FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 38 (2013) No. 2

DOI: 10.2478/fcds-2013-0001 ISSN 0867-6356
e-ISSN 2300-3405

DATA WAREHOUSE FOR EVENT STREAMS VIOLATING RULES[1]

Bogdan Denny CZEJDO[2], Erik M. FERRAGUT[3], John R. GOODALL[3] and Jason LASKA[3]

Abstract. In this presentation, we discuss how a data warehouse can support situational awareness and data forensic needs for investigation of event streams violating rules. The data warehouse for event streams can contain summary tables showing rule violation on different aggregation level. We will introduce the classification of rules and the concept of a general aggregation graph for defining various classes of rules violation and their relationships. The data warehouse system containing various rule violation aggregations will allow the data forensics experts to have the ability to "drill-down" into event data across different data warehouse dimensions. The event stream real-time processing and other software modules can also use the summarizations to discover if current events bursts satisfy rules by comparing them with historic event bursts.

Keywords: data streams, data warehouses, drill-down operation, aggregation, data forensics, situational awareness

1. Introduction

There are many types of data streams that can contain events that are dangerous or malicious [4]. A very good example is the cyber security area where different methods were proposed to detect computer intrusion by analyzing various streams of data including

[1] This research was supported in part by an appointment to the Higher Education Research Experiences (HERE) Program at the Oak Ridge National Laboratory (ORNL) for Faculty, sponsored by the U.S. Department of Energy and administered by the Oak Ridge Institute for Science and Education. This research was also funded by LDRD at Oak Ridge National Laboratory (ORNL). The manuscript has been authored by a contractor of the U.S. Government under contract DE-AC05-00OR22725. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

[2] Department of Mathematics and Computer Science, Fayetteville State University, Fayetteville, USA, email: bczejdo@uncfsu.edu

[3] CSIIR Group, CSE Division, Oak Ridge National Laboratory, Oak Ridge, USA, email: ferragutem@ornl.gov, jgoodall@ornl.gov, laskaja@ornl.gov

network logs [5, 6, 7, 8, and 9]. One of the most practically accepted methods to detect the malicious events is application of rules that define unsafe events [4]. The relationships between malicious events and unsafe events are not always obvious and the interpretation requires human intervention. In order to make the proper interpretation the decision-maker needs to have situational awareness that includes both current and historical knowledge about rules' violations. A data warehouse can provide not only historical awareness but also support the analysis of the current stream of data. In addition, the data warehouse for events violating rules can provide a significant assistance for data forensics [1, 2, and 3].

Our approach, discussed in this presentation, is to enhance the network intrusion detection as presented in [14] to include the rule violation analysis. We extend the aggregation graph for event stream data warehouse to include the rule violation information. The aggregation graph includes fact table and the set of summary tables (aggregations) for rule violations. The fact table contains all network events with the indicators of rules violation. We use the definition of aggregation hierarchy graph and its levels as described in [14] to describe the aggregation hierarchy graph with rule violations. So the emphasis of this presentation is on specifying the directed acyclic graph (DAG) of aggregations and linking it with various groups of events that violate the rules. The traditional data warehouse architecture [11, 12, and 13] can be expanded to provide such constructs. One of the important extensions is a specialized drill-down operator to operate on the graph containing rule violation summarizations. The effective use of such an operator can be improved if flexible options are available to the user and information about these options is clearly presented to the user [10]. Such interface provides important functions both for data forensics and to increase situational awareness.

This paper is organized as follows. In Section 2, we describe the system architecture. In Section 3 we describe rules and also discuss event streams that can contain rule violations. Section 3, also includes and database model in the form of a star schema. In Section 4, summary tables corresponding to different aggregation level of events violating rules are presented. Section 5 contains example of application of our approach. Section 5 contains the summary.

2. System Architecture for processing of Event Streams with Rule Violations

The system architecture for processing Event Streams violating rules is shown in Figure 1. It is built based on a traditional data warehouse architecture and contains three main components: *Event Streams*, the *Event Stream with Violations database* and the *Event Stream with Violations data warehouse*. The *Event Stream* is recorded in the *Event Stream with Violations database* after discovering rule violations. The *Event with Violations database* is used periodically to update *Event with Violations data warehouse*. The periodical update includes update of the *data warehouse Violation Summary*.

