A Graph Similarity-based Approach to Security Event Analysis Using Correlation Techniques

Qishi Wu, Yi Gu
Department of Computer Science
University of Memphis
Memphis, TN 38152, USA
Email: {qishiwu,yigu}@memphis.edu

Xiaohui Cui
Applied Software Engineering Research Group
Oak Ridge National Laboratory
Oak Ridge, TN 37831, USA
Email: cuix@ornl.gov

Praneeth Moka, Yunyue Lin
Department of Computer Science
University of Memphis
Memphis, TN 38152, USA
Email: {pmoka,ylin1}@memphis.edu

***Abstract*—Detecting and identifying security events to provide cyber situation awareness has become an increasingly important task within the network research and development community. We propose a graph similarity-based approach to event detection and identification that integrates a number of techniques to collect time-varying situation information, extract correlations between event attributes, and characterize and identify security events. Diverging from the traditional rule- or statistical-based pattern matching techniques, the proposed mechanism represents security events in a graphical form of correlation networks and identifies security events through the computation of graph similarity measurements to eliminate the need for constructing user or system profiles. These technical components take fundamentally different approaches from traditional empirical or statistical methods and are designed based on rigorous computational analysis with mathematically proven performance guarantee. The performance superiority of the proposed mechanism is demonstrated by extensive simulation and experimental results.**

Keywords: correlation, random matrix theory, intrusion detection, graph similarity

I. Introduction

The successful execution of network-based applications requires a timely, reliable, and accurate flow of information in cyber space. To ensure the security of data transfer and processing, cyber space must be safeguarded and protected against various types of attacks (both stealth and overwhelming) launched by the adversary or malicious users.

Traditional security mechanisms such as firewalls, antivirus programs, authentication and authority tools, and virtual private networks have been widely deployed to protect computer network systems against various cyber threats. However, effective mechanisms that can translate low-level situation information to high-level human cognition for accurate decision making and prompt action taking are still missing. As a matter of fact, system monitoring and event analysis have become increasingly challenging due to the wide variety, large scope, frequent occurrence, and substantive complexity of rapidly evolving attacks. Some new attacks may cause multi-hit damage to the system using very sophisticated techniques and hence are extremely difficult to be identified using traditional detection mechanisms.

We define a general term of "security event" to denote abnormal behaviors including any type of "intrusion" or "attack" launched by malicious users to compromise the security of a specific host or the entire network system, such as a virus infection/breakout or a distributed denial of service (DDOS) attack. An event detection or identification method falls into one of two categories [1]: (i) rule-based (also referred to as signature-based) approach which uses the "signature" of an attack to identify a potential attack; and (ii) statistical-based (also referred to as behavior-based) approach, which attempts to learn event patterns on particular historical data and then match future events with the known patterns to identify abnormalities. The rule-based method may not identify new attacks since it takes time to get new rules updated, while the statistical-based method must compare activities to the stored patterns that model known attacks, unacceptable states, proper configurations, or system security policies.

We propose a graph similarity-based approach to security event analysis using inter-attribute correlations. This approach integrates a number of techniques to collect time-varying situation information, extract correlation between event attributes or indicators, construct correlation networks based on Random Matrix Theory (RMT), and characterize and identify security events using graph similarity measurements. The correlations among a set of carefully selected event attributes are explored to capture the patterns of different events. Diverging from the traditional rule- or statistical-based pattern matching techniques, security events in the proposed mechanism are represented in a graphical form of correlation networks and identified through the computation of graph similarity measurements to eliminate the need for constructing user or system profiles, which often involve subjective human judgement and interpretation. These technical components take fundamentally different approaches from traditional empirical or statistical methods and are designed based on rigorous computational analysis with mathematically proven performance guarantee. For performance evaluation, in addition to simulations, we set up a local network testbed where we can launch, monitor and identify various types of security events. Extensive experimental results collected from this testbed justify the efficacy of the proposed technical approaches.

The rest of the paper is organized as follows. Section II describes the related work. The proposed event analysis approach is presented in Section III. The simulation and experimental results are provided in Section IV. Section V concludes our work.

