Global Information Assurance Certification Paper

John Reuning
8 January 2004

Applying Information Retrieval Techniques to Event Log Analysis for Intrusion Detection

Abstract

This paper explores the application of probabilistic information retrieval theories to the field of log analysis and host-based intrusion detection. Strong similarities exist between intrusion detection and information retrieval. Using information retrieval techniques may yield significant improvements to the performance of intrusion detection systems. This paper provides a brief review of current relevant research in intrusion detection and log analysis, introduces information retrieval methods appropriate for intrusion detection, and proposes a framework for an experimental log analysis system. The proposed system is based on Bayesian probability theory and uses a term frequency-inverse document frequency (TF-IDF) measure to identify anomalies.

Introduction

In the field of information security, intrusion detection refers to the process of identifying unauthorized access to computer systems or electronic data. Several methods exist for detecting intrusions: manual inspection of a system, audit log processing, event log analysis, file integrity checking, host-based intrusion detection, and network intrusion detection. Each method has advantages and pitfalls, either in the amount of human attention required to set up and maintain the system, the accuracy of incident detection, or failure of the system to exclude false positives.

Intrusion detection systems differ on two axes: 1) they can be either network-based or host-based and 2) they can use rulesets or anomaly detection to identify events. A network intrusion detection system is either a dedicated server or network device that examines raw network traffic. A host-based intrusion detection system can run on a server or workstation to watch tasks or files for signs of unauthorized activity. For intrusion identification, systems use rulesets, patterns for matching signatures of known problems; anomaly detection, establishing a baseline for normal activity and flagging deviance; or a combination of these two methods.

Log analysis usually refers to report generation for human inspection but is sometimes used by host-based intrusion detection systems as a data source for identifying problems. System or event logs can be generated by almost any

active software running on a computer and in Unix/Linux are frequently managed by an application named syslog. Log message output reflects the state of the operating system or application at a given time. Messages vary in their level of severity, depending on the software or application design. For example, syslog provides 8 categories ranging from “emerg” to “debug” for differing levels of event severity. Audit logs differ somewhat from event logs. Audit logs usually consist of transactions or process call traces instead of application error or informational messages.

Much research is being conducted on intrusion detection systems, especially ones that are network-based. However, little attention is directed toward event log analysis. Commercial and open source products exist that match known patterns in log messages and alert a system administrator when a match occurs. This method works well, but it requires human attention and expertise.

This paper suggests that applying a probabilistic information retrieval model (TF-IDF term weighting and relevance judgments) to host-based event log analysis could be an effective means of identifying security incidents and system failures. If a system can be developed to accurately highlight problems without regular pattern updates or specialist intervention, large and small IT organizations could save significant system administrator and security analyst resources.

Current Research

Anomaly-based intrusion detection

Researchers agree on the basic goal of an intrusion detection system. Wagner and Soto state it thus: “The goal of an intrusion detection system (IDS) is like that of a watchful burglar alarm: if an attacker manages to penetrate somehow our security perimeter, the IDS should set off alarms so that a system administrator may take appropriate action” (255). The intrusion detection system assists information security professionals in maintaining data integrity.

Intrusion detection systems have traditionally been based on matching patterns in raw network traffic. More recently, however, activity has centered around profiling activity of a system or network and detecting anomalies. Wagner and Soto are critical of traditional, pattern-matching intrusion detection systems. They state that “[s]ignature-based schemes are typically trivial to bypass simply by varying the attack slightly, much in the same way that polymorphic viruses evade virus checkers” (Wagner and Soto 255). They contend that anomaly detection is more resistant to the evasion tactics of attackers (Wagner and Soto 256).

Anomaly detecting intrusion detection systems are not without their own failings.

> The current model-based approaches all share one common problem: a truly robust intrusion detection system must solve a special case of the machine learning problem, a classic AI problem. That is, to prevent false alarms, the IDS must be able to infer, from statistical data, whether the current execution of the system is valid or not. The false alarm rate of present systems is a major problem in practice. (Wagner and Dean 1)

System administrators and security analysts want an IDS that requires little human maintenance, is accurate in identifying problems, and has a low rate of false positive generation.