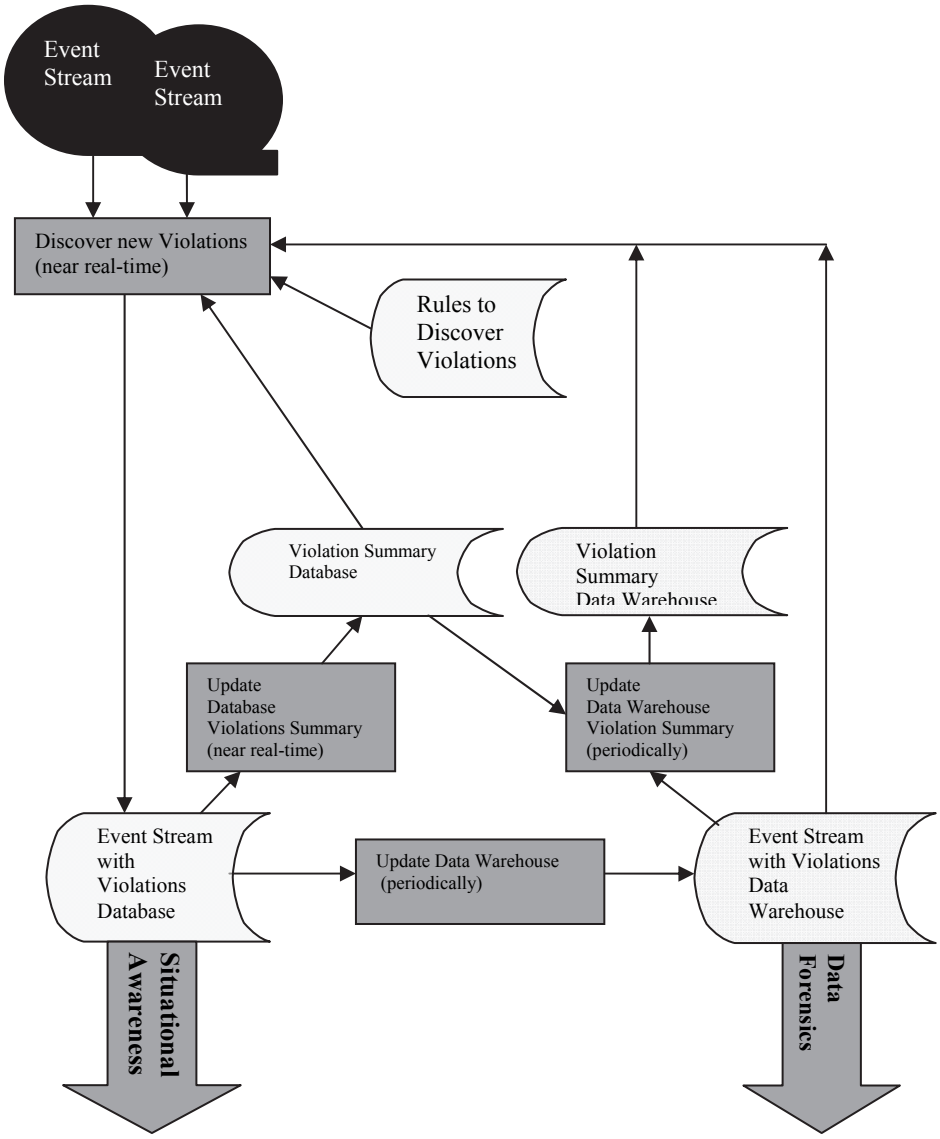

Figure 1. System Architecture for processing Event Streams with Rule Violations

There are five important characteristics of our approach as shown in Figure 1. First, *Event Streams* needs to be processed before being stored in the *Event Stream with Violations database*. The processing includes checking for any rule violation and storing this information. *Initially*, the individual events are checked for violations and if necessary tagged with the rule identification (by the module *Discover New Violations*). Second, the

