing the concept drift detection and update using a training set and the coefficients of reliability with data mining classifiers. We suppose that every action in the training set has a coefficient value. If a data mining classifier uses an action to classify a new case, then the coefficient of reliability of that action is increased and vice versa. When the coefficient of an action in a training set becomes smaller than a predefined threshold value then the action is replaced by another action that reflects the user's behavior better. Thus adaptation of the threshold value have direct influence to the speed of the training set updating.

5. Conclusions

The problem of intrusion detection is very important today, because networked computing unfortunately offers almost unlimited possibilities and opportunities for intrusions.
In this paper we have presented an information representation method for intrusion detection system which uses data mining techniques to detect anomalous behavior. The main assumption behind is that the behavior of users follows regularities that may be discovered and presented using the approach for the recognition of the user. The approach suggests that in cases where the audit trail information is inconsistent it is possible to expose it using temporal interval representation applying temporal algebra. Thus, some attacks (as "spoof" [Phrack 96], etc.) may be detected without significant use of resources and therefore detection may be performed in real-time. Also the primary disadvantage of anomaly detection - the gradual training may be overcome using temporal algebra to determine important relations between events by grouping together actions based on the relations of their endpoints and testing for a trend among them.
The presented approach is interesting from a theoretical point of view. It is still under ongoing research efforts and numerous tests and evaluations with real implementation, which are needed to verify the practical aspects of the approach.

Acknowledgements

We would like to thank Jyväskylä Graduate School in Computing and Mathematical Sciences (COMAS) for supporting this work.

References

[Allen 83] Allen J., *Maintaining knowledge about temporal intervals*, Communications of the ACM Vol. 26 (1983), pp. 832--843.

[Allen and Ferguson 94] Allen J., Ferguson G., *Actions and Events in Interval Temporal Logic*, Technical Report 521. University of Rochester, Computer Science Department. Rochester, New York, 1994.

[Anderson 80] Anderson J.P., *Computer Security Threat Monitoring and Surveillance*. Technical report, James P. Anderson Co., Fort Washington, Pennsylvania, 1980.

[Garfinkel and Spafford 91] Garfinkel S., Spafford G., *Practical Unix Security*, O'Reilly and Associates, Sebastopol, California, 1991.

[Hirsch 96] Hirsch R., *Relation algebra of intervals*, Artificial Intelligence 83 (1996), pp. 267--295.

[Heberlein at al. 91] Heberlein L.T., Levitt K.N. and Mukherjee B.*A Method To Detect Intrusive Activity in a Networked Environment*, In Proceedings of the 14th National Computer Security Conference (1991), pp. 362--371.

[Jensen at al. 98] Jensen Ch., Dyreson C., Böhlen M. and others, *The Consensus Glossary of Temporal Database Concepts - February 1998 Version*, Temporal Databases - Research and Practise (1998), Springer-Verlag, pp. 367--405.

[Kautz and Ladkin L91] Kautz H., Ladkin P., *Integrating Metric and Qualitative Temporal Reasoning*, Proceedings of the Nine National Conference of Artificial Intelligence, CA, USA.

[Kumar and Spafford 94] Kumar S., Spafford E., *A Pattern Matching Model for Misuse Intrusion Detection*, In Proceedings of the National Computer Security Conference, Baltimore, Coast TR 95-06 (1994), pp. 11--21.

[Kumar 95] Kumar S., *Classification and Detection of Computer Intrusions*, PhD thesis, Purdue University, 1995.

[Lane and Brodley 98] Lane T., Brodley C., *Sequence Matching and Learning in Anomaly Detection for Computer Security*, 1998.

[Lunt at al. 89] Lunt T., Jagannathan R., Lee R., Whitehurst A., Listgarten S., *Knowledge based Intrusion Detection*, In Proceedings of the Annual AI Systems in Government Conference (1989), Washington DC.

[Liepins and Vaccaro 89] Liepins G.E., Vaccaro H.S., *Anomaly Detection: Purpose and Framework*, In Proceedings of the 12th National Computer Security Conference (1989), pp. 495--504.

[Phrack 96] *IP-spoofing Demystified: Trust-Relationship Exploitation*,

Phrack Magazine, Vol. 7, Issue 48 (1996), http://www.fc.net/phrack/files/p48/p48-14.html.
[Porras and Kemmerer 92] Porras P.A., Kemmerer R.A., *Penetration State Transition Analysis: A Rule-Based Intrusion Detection Approach,* In Eighth Annual Computer Security Applications Conference (1992), pp. 220--229.
[Schlimmer 87] Schlimmer, J.C., *Concept acquisition through representational adjustment, Ph.D. theses*, University of California, Irvine, 1987.
[Shieh and Gligor 91] Shieh S., Gligor V., *A pattern-oriented intrusion-detection model and its applications,* In Proceedings of Symposium on Security and Privacy (1991), Oakland, CA, pp. 327--342.
[Smaha 92] Smaha S., *Questions about CMAD*, In Proceedings of the Workshop on Future Directions in Computer Misuse and Anomaly Detection (1992), pp. 17--21.
[Smaha 93] Smaha S., *Tools For Misuse Detection,* In Proceedings of ISSA'93 (1993), Crystal City, VA.
[Spafford 96] Spafford E.H. Security seminar, Department of Computer Sciences, Purdue University, 1996.

Alexandr Seleznyov 1975, received his M.Sc. degree on Computer Science in 1997 from the Kharkov State Technical University of Radioelectronics, Ukraine. From the end of 1997 he is Ph.D. student of the Computer Science and Information Systems Department at the University of Jyväskylä. His main interests are Intrusion Detection and Machine Learning.

Seppo Puuronen 1949, received his Ph.D. on Computer Science in 1988 from the University of Jyväskylä, Finland. He is acting as assistant professor of Computer Science at the University of Jyväskylä and he has acted as Professor, Associate Professor and Lecturer of Computer Science at Jyväskylä and Lappeenranta University of Technology. His main research interests are Knowledge Engineering, Expert Systems, and Computers in Education.