Implementation techniques

Researchers have used various machine learning and data-mining techniques for baselining normal system activity and identifying abnormalities. Ye et al. designed and tested a system that used Bayesian networks to identify anomalies in Unix audit log data. The system established profiles of normal activity and identified data that deviated from this norm. The study concluded that “Bayesian network has a promising performance in intrusion detection” (Ye et al. 178). While detailed information was provided on the Bayesian network design, the conclusions were quite vague. Little data was given on which types of intrusions were correctly or falsely identified.

Wagner and Soto examined a system that kept track of series of system calls. The system operated as a finite-state automaton, and like most host-based IDS's, it "learn[ed] the normal behavior of applications and recognize[d] possible attacks by looking for abnormalities" (Wagner and Soto 256). A study by Sequeira and Zaki applied clustering algorithms to create an intrusion detection system. Their system achieved an 80% accuracy rate for detecting intrusions with a 15% false positive generation rate (Sequeira and Zaki 395).

The host-based intrusion detection systems in the Ye et al., Wagner and Soto, and Wagner and Dean studies use process call trace audit logs instead of event log messages. However, the general system designs are widely applicable. While the Bayesian network method in the Ye et al. system is similar to the method proposed in this paper, no research exists on a log analysis tool or intrusion detection system using TF-IDF weights to identify anomalies.

Log analysis

Unlike anomaly-based intrusion detection, the topic of log analysis has received little research attention. Commercial and open source tools are used for generating log file reports (see http://www.loganalysis.org), but these reports must still be examined by a specialist. In addition, many log analysis tools report at regular intervals, such as daily or hourly, not in real time as do intrusion detection systems. Muscat, however, outlines a framework for building an intrusion prevention system based on patterns generated from processing event log data. He suggests that examining system log entries in conjunction with a finite state machine will yield effective patterns for a network-based intrusion detection system (Muscat 78). The usefulness of log message data for intrusion detection is acknowledged, but specific research is needed in this area.

Two log analysis tools exist that differ from the traditional pattern matching approach, SL2 and SIDS. SL2 presents itself as an anomaly detecting system. The documentation describes it thus: "This script will scan the directory where your logfiles reside and report everything that it finds with the exception of the those expressions found in the ignore file, scanlog.ignore" (Fulton and Hoffman). While SL2 detects anomalies, it still uses predefined patterns to flag messages. This requires an expert to configure the system with patterns. SIDS, however, stands for Statistics-based Intrusion Detection System and goes beyond pattern matching. Its primary target is web transaction logs, and it uses a simple form of thresholds to identify anomalies. According to a presentation by the system's designer, though, it is prone to resource overutilization and a large number of false positives (Russell).

Closely related to log analysis is the application of information retrieval techniques to computer source code. Ugurel et al. experimented with automatic classification of source code archive documents using support vector machines.

Their system tried to assign source code files to application categories and achieved an accuracy level as high as 86% with a relatively low frequency of false positives (Ugurel et al. 637). They included programmer comments in their tests, which are similar to the text found in log message output (Ugurel et al. 635). Both are created by software developers and tend to be concise and technically specific. Among the conclusions of the article is that term frequency might have improved their results (Ugurel et al. 638). The system proposed in this paper examines event log messages as documents, much as Ugurel et al. do with source code.

Information Retrieval Meets Intrusion Detection

Intrusion detection aims to identify attempts, either successful or unsuccessful, to attain unauthorized access to a system. To be effective, the IDS needs to provide a high rate of successful identification with a low rate of false positives. If the IDS fails to identify intrusion attempts, it does not complete its task. Additionally, if the IDS identifies too many normal events as malicious, system administrators and security analysts waste time investigating innocuous leads.

Information (or document) retrieval systems are quite similar to intrusion detection systems. A typical information retrieval system takes a user query and matches that with documents, objects, etc. within the system. The basic goal is to produce what the user is trying to find. Consider the example of a Web search engine such as Google. When performing a search, the user types a query into a text box and submits it to the system. The query is generally a few words describing web pages that the user seeks. In response to the query, the search engine returns a ranked list of results. Information retrieval systems primarily use statistical or probabilistic methods to determine the likelihood that a web page matches up with what the user is looking for.