appropriate summarization of violations is performed (update Violation Summary Database). The groupings of events and their violations need to be uniquely identified for possible future retrieval. Third, the summarization is used for checking aggregate rules (violations of event bursts). This is why the module Discover Rule Violation might require querying both database and warehouse Violation summary tables. Therefore there is also a need for the module update Violation Summary Database to provide timely summarization of rules' violations. Related with that is an interesting problem of "violation propagation" to the lower level. It means that events that were already processed need to be virtually or explicitly marked with a violation mark and tagged with the proper rule identification. Fourth, a proper graphical user interface needs to be developed to support situational awareness for the decision-maker. Fifth, also a proper graphical user interface needs to be developed to present historical summary information about events and the rules violations to support data forensics. The choice of appropriate summary of past violations tables should assist in providing such support.

3. Event Streams, Rules, and Database Support

Generally, in a cyber security system, there are recorded violations and violations to be discovered. The recorded violations are available as various detection logs e.g. log of Intrusion Detection System (IDS). The violations to be discovered are computed based on communication or processing logs e.g. log of firewall events.

The log of Intrusion Detection System (IDS) contains already recorded violations. It will be referred here as IDS streams of events. Each event in the IDS stream describes a network event that has been already identified as violating IDS rules. The event, therefore, contains information about priority of violation (number from 1 to 5 with the deceasing priority i.e. 1 means the highest priority) and classification of the rule that was violated (validated?). There are several additional components of this data event. The most important are source IP address, destination IP address, source port, destination port, and time of the event.

The firewall data log contains information that can be used to compute rule violations. The firewall logs are streams of events. Each event describes the activity of the firewall. There are several components of this data event. The main are source IP address, destination IP address, source port, and destination port. In addition there are the firewall event codes including action (e.g. Build) and the protocol (e.g. TCP). Time, as usually, is also included.

There are different types of rules that can be used to discover new violations and each type of rule has a different violation level (as in the case of recorded violations). The rules to discover violations can be classified in various ways. Our classification is based on the following violation types:

- Policy Violation
- Network Role Violation
- Network Address Violation
- Burst Violations

The Policy Violation rule is to identify all events that were against the established policy e.g. disallowing file sharing, relay chat, and remote connection, what practically meant for our data stream to disallow usage of the specific the port numbers. The Network Role Violation rule is to check if the communication between the source and destination machine was according to the roles assigned explicitly by the network architecture e.g. to check if workstation communicated only with Web server or DNS server communicated only with another DNS server. The Network Address Violation rule was to check if the proper IP was used according to the topology of the network. The burst violations are based on rules applying to aggregations of events, e.g. sequence of the Build events in the firewall stream specifying various ports for the same IP address what is sometimes referred as port sweep. There are also other rules that are beyond this presentation such as dynamic rules usually constructed automatically based on one of the artificial intelligence techniques and anomaly detection techniques.

The level for discovered violations was assigned based on two criteria:

- How high or low is asset value present at the discovered violation
- If the discovered violation is somehow related to one or more recorded violations

The list of high asset value objects includes DNS servers, Firewall devices, and Web servers. The discovered violations can be checked against IDS recorded violation. If the discovered violation is related somehow to the recorded violations then the level of the violation is adjusted for this fact. Practically, it meant for our streams, that the level of the violation was adjusted if the source IP address in firewall event violating any rule is also used in the recorded violations in IDS in an approximately the same time.

The first three violation types can be discovered by a concurrent processing of event streams and storing the information about newly discovered violations in appropriate tables. There are many possible approaches to create an event database with violations:

- storing only events with violations vs. storing all events
- storing events with violations separately vs. storing all events together
- storing only aggregated data vs. storing all events
- using a star data warehouse model for both current event stream and historical event stream vs. using different data models

