RC23011 (W0311-147) November 19, 2003
Computer Science

IBM Research Report

PEC: Post Event Correlation Tools for Network-Based Intrusion Detection

Mark Mei, David A. George, Mark Brodie, Charu Aggarwal, Sheng Ma, Philip S. Yu
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

PEC: Post Event Correlation Tools for Network-Based Intrusion Detection

Mark Mei, David A. George, Mark Brodie, Charu Aggarwal, Sheng Ma, Philip S Yu
IBM T.J. Watson Research, Hawthorne, NY, 10532 USA
{gmei, dag, mbrodie, charu, shengma, psyu}@us.ibm.com

Abstract

We describe and implement an approach to intrusion detection using data mining. We focus on analyzing network-based alert logs, and show how valuable information can be extracted from this data using an integrated collection of tools and technologies. These include data enhancement using topology information, event rate analysis, and temporal association. Applying these techniques on real data leads to a considerable reduction in false alarm rate while also allowing low frequency attack patterns and coordinated attacks from multiple sites to be detected.

1. Introduction

Intrusion Detection Systems (IDS) play a vital role in ensuring the security of modern computer installations. Such systems are needed in order to detect hostile activity and to respond appropriately. As networks continue to expand and become more exposed to a diversity of sources (both hostile and benign), IDS need to be able to deal with a large (and ever-increasing) flow of alerts and events. For this reason automatic procedures for detecting and responding to intrusions are becoming increasingly essential.

However, deploying and maintaining such an IDS for a large production system is not an easy task. In practice such systems encounter certain difficulties which must be surmounted before they can be expected to provide value. First, how do we tune a complicated IDS so as to achieve an acceptable level of false alarms while at the same time ensuring that real intrusions are not overlooked? It is well known [13] that IDS often generate a large number of false alarms, so techniques are needed that can identify these false alarms so as to focus attention onto the most relevant events. Second and more importantly, how do we ensure that an IDS continues to behave in the way that we have designed and how do we identify patterns that signify intrusion activities? We believe that historical data is the key information resource for addressing these issues. In this work we develop a data mining system with an integrated set of analysis methods which are used to analyze a large amount of IDS alert log data on an on-going basis. We show that this allows us to discover interesting, previously unknown and actionable information, including reduction of the false alarm and detection of patterns of intrusive activity.

Our work in the Post Event Correlation (PEC) project was started with the goal of providing a multi-site web-hosting service provider with a comprehensive understanding of the trends and changes to trends in the behavior of their networks and with the means to assess emerging problems that are subtle in nature (e.g., below the normal threshold for generating alerts from IDS probe data). The project established a partnership between IBM TJ Watson Researchers and IBM Global Services (IGS) e-business Hosting Security units. IGS provides Network Intrusion Detection System logs (NIDS data) and expert opinion that assists with identification of some of the observed phenomena. TJ Watson Research develops tools that can be used by IGS to extract valuable information from this sort of event log data.

This paper focuses on off-line IDS alarm analysis, in which we aim to convert a large amount of historical data into valuable, actionable information and knowledge. This poses several challenges. First, the data volume can be huge - a typical IDS for a moderate-size installation may generate many thousands of events every day. Second, as an individual alarm may not provide much insight, a tool is needed that can help to visualize a large amount of data and reveal temporal associations and correlations. Third, temporal information and temporal patterns are key indicators for intrusions. This requires the development of algorithms that will automatically discover temporal patterns. Fourth, other types of information, such as network routing information, may provide useful clues for identifying intrusion behaviors, so we must be able to integrate and analyze multiple divergent sources of information. Last, since data arrives every day, reports that summarize significant patterns or trends need to be produced automatically on a regular basis.

This paper reports our work in addressing the above issues. We present an architecture and a data analysis process that allows us to exploits multiple sources of information (including network alerts, logs from multiple sites, topology information, etc.) and provide results in an on-going base. We develop a set of integrated visual and interactive analysis tools that assist an analyst in browsing through a large volume of alert data. We utilize several data mining algorithms that can disover temporal patterns, such as temporal association and event bursts. Finally, we report several cases where interesting, previously unknown information can be uncovered by applying different analysis tools to thousands of IDS events. For example, we show that many false alarms can be removed (over 30% in one case), yet at the same time low frequency intrusions can be detected. We also present a method for using stream clustering in order to detect network intrusions in real time.

This paper is organized as follows. Section 2 discusses background material and related work. Section 3 describes the overall system architecture: the raw event logs are first enhanced in a pre-processing step; then the resulting alarm database is analyzed using a variety of data mining techniques; and finally a report is issued identifying the most important intrusions. Sections 4 and 5 explain the core technologies that are used: data preprocessing (including topology analysis), and data-mining (including static analysis, summarization, event-rate analysis, temporal association, and stream clustering of intrusions). Section 6 describes an application scenario illustrating the integration of these techniques. Section 7 presents our conclusions and directions for future work.

2. Background and Related Work

There are many sources of background information. The Tivoli Risk Manager documentation [14] describes the pervasive problem of securing Internet resources wherein problems arise from both within and outside the managing organization. Added to this is the problem of managing the many "probes" providing state information and identifying incidents that need immediate attention. There are many competitors offering products and services to tackle the "real-time" aspects of risk managing. All, though, have a common practice: alerts are prioritized and provided to a human expert who then must approve actions, such as ACL (Access Control List) changes or notifications to service providers, etc. The expert is faced with a daunting investigative task. Our focus is on the historical analysis of alerts covering hours or months of history.