The field of information retrieval employs the concept of relevance and measures the success or performance of a system in terms of recall and precision. Relevance is used to describe a match between what is available and what a user is trying to find. In terms of a Web search engine, this might be finding Web sites on intrusion detection systems when typing "intrusion detection system" into Google. For an IDS, a highly relevant match would be the identification of an intrusion attempt, since that is what the system is looking for. Nonrelevant would describe Google returning www.4greyhounds.org for the abovementioned query or an IDS flagging a legitimate user login on a system.

Performance of a retrieval system includes measures of recall and precision. Recall refers to the percentage of the total number of relevant documents available to the system that is returned for a given query. A high recall value means that the system correctly identifies all of the desired items. Precision is the percentage of how many relevant documents are retrieved based on a given query. A high precision measure means that few nonrelevant documents are identified, or a low rate of false positives is attained. In terms of recall and precision, a good IDS should have a high value in both. That would indicate that the IDS identifies almost all unauthorized activity and produces a low number of false positives.[1]

[1] Spam filters can also be evaluated in terms of precision and recall. A successful spam filter identifies almost all spam email messages (high recall) and blocks very few legitimate messages (high precision). In fact, some spam filters use the same statistical and probabilistic methods described in this paper to identify unwanted email.

Most information retrieval systems use a ranking method for generating results. Google, for example, ranks web pages by how closely it thinks the page document fits (or is relevant to) a user's query. The relevance of a document is determined by how closely the query terms match the terms in the document. One ranking method is based on a TF-IDF (term frequency - inverse document frequency) weight. Each word or term in an information retrieval system's knowledgebase is given a weight based on how many documents contain that term. When generating results, the retrieval system uses these weights to rank the items presented to the user.

This TF-IDF calculation can also be applied to log analysis. Each log message is treated as a separate entity, much like Google treats web pages. In information retrieval terminology, each log message is a document, and the elements of the message are document terms. Instead of matching a search query with a list of web pages, the log analysis tool would index a set of log messages and use the TF-IDF weighting calculation to determine whether a new log message deviates from the norm.

The "IDF" part of TF-IDF is inverse document frequency. The weight value of a term is inversely related to the number of documents that contain the term. Less frequently occurring terms are given a higher weight. This means that commonly occurring log messages have a lower weight than do those with infrequently occurring elements. For log analysis, the result is that log messages with only frequently occurring elements are given a low weight. Messages that do not occur frequently are given a high weight and can easily be flagged as anomalous. Since only a human expert can make a final determination, the weight calculations in this case represent a higher or lower probability that the log message is anomalous.

An advantage of the TF-IDF method of log analysis is that the weight calculations can be done very quickly. A list of terms and corresponding IDF weights can be stored as a hash, which generally yields fast results for searching. The TF, or term frequency, component merely involves counting the number of times each element occurs in an individual log message. Since log messages are typically quite terse, counting elements should take few system resources beyond parsing the message itself.

Proposed Log Analysis System

The previous section described how information retrieval methods can be applied to log analysis and intrusion detection. The following is a framework for implementing an experimental system.

The system bases identification of security incidents and system failures on event log messages that deviate from the norm on a given server or workstation. A two stage process is required for the system to function: training and running classification. A training process is required for the system to establish a norm. System training involves indexing log messages for a given time period (e.g. one or two months). The process is similar to that of a search engine indexing web pages.

The experimental system employs Croft's TF-IDF weight for identifying anomalous events. This method involves keeping a count of the total number of documents or log messages indexed. The IDF weight for each element or term is calculated as log(N/n), where N is the total number of log messages and n is the number of messages that contain the given term (Croft 288).

A hash table similar to an inverted document index is constructed, keeping track of how many times each term occurs in a document. Afterward, an IDF weight is calculated for each term. The result might look like:

failed	5.734
sftp	3.781
sshd	1.278
unknown_user	8.529
user1	1.241
user2	2.003

As indicated in the previous section, less frequently occurring elements receive higher weights. The TF-IDF weighting assumes that terms will reoccur, so random or semi-random strings would require exclusion (e.g. TCP sequence numbers in firewall log messages).

Once the system has established a norm, new log messages can be processed and anomalies flagged. The steps for log message evaluation are as follows:

1. Parse the log message into elements with term frequency calculated. The frequency is the number of occurrences of a given element in the parsed log message.