There are important implementation issues but for the uniformity of the approach we use one data model for current and historical event database. Also, we choose an approach with access to all events since we need to have various statistics for rule violation e.g. percentage of events violating rules. We chose a data warehouse model since it the best to describe aggregations. In our approach, therefore, the event stream database with violations can be represented by the star schema shown in Figure 2. The model includes the three main dimension tables: Source Machine, Operation, Destination Machine and Time, and a fact table, containing the firewall events. The role of each table is as follows. The Source Machine and Destination Machine dimensions contain the information for a specific machine e.g. IP address, IP address group and port used by the machine. The key identifier for each object in the Machine dimension is the composite attribute {IP_Address, Port}. The Operation dimension contains the information about the protocol, action and other descriptions and classification. The key identifier for each object in the Operation dimension is the composite attribute {Protocol, Action}. Here, we assume that the Time dimension can contain some time classification e.g. part of day. The key identifier for each object in the Time dimension is the composite attribute {Day_Hour}.

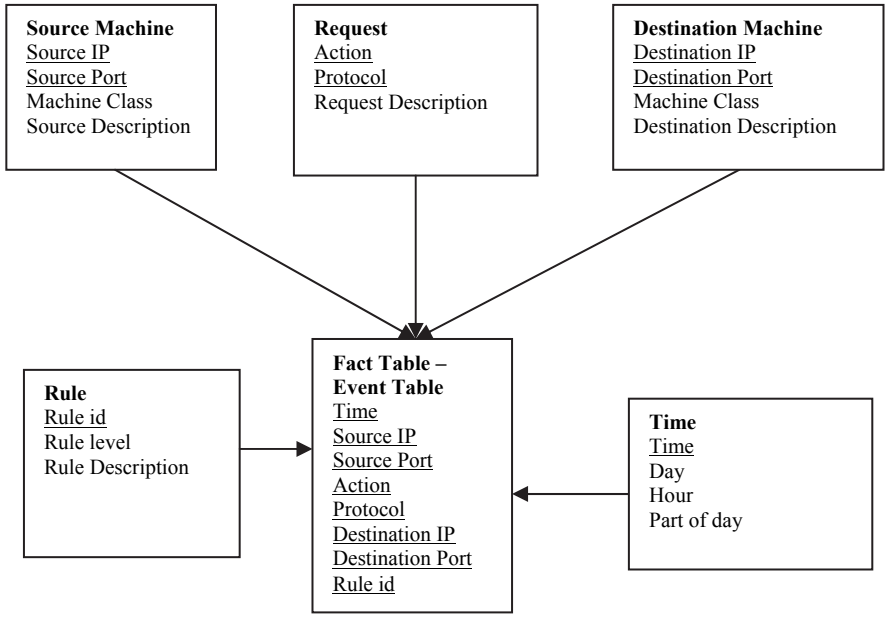

Figure 2. The initial star schema for the cyber-security data warehouse

In our approach we emphasize the need to record the events that violate the set of given rules. There are different types of rules and each rule has a different violation level. For the simplicity of the discussion we will assume here that each type of rule uniquely determines violation level. The *Rule* dimension, therefore, can be added to the star schema to address this important requirement as shown in Figure 2. The *Rule Violation* dimension can contain rule id, rule level and rule description. The key identifier for each object in the *Rule Violation* dimension is the attribute {Rule_id}. For the events that do not violate any rule

The fact table refereed here as event table has a log of all firewall operations made from one machine to another machine at a specific time. The event table has eight key attributes From_IP Address, From_Port, To_IP_Address, To_Port, Day_Hour, Protocol, Value, Rule_id referred to also as index attributes. The additional attributes are allowed for all dimension and fact tables.

4. Violation Summary Database

In our approach one of the fundamental operations on fact table is violation information summarization stored as violation summary database. Generally, the tables obtained by data aggregation can provide appropriate summary information. Various summary tables contain different grouping of events as aggregate objects e.g. all traffic from a given IP address. Summary tables are introduced to provide a baseline for automatically discovering event bursts violations and to provide a better situational understanding of a cyber security analyst. Therefore, it is important to have the proper design and implementation of the hierarchy of summarization tables.