Historical IDS data analysis has been commonly used, but mostly manually to identify suspicious intrusion patterns. For example, Green [9] discussed several intrusion patterns founded in IDS logs. Dunigan et al. [3] argued that correlating information across multiple information sources is important.

The use of data mining for intrusion detection is becoming increasingly popular. These approaches can be classified into two groups, depending on whether data with already labeled attacks is available ("audit data") or not ("log data"). In audit data the labeled attacks can be generated either by deliberately conducting attacks ("ethical hacking") [11] or by constructing artificial anomalies [8]. However in practice labeled data is often difficult and expensive to obtain.

We focus on event log data that does not contain known intrusion events – here the challenge is to detect informative patterns that are reported to the human operator as a guide to further action. Association rule mining [13, 11] seeks to find informative associations between alarms. Correlation methods [7] and alarm clustering [10] have been used to reduce false positive rates. [15] developed an algorithm for detecting "back-door" attacks. Other approaches [12, 1] model normal system behavior that is then used as a baseline for detecting abnormal behavior

Our work is motivated by the needs of large-scale analysis of IDS data. It differs from previous work in several ways. First, we develop an architecture and data analysis process that supports large scale, interactive analysis. Second, to allow the user a flexible, exploratory approach, we integrate a wide range of different methods, including visual summarization, pattern analysis and others. This allows the user to select the appropriate analysis tool based on the results of previous exploration – the synergy between the different tools creates considerable value. Different approaches can be combined to accomplish a certain task. Last, we developed several new methods that provide a systematic way to analyze the topology, temporal and aggregated information.

3. System Architecture and Data Analysis Flow

A typical real-time IDS monitors network traffic and host behavior and correlates information to detect intrusions. Sensors that monitor network traffic are placed at a firewall as well as at key servers, such as web servers. When a predefined signature or abnormal behavior is detected, an alert is generated and forwarded to a correlation engine whose role is to consolidate the alerts from different sources and locations and produce a high level alarm which leads, if needed, to further action being taken.

This paper focuses on the off-line analysis for IDS alarms. In particular, we discuss a set of tools and analysis methods developed for systematically analyzing IDS data in

the PEC project. Our data analysis process starts with data pre-processing, which normalizes the raw events into a format that can be stored in a database; and, then enhances the raw events by augmenting the data with other relevant information not found in the raw event logs, such as routing information, domain names of the IP addresses, and so on. This process is described in Section 4. The data are stored, merged and maintained in a database. Then, an integrated data analysis tool is used to analyze any collection of events from the database – this is described in Section 5. The tools can be used in both batch and interactive modes. Using batch mode, reports, such as summarization reports, can be generated on a regular basis. These reports can be viewed by an analyst through web interfaces or other means. If some problems are identified, either by analyzing the report or because some unusually suspicious events have occurred, an exploratory mode can be used by the analyst to interactively visualize and mine events from the database. Data analysis may lead to some corrective action being taken; for example, modifying the sensor rules to reduce the false alarm rate, modifying the correlation rules to capture newly identified intrusion patterns, and collecting evidence about hackers that can be used as a basis for additional actions.

4. Data Preprocessing

In this section we describe the initial format of the data, followed by the data parsing and enhancement that are used to transform the data into a database format that is acceptable to the suite of data-mining tools described in the subsequent section. It is assumed that the data stream consists of a set of multi-dimensional records $\overline{X}_1, \ldots, \overline{X}_k \ldots$ arriving at time stamps $T_1, \ldots, T_k \ldots$. Each $\overline{X}_1, \ldots, \overline{X}_k \ldots$ is a multi-dimensional record containing d dimensions which are denoted by $\overline{X}_i = (x_i^1 \ldots x_i^d)$.

4.1 Sample data input format and parsing

Raw IDS alarms may come in a variety of different formats. However, there are a couple of common information types. To illustrate this, we show below one sample message of a network IDS log (the first element is the Alert Level, … the fifth element (i.e., d=5) is the destination IP):

Alert Level: 3 **Date:** Fri Feb 18 2000 22:31:44 **Sig :** IDS-S1-Host-Page:N/A DOS - Possible connection flood **Host:** 10.255.123.252**:**[50] **to** 192.168.222.222**:**[210]

In this network IDS log, each line begins with **"Alert Level:** " and contains tokens of information in a non-delimited format. In our example, [] denotes optional content, **bold** represents constant information. There are four key attributes: signature for representing the type of an alert; source IP address and port number of an attacker; destination IP and port number for a victim host of this attack; detailed message for additional exaplanation of this alart. In our example alert above, its signature is IDS-S1-Host-Page:N/A DOS; the source IP and port number are 10.255.123.25 and 50; the destination IP and port nubmer is 192.168.222.222 and 210; and the detailed message is "Possible connection flood". Clearly, this alert signified a possible connection flood (or DOS attack) from 10.255.123.25 to 192.168.222.222.