2. Look up each term in the table and obtain its IDF weight. Unknown terms will be assigned an IDF value of the most uncommon term in the index.[2]

3. Calculate the TF-IDF weight for each term and a total score for the log message. The total score for the message is the sum of the TF-IDF weights of the terms.

Log messages with a total score above a given threshold are considered anomalous and are flagged as indicating a security incident or system failure. Determining the most effective threshold for flagging messages will require experimentation during system development.

In keeping with a multilayered approach to security, this method of log analysis would be best applied either as a component of an IDS or in conjunction with other security tools. For example, this log analysis tool could be combined with a network IDS (or IPS), file integrity checking, or another host-based IDS. The system framework described here is meant for logs such as Unix/Linux syslog. However, the theory could be applied to snort logs (to reduce the high false positive rate) or to firewall logs.

[2] Adding a constant to this weight for unknown terms may provide better system performance. Further research during system development is needed.

Summary

Intrusion detection systems is currently an active field of research in information systems and security. As indicated above, most of the current attention is focused on network-based systems, less being devoted to host-based systems. Existing research has examined several data mining and machine learning techniques in host-based IDS's, yet none has used the TF-IDF ranking approach for classifying log messages.

The specific implementation method outlined in this paper has not been explored. If this approach proves successful, the resulting system will help further the field of information security and improve the quality of automated intrusion detection systems. This paper focuses on Unix and Linux log messages generated through syslog; however, the results should be applicable to other types of event logs on a wide range of operating systems.

References

Croft, W. B. and D. J. Harper. "Using Probabilistic Models of Document Retrieval Without Relevance Information." *Journal of Documentation* 35(4) (1979): 285-295.

Fulton, Russell, and Harry Hoffman. SL2 source code. <http://www.ip-solutions.net/syslog-ng/sl2>

Innella, Paul, and Oba McMillan. "An Introduction to Intrusion Detection Systems." December 6, 2001. <http://www.securityfocus.com/infocus/1520>

Muscat, Andre. "A Log Analysis Based Intrusion Detection System for the Creation of a Specification Based Intrusion Prevention System." *Proceedings of the 2003 University of Malta Computer Science Annual Research Workshop*, July, 2003. <http://www.cs.um.edu.mt/~csaw/Proceedings/LogAnalysisIDS.pdf>

Russell, Ryan. "Statistics-based Intrusion Detection System." <http://www.internettradecraft.com/sids/sids.ppt>

Sequeira, Karlton and Mohammed Zaki. "ADMIT: Anomaly-based Data Mining for Intrusions." *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July, 2002.

Sparck Jones, Karen. "Search Term Relevance Weighting Given Little Relevance Information." *Journal of Documentation* 35(1) (1979): 30-48.

Sparck Jones, Karen and Peter Willett, eds. *Readings in Information Retrieval*. San Francisco: Morgan Kaufmann, 1997.

Ugurel, Secil et al. "What's the Code?: Automatic Classification of Source Code Archives." *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July, 2002.

Wagner, David and Drew Dean. "Intrusion Detection via Static Analysis." *Proceedings of the IEEE Symposium on Security and Privacy*, May 2001. <http://www.cs.berkeley.edu/~daw/papers/ids-oakland01.pdf>

Wagner, David and Paolo Soto. "Intrusion Detection: Mimicry Attacks on Host-based Intrusion Detection Systems." *Proceedings of the 9th ACM conference on Computer and Communications Security*, November, 2002. <http://www.cs.berkeley.edu/~daw/papers/mimicry.pdf>

Ye, Nong et al. "Probabilistic Networks with Undirected Links for Anomaly Detection." *Proceedings of the 2000 IEEE Workshop on Information Assurance and Security*, June, 2000.

Zanero, Stefano and Sergio. M. Savaresi. "Unsupervised Learning Techniques for an Intrusion Detection System." *Proceedings of the ACM Symposium on Applied Computing* (ACM SAC 2004), March, 2004. <http://www.elet.polimi.it/upload/zanero/papers/IDS-SAC.pdf>

http://www.loganalysis.org/

http://www.logwatch.org/

Last Updated: July 27th, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Indianapolis SEC401	Indianapolis, IN	Oct 09, 2017 - Oct 14, 2017	Community SANS