Let us use the definition of aggregation hierarchy graph and its levels as described in [14] to describe the aggregation hierarchy graph with rule violations as shown in Figure 3. Level zero will be for the event table (referred to as T01) which describes all events with violations if any. The level one will include summary obtained from the event table by reducing it by a single index attribute, e.g. the summary table T11 is obtained by summarizing information about all T01 events that came from the same Source Machine, had the same Request, were directed to the same Destination Machine, and were involved in the same violation (Rule id) , but happened any Time. The second level will be the one consisting of summary tables containing information about aggregations obtained from the first level by reducing it by an additional index attribute. For example, the summary table T21 could be obtained by summarizing information taken from T11 for each group that came from the same Source Machine, had the same Request, and were directed to the same IP address. The different tables have different data granularity.

The granularity of the tables corresponds to the granularity of violation summary. The most aggregated table can show general measures such as number (or percentage) of all violations. Once this general measure is available for cyber security analyst, the information about more specific rule violations can be requested.

In the case of network traffic data, the timing of violations should be emphasized and the simple aggregations are not sufficient. The complex violation aggregations based on time are of crucial importance. Practically, such aggregations use the window concept or are based or event proximity. In our project we use fixed time window concept i.e. we aggregate of all events within the windows. Each window needs to be uniquely identified since we want to perform various window based computations. The beginning time of window was be used for that purpose. The information about window based aggregations can be stored in a summary table, e.g. the new table T01A1as shown in Figure 3. The newly created table can be a starting node of a new hierarchy i.e. data in table T00A1 can be summarized into another table T11A1.

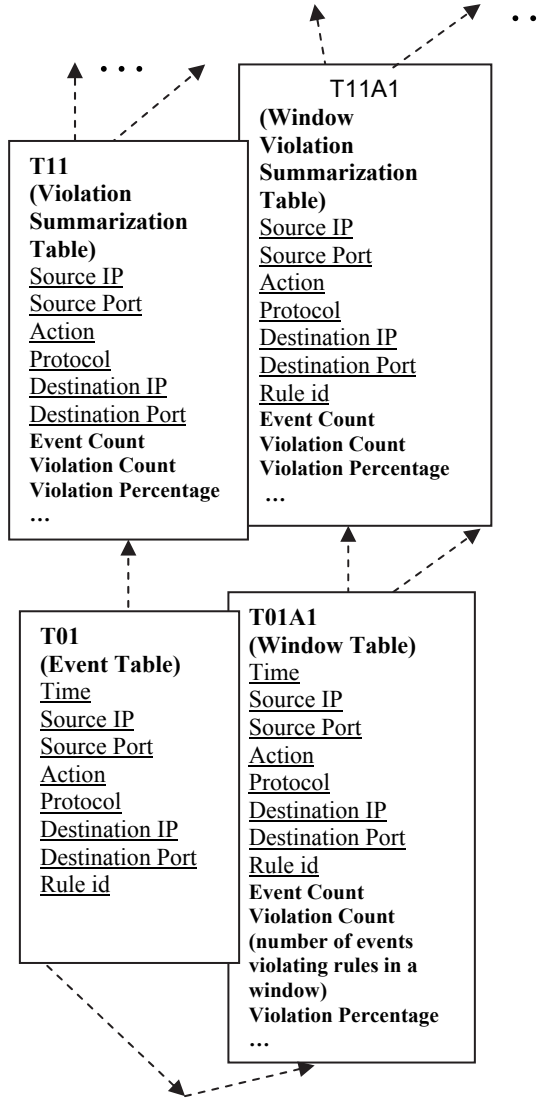

Figure 3. The hierarchy including complex summary tables for the window based aggregations

5. Using Violation Summarizations for Situational Understanding

We implemented our cyber-security data warehouse model on the IEEE Vast 2012 Situational Understanding and Cyber-forensics data. The finest representation of the data in the challenge is composed of the following 7 parts: Source IP, Destination IP, Source

Port, Destination Port, Protocol, Time, and Message Code. By varying the grouping structure of the 7 parts we observed noteworthy cyber-security events.