4.2 Data Enhancement

In addition to the information in a raw message, other information can be obtained and augmented into original alerts. Two types of information are important for further analysis. They are the router information and IP-name information. To obtain the routing information, a trace-route is performed to both the source and destination addresses. This yields the "last drop router", or "LDR". The LDR is a router that is closest to the designated source or destination. This additional LDR information can be very informative when looking for patterns using data mining, as shown in Section 5.2.

To obtain the IP-name information, an NS-lookup is performed on both the source and destination addresses to obtain their IP-name, if possible. This is very useful in identifying important customer sites, distinguishing foreign from domestic sites, and so on. It is also helpful to the human operator to be able to see the site-name when visualizing the data, as shown in Section 6.

4.3 Alarm Database

Once the log data has been enhanced, it is incorporated into the overall IDS alarm database. Thus the result of the data pre-processing is a comprehensive IDS alarm database where the data has been arranged in a format that is acceptable for a variety of data-mining tools.

We note that there are several advantages in using the database. First, it allows for easy integration of information from multiple sources, multiple sensors and multiple sites. Second, it allows us to incorporate multiple analysis tools. In our case, we use not only state-of-the-art reporting tools such as Crystal Reports, but also run our tools for interactive, visual analysis and for mining events to gain a deeper understanding. Third, knowledge about alerts can be easily integrated to support real time event correlation.

4.4 PEC Workbench

The Post Event Correlation Workbench is a middleware, Windows® GUI that allows the user to analyze IDS logs, and many other data sources in a systematic and automatic way. The general functional parts are accessible on separate tab identified dialog panes and provide the following functions:

- Data Sources specifier and filter
- Screen Scraper
- Batch Data Preparation Script Executor
- Reports Launcher
- "Blacklist" manager

of establishing a set of one or more databases to use as source data, selecting which sources to process at the moment, specifying the range of dates and times per source then proceeding with processing. Processing can be interactive or batch. Interactive steps include the selection of tabular and drill-down reports, customization of blacklists used for various correlations and the launching of batch scripts. A blacklist is simply a collection of host IP addresses (possibly including other attributes) that have been extracted by processing. The Screen scraper allows for incorporating IPs from web pages, XML, or flat files. The "blacklist manager" allows the user to save named lists in the database, as well as aggregate lists.

5. Data Mining

Data mining refers to technologies and algorithms that analyze the underlying data looking for patterns and salient characteristics that are interesting, unusual or informative. An IDS utilizes data-mining for a variety of reasons, including identifying intrusive and dangerous behavior, removing false alarms, discovering new and unusual attack patterns, locating correlations between sites, and so on. The data-mining tools are listed here; each is then described in detail below. These tools constitute a piece of software called the Event Browser/Event Miner [5].

1. Summary Analysis – this step provides an overall summary of the data by grouping together common records – e.g. which signatures occurred most often, which hosts generated the largest number of alarms, and so on. This is useful in determining where attention should be focussed.

2. Topology Analysis – this step groups together sites that seem to be associated with one another, using, for example, router or signature information. This creates a more compact representation of the data in which multiple sites are grouped together and higher-level relationships and correlations between them can be detected more easily.

3. Temporal Analysis – this step represents along a time-line the data of whichever of the variables is of interest (e.g. signature). This allows long-term historical trends to be identified, and also provides clues as to the presence of coordinated simultaneous attacks from multiple sites.

4. Event-rate Analysis – this step computes the flow of the number of events occurring in a specified time window. This allows event bursts to be detected, which can further guide the intrusion detection process.

5. Temporal Association - this step seeks to find periodic patterns and signature associations that often occur together within a certain time window. This can be used to find common attack patterns and relationships.

6. Stream Clustering- this process seeks to identify clusters of abnormal behavior in the network intrusion data in real time. This technique can find some of the subtle trends which cannot be found currently.

We now describe each of the above data-mining tools in more detail.

5.1 Summary Analysis

Summary reports are structured to provide the analyst with a basis of information in order to help decide which of the additional tools should be used. Any of the database fields can be summarized by ordering the data counts using that particular field. The results are displayed in a visual format using the Event Browser [6] that uses coloring techniques to convey the information in a graphically informative way.

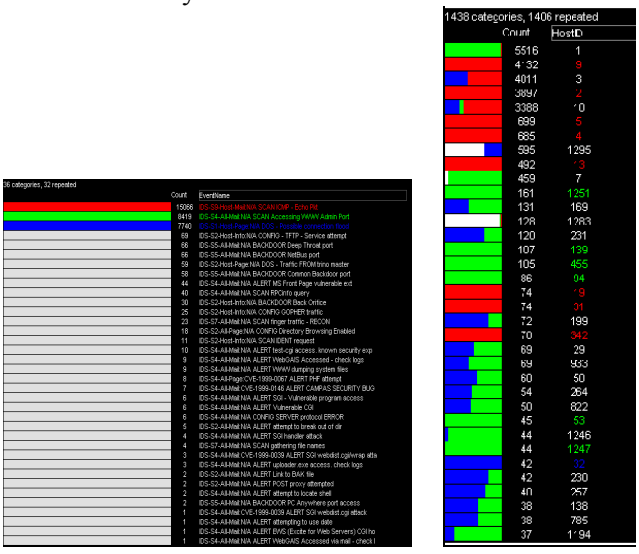

Figure 1. An Example of Summary Analysis. The three most common signatures, listed on the left are colored red (top), green (middle) and blue

(third). The right hand side is the (enumerated) source addresses based on the color code of the left hand side and show relative proportions per address.