II. Related Work

It has been the primary interest in cyber security to provide automated capabilities of detecting intrusions or other abnormalities in computer systems, reporting them in useful ways, removing discovered anomalies, and repairing damage they may have caused [2].

Security data usually falls into two categories: (i) time-series data such as firewall or system log files and (ii) static data such as user and equipment information about the environment [3], both of which can be spawned from various points in network segments and host systems. The design and deployment of sensors at both network and host locations for security data collection have been extensively studied in the literature [4]–[6]. In practice, various utilities and tools can be used to gather security situation information, spanning from operating system-level commands such as vmstat, ps-ef, etc. to network-level commercial agent software such as snort and wireshark. The situation information collected at different levels serves as the basis for security event detection.

The technology of security event monitoring and detection is based on observation, experience, and classification of attacks, vulnerabilities, and countermeasures [7]. Data mining is one of the most widely used approaches in the literature for event data analysis [8]–[10]. Thuraisingham provided an overview of data mining techniques and cyber threats, and discussed several developments in applying data mining for cyber security analysis [8]. Chandola *et al.*provided an overview of the Minnesota Intrusion Detection System (MINDS), which uses a set of data mining based algorithms to address different aspects of cyber security [9]. In [10], Singhal *et al.*discussed data mining and data warehousing techniques to improve the performance and usability of Intrusion Detection Systems (IDSs), which can support historical data analysis and data summarization.

Correlations, often measured as correlation coefficients between different physical or logical entities (random variables), have been the focus of research for decades in many applications. Pearson product-moment correlation coefficient, one of the best known correlation coefficients [11], is a measure of the strength of linear dependence between two variables and is obtained by dividing the covariance of the two variables by the product of their standard deviations. RMT has been successfully applied to the study of behaviors of complex systems including stock market [12], spectra of large atoms [13], metal insulator transitions in disorder systems [14], and spectra of quasiperiodic systems [15]. However, its applicability in cyber security remains largely unexplored. We hypothesize that the universal properties of RMT are also applicable to the sensor data in cyber space and the correlation threshold can be determined by characterizing the correlation matrix of network profiles using RMT. Network characterization and comparison have been studied in various domains, especially biological and bioinformatics systems. Most studies of biological networks compare their connectivity properties to theoretical or other types of well-studied graphical systems [16], [17]. There exist a number of approaches to the comparison of biological networks with focus on either the general topological statistics of subgraphs [18] or the statistical prevalence of different types of node connection patterns [19]. The network comparison procedure in [20] is based on the shared-edge ratio.

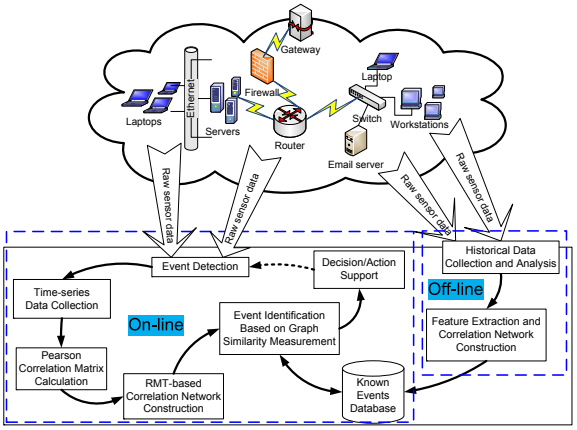

Fig. 1. Framework of the integrated event detection approach.

III. Technical Details of the Proposed Mechanism

We propose a dynamic computational approach to event detection and identification for cyber security using RMT-based correlation network construction and similarity-based graph comparison. The overall framework of the proposed mechanism is illustrated in Fig. 1.

A. Offline Event Database Construction

We first analyze the historical cyber data of security events collected by local sensors that are distributed in both networks and systems. Each event is associated with several attributes or indicators which can be used to describe and measure the corresponding event. For example, a virus breakout typically causes an abrupt increase in the use of CPU cycles or disk space and the number of opened files on the affected machines over a certain period, while in a DDOS attack, one typically sees high bandwidth usage on the gateway router and a large number of connections originated from different locations but targeting the same victim machine without legitimate data exchange.