First, the most general rule violation information was retrieved based on the table on the seventh level of aggregation (called table T71) with the attribute Violation Percentage indicating percentage of events in all data set violating the rules. Generally, this number is pretty low but for the VAST 2012 data, was close to 30%.This anomaly motivated a deeper traversal of the graph of aggregations. We used the part of the graph hierarchy corresponding to window based aggregations. These aggregations were constructed for non-overlapping windows with the five minutes window duration.

The aggregations corresponded to the summary table (called table T62) with the single index attribute Time indicating the beginning of the window, and non-index attribute *ViolationPercentage* indicating percentage of events in each window violating the rules. Drilling-down to this grouping, we were able to discover of two windows where over 90% of traffic was against policy. Further drill-down to the summary table (called table T52) with the two index attributes Time and Source IP allowed us to discover one source IP that was responsible for all traffic (most of the traffic violating rules). This indicated a possible misdirection or man-in-the-middle attack as all traffic may be routed through an intermediate contact.

6. Conclusions

In this presentation, we considered the problem of generalization of the data warehouse aggregation graph and drill-down operator to be used for streams with rules violations. We introduced the concept of a general aggregation graph that contains both data aggregations and summation of rule violations related with each aggregation level. The aggregation graph can be explored using drill-down operator for data forensics and to achieve situational awareness of stream data rules violations.

References

[1] Michael T. Goodrich, Mikhail J. Atallah and Roberto Tamassia, Indexing Information for Data Forensics, Lecture Notes in Computer Science, 2005, Volume 3531/2005, 206-221.

[2] Federico Maggi, Stefano Zanero, Vincenzo Iozzo, “Seeing the invisible: forensic uses of anomaly detection and machine learning” ACM SIGOPS Operating Systems Review, Volume 42 Issue 3, April 2008, 51-58.

[3] Hal Berghel “Hiding data, forensics, and anti-forensics”, Communications of the ACM CACM, Volume 50 Issue 4, April 2007, 15 – 20.

[4] Sushil Jajodia, Peng Liu, Vipin Swarup, Cliff Wang, 2009, Cyber Situational Awareness: Issues and Research, Springer Publishing Company, 2009.

[5] Ferragut, E.M.; Darmon, D.M.; Shue, C.A.; Kelley, S., Automatic construction of anomaly detectors from graphical models”, IEEE Symposium on Computational Intelligence in Cyber Security (CICS), 2011 IEEE Symposium on

[6] Sung-Bae Cho, “Incorporating soft computing techniques into a probabilistic intrusion detection system” IEEE Transactions on Systems, Man, and Cybernetics, May 2002, vol. 32 , issue: 2, pp: 154 – 160.
[7] Denning, Dorothy, "An Intrusion Detection Model," Proceedings of the Seventh IEEE Symposium on Security and Privacy, May 1986, pages 119-131.
[8] Teng, Henry S., Chen, Kaihu, and Lu, Stephen C-Y, "Adaptive Real-time Anomaly Detection Using Inductively Generated Sequential Patterns," 1990 IEEE Symposium on Security and Privacy
[9] Jones, Anita K., and Sielken, Robert S., "Computer System Intrusion Detection: A Survey," Technical Report, Department of Computer Science, University of Virginia, Charlottesville, VA, 1999
[10] Czejdo. B, Taylor M. and Putonti C.,(2000); “Summary Tables in Data Warehouses”. Proceedings of ADVIS’2000.
[11] Gupta A., Harinarayan V., and Quass D. (1995); "Aggregate-Query Processing in Data Warehousing Environments", *Proceedings of the VLDB*.
[12] Bischoff J. and Alexander T. (1997); *Data Warehouse: Practical Advice from the Experts*. New Jersey: Prentice-Hall, Inc.
[13] Widom J. (1995); “Research problems in data warehousing", *Proceedings of the 4th Int. Conf. CIKM*.
[14] Bogdan Denny Czejdo, Erik M. Ferragut, John Goodall and Jason Laska “Network Intrusion Detection and Visualization using Aggregations in a Cyber Security Data Warehouse”, accepted for publication in *International Journal of Communications, Network and System Sciences*, (IJCNS)

Presented at the 16th East-European Conference on Advances in Databases and Information Systems September, 17-20, 2012, Poznań, Poland