Figure 1 depicts one day's worth of alerts. In the left-hand chart, out of a total of 31826 events, the three most frequent ones are labeled red, green, and blue. About 47% of the attacks were generated by ICMP Echo packets (colored red), 26% by attempted access of the port defined for WWW administration (colored green), and 24% are connection-based port-floods, a type of Denial-of-Service (DOS) attack (colored blue). There are about 3% other attacks (colored white).

In the right hand chart, one can see that the green signature was the major activity for host ID 1, which caused 5516 hits out of a total of 8419 total hits. Similarly, host ID 169 has approximately 50% of each of the DOS and Accessing Admin Port type of signatures. As an example of summarization, the alerts are shown ordered from highest to lowest hit count based on signatures.

This example will be used to illustrate a number of additional data mining tools below, which will demonstrate the power of using color-coding to link across different views. The summary analysis serves mainly to focus our initial attention on the most common types of events.

5.2 Topology Analysis

In our concept of the organization of sources and destinations in the Internet, we can only partly determine the pathway from source to destination because our analysis is based on a reference point separate from either of true source or destination. Consequently, we construct a mapping close to the actual crime scene by determining via traceroute, what routers are nearest both the sources and destinations. These serve as reliable anchor points in the topology diagram and are sufficient for datamining. Sites can be grouped using their "last drop router" (LDR). This results in sites being collected into domains and regions (such as a corporation), and this information can lead to the identification of false alarms, as we now explain.

Suppose D_1 is a collection of destination addresses that share a common LDR. D_2 is another collection of destinations that share a common LDR different than the previous one, and so on. Some routers share a common property, for example, they are owned by the same corporation (e.g. IBM). All the destination addresses who's LDR is in this collection are grouped together into a "DR", representing the destination router. A count called DR-Count is associated with each DR; it is computed by summing the number of alerts for each destination address whose LDR is in the group.

The same technique is applied to the source addresses, producing groups called "SR" consisting, for example, of all source addresses whose associated routers are owned by the same corporation, with an associated SR-Count. The ratio SR-Count/DR-Count gives the fraction of alerts generated from internal sources. It is most likely that these are not hostile intrusions - they are probably the result of incorrectly configured sensors or other causes.

As an example, in Figure 2 three destination servers share the same LDR – "ibm-router2". These servers are grouped together in the same DR, as shown in the figure. The numbers in parenthesis after each server and router denote the counts. Notice that the sum of the counts of the servers in the DR equals the total count at the router. Another DR containing two servers with LDR "ibm-router1" is also shown. Similarly, three source routers are shown belonging to the same SR. The destination router for this SR is "ibm-router1", the same router. Thus of the 10997 alerts associated with "ibm-router1", 8160 go to the left DR and the remainder to the right DR. The count for the right DR is 11753 because the router "ibm-router2" has other source routers, as shown.

Note that all the 10997 alerts associated with source-side "ibm-router1" go to destination routers in the same company (IBM). Such a close association of addresses owned by the web provider may result in a large number of of false positives. Recall from Section 5.1 that there were 31826 alerts in total. Thus over 34% of the alerts have been identified as originating from internal sites – these were confirmed to be false alarms caused by misconfigured sensors.

One could envisage a more extensive hierarchical grouping of source and destination routers, e.g. according to geographical location (cities, countries etc), groups of enterprise, and so on. This could provide useful information about attack patterns.

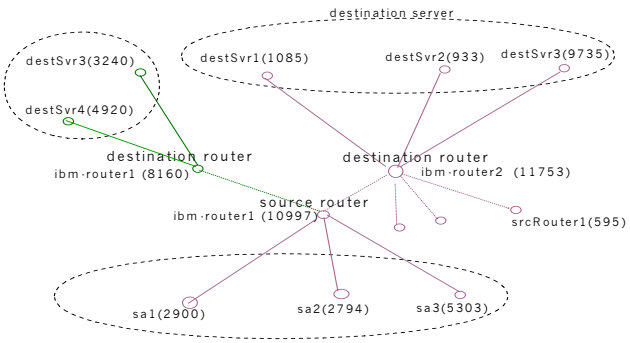

Figure 2: Topology Analysis, showing source and destination routers grouped together by common LDR, with associated alert counts.

5.3 Temporal Analysis

In Temporal Analysis any variable of interest is displayed in a 2-dimensional graphic display, where the x-axis represents a time-line and the y-axis represents the variable. Each (x,y) point that is plotted in the display records alert activity "y" at time unit "x". The variable can be Signature (SIG), Source Address (SA), Destination Address (DA), Source Router (SR), Destination Router (DR), etc.

For example, suppose the variable of interest is Signature. One can focus on a particular time or time window and examine all the signatures that occurred at that time or during that time window; this appears along the vertical dimension. One can also examine the time intervals during which alerts with a particular signature or set of signatures continue unbroken; this appears along the horizontal dimension.

The graphic below shows both destination addresses and alert type (signature) for the events summarized in Section 5.1. The left side shows the destination addresses (only the colored events are shown – the other events were dropped from the display by operator selection). To make the display easier to view each destination address is listed as a number, not the "real" IP address. The right side shows the signature.