We use those collected measurements to construct a database of known events, on which we have a complete knowledge *in priori*. Such events could be launched and monitored in a controlled network environment with a specified start and end time. Local sensors are deployed to collect the measurements of different attributes or indicators for those events. Considering the randomness and noise in data collection, we deploy redundant sensors for each attribute and collect multiple measurements in different situations or time periods. We design the following mathematical techniques to process and analyze the event attribute measurements. A similar procedure is applied to online event identification and identification before performing graph similarity comparison.

1) Correlation Network: We construct a correlation network for each event based on Pearson's correlation coefficients, which calculate the correlations between all pairs of attributes by transforming the time-series event attribute measurements into a correlation matrix with each element calculated as:

$$\rho_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2} \cdot \sqrt{n\sum y_i^2 - (\sum y_i^2)}}, \quad (1)$$

where n is the total number of time steps recorded for each attribute, and $\sum x_i$ or $\sum y_i$ are the time-series measurements (vectors) of different attributes for each event. The correlation matrix captures the relationship between each pair of event attributes under the current cyber situation from the event starting time to the latest time step. Since a security event is constantly evolving, the number of time steps sampled so far may not be sufficient to cover the entire period of the event, resulting in incomplete measurement data. Furthermore, the measurement data are generally imperfect due to the inappropriateness of event attributes selection, inaccurate measurements, and delay effects. Therefore, the correlation matrix contains noise or random components that must be filtered out to reflect the true relationship between each pair of event attributes.

2) Denoising: It is generally hard to determine an appropriate threshold (cutoff value) to distinguish the true correlation from the random noise in the obtained correlation matrix due to the lack of comprehensive and accurate system knowledge and control. We apply a similar RMT-based procedure in [21] to our event analysis. Given a Pearson's correlation matrix, we construct a series of new correlation matrices using different cutoff values. For each cutoff value, we set the element value in the original correlation matrix, who has an absolute value less than the selected cutoff, to be 0 in the new matrix, and keep the remain elements unchanged in the new matrices. We calculate the eigenvalues of each correlation matrix using direct diagonalization of the matrix. Standard spectral unfolding techniques are applied to have a constant density of eigenvalues and subsequently the nearest neighbor spacing distribution, which is employed to describe the fluctuation of the eigenvalues of the correlation matrix. We use χ^2 test to determine two critical threshold values that define the transition range from Gaussian Orthogonal Ensembles (GOE) to Poisson distribution at a certain confidence level, and the value at which the reference point starts to follow Poisson distribution is selected as the threshold or cutoff value.

Once the threshold is determined, a correlation network is constructed from the original correlation matrix by keeping those correlation coefficients higher than the threshold and eliminating all others below the threshold. Such a correlation network is further converted to the graphical representation of a security event under the current cyber situation. The graphical representations of security events are stored in the database for comparison with future detected events. This process might be done off-line *a priori*, while time-series attribute measurements for new events should be collected by different sensors in real networks in real-time.

B. Online Event Detection and Identification

1) Event Detection: A fundamental issue in most event detection systems is to determine what value(s) or metric(s) can be used to subjectively decide whether it is a normal or abnormal behavior. While situation information is being continuously collected by sensors at different network and system locations, most event detection components use certain preset thresholds to assess the health of cyber space or discern the legitimacy (normal or abnormal) of the current activity, which inevitably triggers false positive or false negative (i.e. false alarms or miss true cases).

We use a hard fusion algorithm with analytically proven performance guarantee to make a prompt and reliable decision on the occurrence of an intrusion from a global perspective based on local votes casted by individual sensors [22]. Each sensor makes a local threshold-based binary decision on the occurrence of a security event and sends its decision together with the raw event attribute measurements to a frontend data center. The final global decision is reached by integrating the local binary decisions made by multiple sensors that are monitoring the same event from different aspects.

2) Event Identification: Many security systems identify attacks by analyzing network traffic flow and looking for known signatures. The main drawback of such an approach is the development and implementation of signatures [1]. Moreover, the type, signature, and effect of cyber threats are continuously changing over time. Rule-based identification cannot detect new attacks whose signatures have not been implemented in the system beforehand.