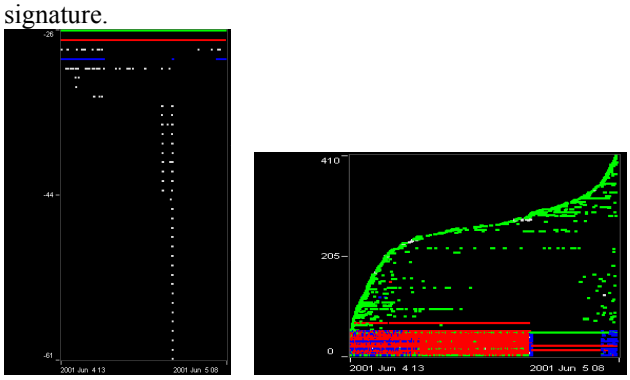

Figure 3. Temporal Analysis. The left chart shows destination addresses on the vertical axis, the right chart shows alert signatures. Note that many alerts at different destinations have the same signature, so the right side appears much less dense.

The concentration of events in the addresses near the bottom of the left side represent sustained attacks that are directed at the web server. The dispersion of green dots throughout the diagram reveal a sequential scan of destination addresses that occur over a day's time. This may be due to a number of root causes, including intrusions and off site scan.

The right side shows that the red ICMP and green port-scan alerts occur continuously in time, while the blue DOS events occur in a number of scattered patches. We also notice a vertical row of dots – a "spike" of alerts of many different signatures occurring at the same. It is preceded by a much smaller spike containing a subset of the same signatures. This is clearly suspicious and leads us to consider our next tool, event-rate analysis.

5.4 Event-rate Analysis

The event-rate is the number of events (alerts) that occur in a particular time window selected by the user. As time progresses new events are included in the window as old ones drop out. The user can set a threshold – if the event-rate exceeds the threshold the corresponding "burst" of events can then be examined. By using the coordinated coloring across different views (e.g. summary analysis) the signatures, sources and destinations of the burst can be obtained.

In the example below the threshold is set to 180, corresponding to 3 events per second. Two event bursts above the threshold can be seen. Notice that the second (and biggest) event burst corresponds exactly to the vertical row of alerts seen in the right hand side of Figure 3 in Section 5.3 above. By using the coordinated coloring across the different views, the signatures, sources and destinations of each burst can be examined. One can then construct a "blacklist" of the sources that generate most of the events in the burst. These sites can then be watched more closely, or one may try gain other information about them.

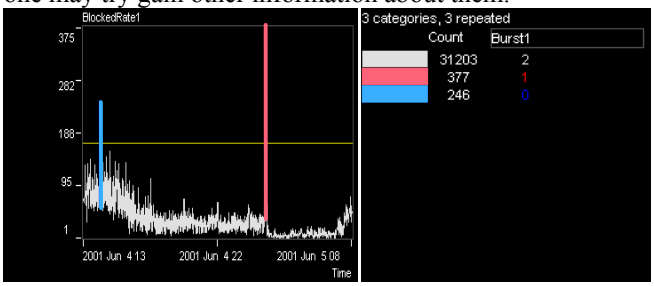

Figure 4: Event-Rate Analysis. The threshold (shown as horizontal line in the left hand chart) is set at 3 events per second (equivalent to 180 "hits" per minute). The second burst (vertical line) corresponds exactly to the vertical row of alerts seen on the right hand side of Figure 3 in Section 5.3 above. These events account for 377 "hits/minute" (see middle bar on chart on the right) and 246 "hits/minute" for the bottom bar (smaller burst). The window size was set to 1 minute. Note: the white color denotes "other" events, i.e., under the threshold.

5.5 Temporal Association

Temporal association attempts to detects pairs of events (alerts) that are highly associated with one another. The user selects a time window and a probability threshold. Let E be an event. The algorithm finds all other events E' such that of all the times that E occurs in the data, E' occurs either before or after with a probability exceeding the user threshold. This is repeated for all events, thus finding all pairs of highly associated events. Note that any pair of events can occur in either order – a probability is reported in each case. For more details concerning the algorithm see [5].

Any of the variables can be used for temporal association. For example, suppose we use signature and that the following pairs of signatures are reported:

SIG	SIG1	Count	P.1.2	P.2.1
35	37	55	0.797	0.833
35	41	45	0.652	0.662
36	44	26	0.650	0.867
37	42	46	0.697	0.780
…				

The first row of this table should be interpreted to mean: "If signature 35 happens than 79.7% of the time signature 37 occurs (within the user-selected time window), and if signature 37 happens then 83% of the time signature 35 is also observed within the window", and similarly with the other rows.

Temporal association can be used to group signatures into clusters under transitive closure. In the above example, signatures {35, 37, 41, 42} form one cluster, because for each one of them there is a high association with at least one other member of the cluster. Similarly {36, 44} forms another cluster. The use of this is illustrated in the application scenario below.

5.6 Stream-based Clustering for Intrusion Detection

An interesting method for intrusion detection is to use stream-based clustering. In this application, we have a stream of data records which are generated by sensor detection of web page accesses from different sources, It is assumed that the data records contain attributes such as

the source URL, the destination URL, and the ports of the suspected attack. In many cases, when an attack occurs, a number of anamolous records appear in the form of new clusters in the data.. The advantage of stream based clustering is that it is possible to find anamolies in real time and diagnose them as possible attacks. This is because when new patterns of behavior are detected in the data, they are often the result of significant changes in the underlying process. We note that stream based clustering is quite different from traditional clustering methods in many ways because of the real time requirements of the process.

The clustering algorithm works by using a process of micro-clustering [2] in a real time fashion. The essential idea in micro-clustering is to exploit a summary representation of the clusters in order to construct the clusters of the data in incremental fashion. In the case of the intrusion detection application, the records $\overline{X}_1$, …, $\overline{X}_k$ … were each assumed to be d dimensional records. We also assume that each record is assocated with a time stamp corresponding to its time of arrival and an incident id. This incident id can simply be considered a unique record For each micro-cluster, the following statistical data was maintained:

- For each dimension, we maintain the list of attribute values which have appeared at least once in the cluster. We also store the frequency of that particular attribute value. Thus, for a given dimension j, we store the list id_1^j … $id_{n_j}^j$, f_1^j, … $f_{n_j}^j$. Here $id_{n_j}^j$ is the identifier of a categorical attribute value for dimension j, and $f_{n_j}^j$ is the corresponding frequency. Let us denote this vector by V(j, C) for the cluster C.

- We store the incident id of the first incident in each cluster. This defines the time when the cluster was born. For the cluster C, we denote this by B(C).

- We store the last time-stamp that a point was added to the micro-cluster. For the cluster C, we denote this by U(C).

- We store the number of points in the micro-cluster. For the cluster C, we denote this by n(C).

Thus, for a d-dimensional database, we maintain d.v + 3 values. Here v is the number of possible values of a categorical attribute.

The actual micro-cluster maintenance algorithm uses a nearest neighbor maintenance procedure. In this procedure, a maximum number of micro-clusters are maintained concurrently by the algorithm. Whenever a new data point arrives, it is either added to an existing micro-cluster, or a new micro-cluster is creating containing solely that data point. In order to decide, which micro-cluster an incoming point should be added to, we use a nearest neighbor maintenance procedure. For each dimension, we find the

similarity value of the centroid of the micro-cluster to the data record. For this purpose, the cosine similarity of the vector V(j, C) to the corresponding vector for the individual record is computed. We find the closest micro-cluster to the incoming data point based on this metric. If this metric is higher than a user-defined threshold, the data point is added to the micro-cluster. When the addition is performed, the centroid V(j, C) and number of points n(C) of the appropriate micro-cluster is updated. Otherwise, a new micro-cluster is created containing only that data point. The birth time B(C) of this newly created micro-cluster is the current time stamp. When a new micro-cluster is added, it is first determined whether there is sufficient buffer space to accommodate it. If this is not the case, then one of the earlier micro-clusters needs to be removed. For this purpose, we use an LRU algorithm, in which the cluster which was least recently updated is removed from the data. This corresponds to the cluster with the smallest value U(C).

Although this process is primarily designed to serve real time requirements, it can also obviously be applied to a batch processing scenario in which the entire data set is available from scratch. This technique has been applied to IDS data which contains attributes such as the source and destination address, port, and signature of the logged event. Each of these attributes belong to a categorical domain. Each cluster consists of a possible pair of the two most dominant of the following attributes: source/destination IPs and Ports, and signatures. The inclusion of possible pairs allows the inference of larger clusters by association on common attributes, hence by induction one can determine "macro" clusters that link via a transitive closure. In the screen shot below of the "Cluster Viewer", the view was limited to a small quantity of the 535 available entries. The y-axis identifies the cluster numbers (newer clusters appear at the bottom), the x-axis identifies the time line (earlier entries appear to the left). The left edge is the B(C), the right edge of any cluster demarks the last U(C). The cursor is used to identify a cluster of interest as shown in the Figure 5. Cluster 581247 is characterized mainly by a single signature directed at a single server's web server port. Also note that the tool tip identified the B(C), and the number of incidents in the cluster. The rectangular areas are color coded to help differentiate from one cluster to the next, otherwise, there is no special meaning to color. Thicker cluster lines denote those with more than 12 events from the sparse entries. (For clarity, the threshold for showing clusters was adjusted to eliminate the thin lines). The bottom-most line denotes the onset of a significant, newly recognized cluster after time gap in time.

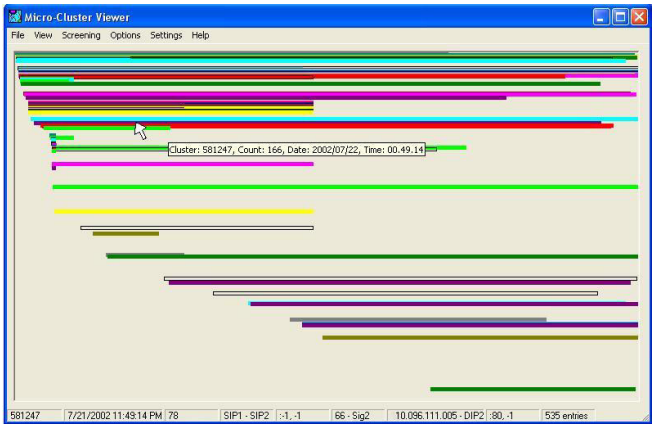

Figure 5. Micro-Cluster Viewer dialog.