We apply a similar procedure as described in Section III-A to construct a correlation network for each detected suspect event. When a new security event is detected, the correlation engine is invoked to construct an event attribute correlation matrix from time-series raw situation measurements collected by sensors up to the current time step, which is further converted to a correlation network of event attributes using the RMT technique. The graphical representation of the current security event is then compared to those of known security events stored in the database to identify the type of the current event based on graph similarity measured by a graph matching technique, defined as:

$$s(c,k) = \frac{1}{3}\left(\frac{\sum_{i,j\in A_{comm}}\left(|\rho^c_{i,j}\cdot(w_i+w_j)-\rho^k_{i,j}\cdot(w_i+w_j)|\right)}{n_1} + \left|\frac{\sum_{i\in A_{comm},j\in A_{curr}}\left(|\rho^c_{i,j}\cdot(w_i-w_j)|\right)}{n_2} - \frac{\sum_{i\in A_{comm},j\in A_{known}}\left(|\rho^k_{i,j}\cdot(w_i-w_j)|\right)}{n_3}\right| + \left|\frac{\sum_{i,j\in A_{curr}}\left(\rho^c_{i,j}\cdot(w_i+w_j)\right)}{n_4} - \frac{\sum_{i,j\in A_{known}}\left(\rho^k_{i,j}\cdot(w_i+w_j)\right)}{n_5}\right|\right), \quad (2)$$

where c and k represents the current and known events, respectively; A_{comm}, A_{curr}, and A_{known} represent the set of common attributes shared by both events, the set of attributes

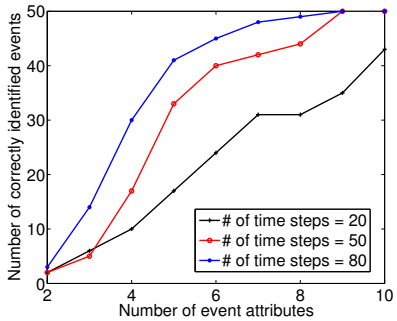

Fig. 2. Event identification performance vs. the number of event attributes in data collection.

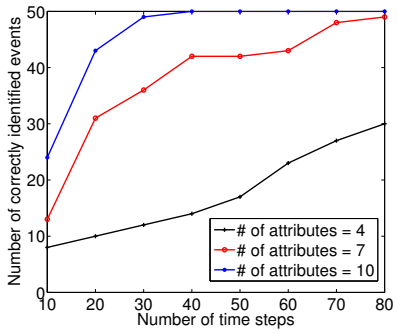

Fig. 3. Event identification performance vs. the number of time steps in data collection.

that only belong to the current event, and the set of attributes that only belong to the known event, respectively; w represents the weight of an attribute; and n represents the number of edges within an attribute set or across two attribute sets. A high value of s indicates a low similarity between the current and known events.

For the accuracy and robustness purposes, we collect multiple sets of measurements for each controlled event whose pattern (correlation network) is stored in the known event database for future comparison with detected events, while for online event identification, only one set of measurements are collected in real-time for the current event.

IV. Simulation and Experimental Results

A. Simulation Results

We conduct two sets of simulations to study the effects of the number of event attributes and the number of time steps on the event identification performance, respectively. These performance measurements provide us with valuable insight into how the raw data collection process should be conducted and how the system would respond at various time points with different numbers of event attributes. Fig. 2 shows that collecting more time steps of attribute measurements results in a better performance of event identification, and Fig. 3 indicates that the more attributes of an event are monitored, the better performance of event identification the system produces.

B. Experimental Settings

We further conduct experiments using two real systems. The first system is the victim computer with Windows XP Professional operating system with no updates installed, equipped with Intel Pentium 4, 2.0 GHz processor, and 1.25 GB RAM. This computer was used as the receiver end, to which the attacks are targeted. The second system is the source (attacker) computer with Windows 7 Home premium operating system (64-bit), equipped with Intel Core 2 Duo, 2.66 GHz processor, and 4 GB RAM. This computer is used as the source for generating attacks. To avoid the spreading of an attack to other systems in the network, we use an isolated local network to connect these two computers.