Screening and other options make it possible for the analyst to identify the emergence of new patterns that my elude conventional console reporting. The analyst would adjust screening and number of events thresholds to focus on identification of new patterns.

6. Application Scenario

There are two typical application scenarios. The first one is that our customer has a large amount of IDS data, say one month or even more, with many thousands or even millions of events, and we are asked to perform analysis so as to identify installation issues as well as unknown patterns. For this, we often start with the summary analysis by date, by source and by signature. We then run other analysis approaches for high volume alerts. The second scenario is more of an on-going effort. Very often, through an initial data analysis, some types of analysis approaches are found to be valuable. Thus several typical analyses are run routinely. Once some unknown situations are encountered, further analysis can be performed to obtain detailed information by using other analysis techniques.

Clearly, in both scenarios, we need to apply multiple analysis approaches. To illustrate what can be achieved by integrating the various data-mining tools described above, we discuss one particular case in below.

The raw event logs are obtained from a Web hosting environment where thousands of servers support web sites for multiple customers. In a typical day, over one hundred thousands of alerts may be received. In this analysis, we first use the temporal analysis to identify suspicious patterns; we then use summarization to confirm the hosts and signature types in these suspicious patterns.

The following graphic shows the associations of alerts based on signature within a time window of 1 minute with a

probability threshold of 0.60. Recall from Section 5.5 that each row of the table should be interpreted as showing the likelihood of the pair of signatures in either order; thus the first row means that if signature 35 occurs then 79.7% of the time signature 37 occurs within 1 minute, and if signature 37 occurs then 83% of the time signature 35 is also observed within 1 minute.

Unscaled	SIG	SIG1	Count	P.1.2.	P.2.1.	Pm
0	-35	-37	55	0.797	0.833	0.797
1	-35	-41	45	0.652	0.682	0.652
2	-36	-44	26	0.650	0.867	0.650
3	-37	-41	53	0.803	0.803	0.803
4	-37	-42	46	0.697	0.780	0.697
5	-37	-43	46	0.697	0.793	0.697
6	-38	-39	21	0.840	0.913	0.840
7	-38	-44	23	0.920	0.767	0.767
8	-39	-44	19	0.826	0.633	0.633
9	-41	-42	55	0.833	0.932	0.833
10	-41	-43	48	0.727	0.828	0.727
11	-42	-43	50	0.847	0.862	0.847
12	-48	-47	9	1	0.900	0.900
13	-48	-46	8	0.889	1	0.889
14	-48	-49	7	0.778	1	0.778
15	-48	-51	8	0.889	0.889	0.889
16	-47	-46	8	0.800	1	0.800
17	-47	-49	7	0.700	1	0.700
18	-47	-51	9	0.900	1	0.900
19	-46	-50	5	0.625	1	0.625
20	-46	-49	7	0.875	1	0.875
21	-46	-51	7	0.875	0.778	0.778
22	-50	-49	5	1	0.714	0.714
23	-49	-51	6	0.857	0.667	0.667
24	-51	-56	6	0.667	1	0.667
25	-51	-55	6	0.667	1	0.667
26	-56	-55	6	1	1	1

Figure 6. Temporal Association. The pairs of most commonly occurring signatures (in either order) are shown, with associated counts and probabilities. Signatures are grouped together into clusters, where any signature in a cluster is temporally associated with at least one other signature in the cluster (see Figure 7).

Signatures are clustered together using transitive closure – two signatures that appear in any row of the table are placed in the same cluster. Using the coordinated coloring across views, the source addresses associated with the signatures in any cluster can be located. If a particular source address is also found on the “blacklist” (the list of sites responsible for most of the events in a burst uncovered by event-rate analysis, as shown in Section 5.4), then the cluster of signatures can be regarded as a characteristic of hostile activity. If another source address matches the same cluster of signatures, then that address may also be manifesting a similar pattern of hostile behavior, even if it generates very few alerts. Certain low frequency intrusions can be detected in this way.

The next chart (below) shows the clustering (in time) of those signatures that are closely related to each other by the previously shown chart. We are using a different coloring scheme (magenta, yellow and cyan) from before to identify the attack patterns.

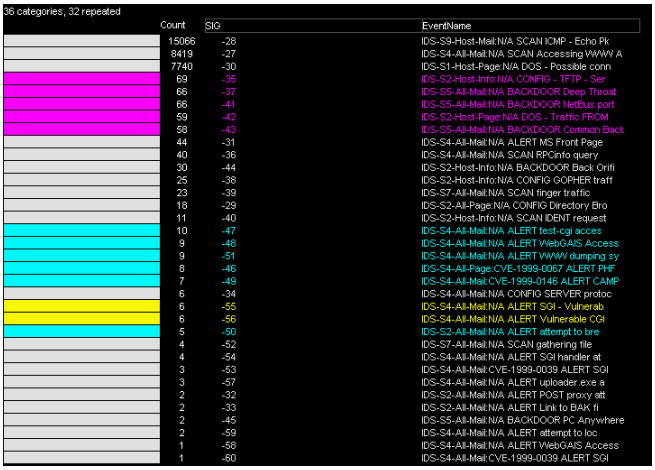

Figure 7: Summary Analysis by signature for the clusters of signatures identified by the temporal analysis of Figure 5. Note that we are using different colors than Figures 3, 5 and 6 to identify attack patterns.

Recall that signatures in these clusters occur within 1 minute of each other. This "tight" association in time most probably shows the use of hacker tools, because such associations would be extremely unlikely to occur manually. This list does not show the host systems that source the intrusions or attacks (there may be multiple hacker hosts). Therefore we extend the analysis by finding the hosts responsible for these attacks, shown in Figure 7 below.

Figure 7 plots the hosts and signature mapping for the signatures in Figure 6. This reveals the "true" list of attackers. Notice again the common color coding of alerts across Figures 8 and 9; e.g the magenta refers to the cluster of signatures {35, 37, 41, 42, 43} shown in Figure 6.

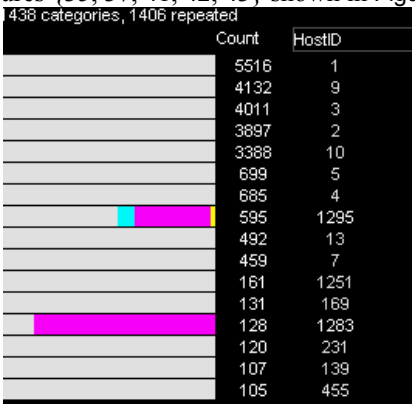

Figure 8: Summary Analysis by Host ID for the signature clusters seen in Figure 7. The color coding is the same – for example, the magenta refers to the magenta signatures in Figure 7.

In this case, two hosts are identified as causing what appear to be carefully orchestrated intrusions, perhaps by running scripts or programs to perform a methodical series of attacks. In this illustration, one source (Host 1295) is known to be "blacklisted"; i.e it was responsible for a large number of the alerts that occurred in the largest event burst shown in Figure 4. The other source is a low frequency attacker that did not contribute significantly to any event burst. It has been detected through an integrated approach that uncovered that its behavior was remarkably similar to that of the blacklisted host.

6. Conclusions and Future Work

We have defined and implemented the infrastructure needed for capturing and enhancing large quantities of network-based alert data and making it readily available for analysis in order to extract valuable information which can then be reported as a guide to further action. Analysis is performed using a variety of data-mining techniques integrated in the Event Browser/Event Miner tools. The power of this methodology is that it enables the user to interactively select the appropriate analysis tool as he/she explores interesting patterns in the data. When applied to real data, this approach has be used to reduce false alarm rates, reveal troublesome source addresses, including those causing low frequency events, and indicate the use of scripting tools used by hackers. We envision future work in both the database and data-mining areas. We expect to extend the current database schema to permit OLAP-like rendering of data to provide the expert user an additional way of gaining insight into undiscovered trends. We also expect to research nonlinear time modeling, and investigate means to exploit mathematical transforms that characterize states, trends, and changes to trends (e.g., predictive filtering technology). We will continue to focus our efforts on e-business hosting logs, and as applicable, we will define security service level agreement specifications in order to provide a secure customer environment in the web hosting realm.

7. References

[1] S. 1lsson, "A preliminary attempt to apply detection and estimation theory to intrusion detection. Technical report", Department of Computer Engineering, Chalmers University of Technology, Goteborg, Sweden, 2000.

[2] C. Aggarwal, J. Han, J. Wang, P. Yu.
"A Framework for Clustering Evolving Data Streams",
VLDB Conference, 2003.

[3] T. Dunigan and G. Hinkel, "Intrusion Detection and Intrusion Prevention on a large network, a case study", Usenix 1999.

[4] S. Ma, J.L. Hellerstein and C. Perng, "EventMiner: An integrated mining tool for scalable analysis of event data", Visual data mining workshop, 2001.

[5] S. Ma and J.L. Hellerstein, "Mining Event Data for Actionable Patterns", Computer Measurement Group, 2000.

[6] D.J. Taylor, N.Halim, J.L Hellerstein, and S. Ma, "Scalable Visualization of Event Data", Distributed Systems Operations and Management, 2000.

[7] A. Valdes and K. Skinner, "Probabilistic Alert Correlation", RAID 2001.

[8] W. Fan et al, "Using Artificial Anomalies to Detect Unknown and Known Network Intrusions", ICDM 2001.

[9] J. Green, D. Marchette, and S. Northcutt, "Analysis Techniques for Detecting Coordinated Attacks and Probes", Proceedings of the 8th Usenix Security Symposium, 1999.

[10] K. Julisch, "Mining Alarm Clusters to Improve Alarm Handling Efficiency", 17th Annual Computer Security Applications Conf., 2001.

[11] W. Lee and S. Stolfo, "Data Mining Approaches for Intrusion Detection", Proceedings of the 7th USENIX Security Symposium, 1998.

[12] W. Lee and D. Xiang. "Information-theoretic measures for anomaly detection", Proceedings of the 2001 IEEE Symposium on Security and Privacy, May 2001.

[13] S. Manganaris et al, *A Data Mining Analysis of RTID Alarms, Recent Advances in Intrusion Detection*, 1999.
[14] IBM International Technical Support Organization, *e-business Risk Management with Tivoli Risk Manager*, September 2001.

[15] Y. Zhang and V. Paxson, "Detecting backdoors", Proceedings of the 9th USENIX Security Symposium, 2000.