These two computers are connected in a network using Ethernet cables and an Ethernet switch. After the computers are set up, various attacks from the source computer are launched to attack the victim computer using the penetrating testing tool METASPLOIT V3.3. Gaining access to an unauthorized computer can be a difficult task. However, there are many simple techniques like phishing attacks and authentication attacks to accomplish this task. One of the most dangerous and effective methods now in use is Software Exploitation attack, which uses the vulnerabilities in a software to launch a malicious code, also called a payload. Payload is the code that is executed once vulnerability is triggered. There are different types of payloads such as uploading/downloading a file and executing it, adding user accounts, executing commands, and so on. METASPLOIT provides information about various vulnerabilities in an operating system or an application and allows us to execute various payloads on the victim computer[1].

To monitor the effects on the victim computer, we use Performance Monitor, an built-in tool provided by Windows Operating System. Performance Monitor shows how the attributes related to various performance objects (process, processor, cache, physical disk, memory, etc.) change as the time progresses. Each performance object represents a component of the victim computer and it has a set of various attributes. This tool allows us to generate log data for all the attributes of each performance object for each time step, which in our case is set to be one second. We monitored the performance attributes of the victim computer each time when an attack is launched.

C. Experimental Results

We monitor the total 109 attributes for each real attack and record 120 time steps of measurements. The performance parameters related to processors include: (i) C1 Time, i.e. the percentage of the time the processor spends in C1 low-power idle state; (ii) C2 Time, i.e. the percentage of time the processor spends in C2 low-power idle state; (iii) Interrupt Time, i.e. the percentage of time the processor spends handling hardware interrupts; and (iv) Processor Time, i.e. the percentage of elapsed time that the processor spends executing a non-idle thread. The performance parameters related to cache are as follows: (i) Async Copy Reads per sec, which gives the frequency of reads from pages of file system cache involving

[1]This tool can be downloaded from the link http://www.metasploit.com/framework/download/.

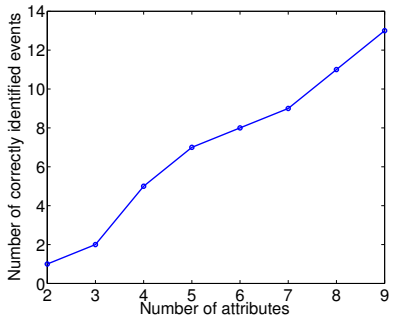

Fig. 4. The number of correctly identified events in response to the number of attributes using 120 time steps of measurements.

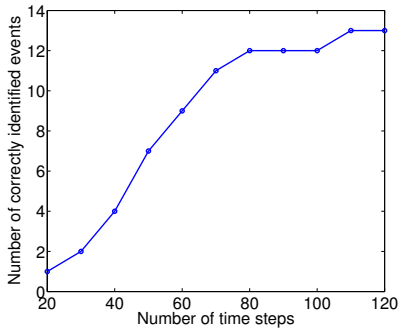

Fig. 5. The number of correctly identified events in response to the number of time steps using all useful attributes.

copy of data from cache to application's buffer; (ii) Async Data Maps per sec, which gives the frequency of an application using file system to map each page of a file into the file system cache for reading the page; and (iii) Fast Reads per sec, which gives the frequency of reads from the file system cache that bypasses the installed file system and retrieves the data directly from the system cache.

We test 15 events in total in this experiment. For each event, we filter the noisy attributes and only keep the useful ones. We plot the event identification performance of the proposed system in Figs. 4 and 5. In Fig. 4, we plot the number of correctly identified events in response to the number of attributes using 120 time steps of measurements, while in Fig. 5, we plot the number of correctly identified events in response to the number of time steps using all useful attributes. We observe that the event identification performance increases as more time steps or more attributes are used in sensor data collection, which is consistent with the observation in the previous simulation results.

V. Conclusion

We investigated a dynamic computational approach to data analysis and event detection for cyber security, which integrates a number of component techniques including event detection, correlation computation of event attributes, network representation of security events, and event identification based on graph matching and network similarity measurements. Both the simulation and experimental results demonstrated the superiority of the proposed approach.

It is of our future interest to investigate new methods to accurately represent the correlations among event attributes and measure the similarities between correlation networks. We will also make extensive performance comparisons with traditional rule-based pattern matching techniques used in existing intrusion detection and identification systems and tools.

References

[1] J. Farshchi, "Statistical-based intrusion detection," http://www.securityfocus.com/infocus/1686, Apr. 2003.

[2] L. LaPadula, "State of the art in cybersecurity monitoring," Center for Integrated Intelligence Systems, Bedford, Massachusetts, Report MP 99B0000020R1, Jul. 1999.

[3] M. Raffael, *Applied Security Visualization*. Addison-Wesley Professional, 2008.

[4] J. Balasubramaniyan, J. Fernandez, D. Isacoff, E. Spafford, and D. Zamboni, "An architecture for intrusion detection using autonomous agents," COAST Laboratory, Technical report COAST TR 98-05, 1998.

[5] S. Janakiraman and V. Vasudevan, "Intrusion detection using autonomous agents," *Int. Journal of Soft Computing*, vol. 3, no. 6, pp. 403–406, 2008.

[6] D. Dasgupta and F. Gonzalez, "An intelligent decision support system for intrusion detection and response," in *MMM-ACNS'01: Proc. of the Int. Workshop on Information Assurance in Computer Networks*. London, UK: Springer-Verlag, 2001, pp. 1–14.

[7] L. Lapadula, "State of the art in cybersecurity monitoring," 1999, http://www.mitre.org.

[8] B. Thuraisingham, "Data mining and cyber security," in *Proc. of the 3rd International Conference on Quality Software*, Nov. 2003, p. 2.

[9] V. Chandola, E. Eilertson, L. Ertoz, G. Simon, and V. Kumar, *Data Warehousing and Data Mining Techniques for Computer Security*. Springer, 2006, ch. Data mining for cyber security.

[10] A. Singhal and S. Jajodia, "Data warehousing and data mining techniques for intrusion detection systems," *J. of Distributed and Parallel Databases*, vol. 20, no. 2, pp. 149–166, 2006.

[11] J. Rodgers and W. Nicewander, "Thirteen ways to look at the correlation coefficient," *The American Statistician*, vol. 42, pp. 59–66, 1988.

[12] M. Egi, T. Matsushita, S. Futatsugi, and K. Murakami, *Practical Fruits of Econophysics*. Springer Tokyo, 2006, ch. Random matrix theory applied to portfolio optimization in japanese stock market, pp. 286–290.

[13] M. Mehta, *Random Matrices (2nd edition)*. Academic Press, 1990.

[14] H. Hasegawa and Y. Sakamoto, "Random matrix theories and metal-insulator transition," *Journal of Prog Theor Phys Suppl*, no. 139, pp. 112–127, 2000.

[15] J. Zhong, U. Grimm, R. Romer, and M. Schreiber, "Level-spacing distribution of planar quasiperiodic tight-bindind models," *Physical Review Letter*, vol. 80, pp. 3996–3999, 1998.

[16] A. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, p. 509512, 1999.

[17] R. Albert, H. Jeong, and A. Barabasi, "Error and attack tolerance of complex networks," *Nature*, vol. 406, p. 378382, 2000.

[18] H. Yu, X. Zhu, D. Greenbaum, J. Karro, and M. Gerstein, "Topnet: a tool for comparing biological sub-networks, correlating protein properties with topological statistics," *Nucl. Acids Res*, vol. 32, p. 328337, 2004.

[19] R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, and I. Ayzenshtat, "Superfamilies of evolved and designed networks," *Science*, vol. 303, p. 15381542, 2004.

[20] Pairing subgraphs Using Network Environment Equivalence. http://www.sbg.bio.ic.ac.uk/ phunkee/.

[21] F. Luo, Y. Yang, J. Zhong, H. Gao, L. Khan, D. Thompson, and J. Zhou, "Constructing gene co-expression networks and predicting functions of unknown genes by random matrix theory," *BMC Bioinformatics*, vol. 8, pp. 299+, August 2007. [Online]. Available: http://dx.doi.org/10.1186/1471-2105-8-299

[22] M. Zhu, R. Brooks, Q. Wu, N. Rao, S. Ding, and S. Iyengar, "Fusion of threshold rules for target detection in self-organizing sensor networks," in *Proc. of the 9th ONR/GTRI Workshop on Target Tracking and Sensor Fusion*, Gatlinburg, TN, June 22-23 2006.