Computers & Security (2004) **23**, 312–319

ELSEVIER

Computers & Security

www.elsevier.com/locate/cose

SAD: web session anomaly detection based on parameter estimation☆

Sanghyun Cho*, **Sungdeok Cha**

Division of Computer Science and AITRC/SPIC/IIRTRC, Department of Electrical Engineering and Computer Science, KAIST, Yusung-gu, Daejeon 305-701, Republic of Korea

Received 1 August 2003; revised 3 December 2003; accepted 9 January 2004

KEYWORDS
Computer security;
Intrusion detection;
Anomaly detection;
Web attacks;
Parameter estimation;
Machine learning

Abstract Web attacks are too numerous in numbers and serious in potential consequences for modern society to tolerate. Unfortunately, current generation signature-based intrusion detection systems (IDS) are inadequate, and security techniques such as firewalls or access control mechanisms do not work well when trying to secure web services. In this paper, we empirically demonstrate that the Bayesian parameter estimation method is effective in analyzing web logs and detecting anomalous sessions. When web attacks were simulated with Whisker software, Snort, a well-known IDS based on misuse detection, caught only slightly more than one third of web attacks. Our technique, session anomaly detection (SAD), on the other hand, detected nearly all such attacks without having to rely on attack signatures at all. SAD works by first developing normal usage profile and comparing the web logs, as they are generated, against the expected frequencies. Our research indicates that SAD has the potential of detecting previously unknown web attacks and that the proposed approach would play a key role in developing an integrated environment to provide secure and reliable web services.

Introduction

World Wide Web has fundamentally changed the way people interact with each other and share information. Electronic commerce, ranging from Internet shopping malls to web-based banking and stock trading, has become an integral component of modern society. Use of the web is not limited to commercial activities. The web is an important medium for an individual, company, or even a country to share and promote information to achieve personal, financial, political or military objectives. Unfortunately, according to the most recent CSI/FBI annual survey on computer security issues and trends (Power, 2002), organizations known to have suffered unauthorized access or misuse of their web sites grew from 23% in 2001 to 38% in 2002.

☆ This work was partially supported by the Korea Science and Engineering Foundation (KOSEF) through the Advanced Information Technology Research Center (AITrc), Software Process Improvement Center (SPIC) and by Internet Intrusion Response Technology Research Center (IIRTRC).
* Corresponding author. Tel. +82-42-869-3575; fax: +82-42-869-3510.
E-mail addresses: shcho@salmosa.kaist.ac.kr (S. Cho), cha@salmosa.kaist.ac.kr (S. Cha).

0167-4048/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2004.01.006

There are many commercial IDS products in use that rely on attack signatures. Misuse detection techniques are considered a mature technology, and there are many commercial products that can examine packets travelling the gigabit network. They are scalable in that when new attacks are found, only the rule database, containing the attack signatures, needs to be updated without having to extend the IDS core engine. However, misuse detection technique is inadequate in properly dealing with threats of web attacks for the following reasons:

- Only the attacks containing known and fixed attack patterns can be detected. When new attacks are initiated or an existing attack modified, misuse detection systems are practically defenseless until attacks are (often manually) analyzed, attack signature developed, and rule database updated. Fig. 1 is an example of a Snort (Roesch, 1999) signature designed to detect attack exploiting phf-CGI vulnerability on the web server. When the urlcontent of the packet includes the string specified in the signature, Snort issues "WEB-CGI phf access" warning to alert system administrators. If an intruder were to change the string stored in the urlcontent field of the packet to '%c1%1cphf.cgi', Snort will fail to generate an alarm unless additional signatures to deal with such possibilities are explicitly added (Patton et al., 2001). There are too many variations an intruder may try, and increased number of signatures will inevitably have a negative impact on the performance of Snort, thereby making near-real-time intrusion detection more difficult (Ptacek and Timothy, 1998).
- While there are lots of attack signatures available in the public domain, majority of them may turn out to be irrelevant to an organization. For example, in the Snort version 1.8.6 rule set, 516 out of 1267 entries are related to web attacks. However, if an organization employs the IIS web server, which is quite popular, all but 88 of them are irrelevant.
- Details of web service architecture are site-dependent, and so are the vulnerabilities. Therefore, attack patterns cannot be generalized. Furthermore, some attacks, especially those designed to cause local buffer overflow or exploit race conditions, do not necessarily have a fixed pattern. Worse yet, web services are intended to be open in nature, and security techniques such as firewalls or access control mechanisms are inadequate as a means of securing web services.
- There are several known techniques to bypass or disable IDS. For example, an intruder may attempt to flood the network with lots of decoy packets which contain an attack signature as substrings in an attempt to force IDS to generate lots of alarms and effectively shut it down. When overloaded, IDS may let some hostile packets pass unexamined.
- Misuse detection systems, which rely on analysis of packet data, are unable to detect encrypted attacks. Decisions as to whether a packet is hostile can only be made at application level.
- Should IDS sensor software run on OS platform that is different from that of the target system, many different types of evasive attacks are known and lots of false alerts may be generated.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80
(msg:"WEB-CGI phf access";flags:
A+; uricontent:"/phf.cgi"; nocase; reference:bugtraq, 629;
reference:arachnids,128; reference:cve, CVE-1999-0067;
classtype:attempted-recon; sid:886; rev:3;)
```

Figure 1 The example of the Snort IDS.

Anomaly detection has long been suggested as a promising approach to detect previously unknown attacks. Code Red and Nimda worms (Mackie et al., 2001), both of which exploited flaws in the web server software implementations, made several queries that were hardly related to each other and rare in sequence when compared to normal usage. Had anomaly detection system been in place, such worms could have been detected earlier and caused less financial damage to victim organizations.

In this paper, we report that Bayesian estimation technique is useful in detecting anomalous web sessions and that the rate of false alarm is low enough to be acceptable in practice. Bayesian technique has long been successfully used in pattern recognition and natural language processing applications. In order to empirically demonstrate effectiveness of the proposed approach, we implemented a prototype and conducted a small-scale experiment. Usage profile was developed, based on web access logs, and frequency of sequence of web page requests was computed. We then simulated web attacks by executing Whisker software to simulate web attacks because "live" data containing genuine, as opposed to simulated or artificially injected, web attacks are unavailable in research community.

The rest of the paper is organized as follows. The next section reviews the related works with emphasis on anomaly detection techniques. The third section describes in detail how our approach,

session anomaly detection (SAD) frameworks, works and which software tools were implemented. The fourth section reports results from the controlled experiment and discusses lessons learned followed by the conclusions in the last section.

Related work

There have been several attempts to detect anomalous and suspicious activities by examining the network packets without having to rely on specific signatures. Research conducted by Mahoney and Chan (2001) and by Krugel et al. (2002) are such examples where anomaly detection is made based on analysis on packet header. Even though these attempts may be unable to decide, with certainty, whether some packets are hostile, they offer the potential of detecting previous unknown attacks including distributed denial of service attacks. Such detection is possible because behavioral patterns of hostile codes, while specifics are unknown and unpredictable, are most likely quite different from that of normal usage.

In Mahoney and Chan (2001), the packet anomaly score is calculated to model normal IP packets. The value of each packet field is converted into 1—4 bytes through hashing or clustering, and the probability of occurrence of each value is computed. The lower the probability and the older the time, the larger anomaly score assigned. They experimented with network traffic, which includes simulated attacks, using the DARPA IDS evaluation testsets (Lippmann et al., 2000). However, their approach is unable to detect attacks on specific application such as SMTP, HTTP, and DNS because they only focus on IP packet header fields, not on payloads.

Krugel et al. (2002) calculated anomaly score using three properties: type of request, length of request and payload distribution. In their approach, if the length of the request is longer than the average length, the probability of it being an attack is considered high. The approach is based on the observation that attack codes often contain a large number of NULL opcode characters, '0x90' in Linux, so as to cause buffer overflow and manipulate process's return address. The Nimda worm is such an example. When a hostile code tries to send a shell code to a vulnerable system, certain characters are repeated many times. Such cases are highly unlikely to occur in typical and non-hostile software development. Krugel et al. implemented a prototype that can process HTTP and DNS traffics and obtained impressive results in detecting DNS attacks. Unfortunately, web attacks are less likely to be detected by this approach because only the length of data and character distribution are analyzed. Attacks scanning specific CGI vulnerabilities, requesting prohibited pages, and manipulating arguments sent to web application are unlikely to be detected.

Registry anomaly detection (RAD) (Apap et al., 2002), similar in concept to the proposed approach, uses Bayesian parameter estimation method on windows registry to detect viruses, worms and trojan horses. It is based on the assumption that hostile processes most likely access different registry keys even if they assume the system process's identity in attempts to avoid detection. Using the Bayesian parameter estimation suggested by Friedman and Singer (1999), RAD examines process names, query types, actual keys being accessed, query outcomes, and result values. The simple Bayesian method generally demonstrates poor performance when data exhibiting sparse multinomial distributions are analyzed (Griffiths). Friedman and Singer, on the other hand, introduced another variable used in the calculation of likelihood of an event. That is, even if an event had never been observed in the training data, its probability is not assigned zero, but a very small value. Revised estimation technique was shown to be effective when the size of the alphabet, possible events, is unbounded. Empirical evaluation of the RAD demonstrated highly accurate detection of viruses and trojan horses in software while keeping the rate of false alarm low.

SAD: session anomaly detection

In this paper, we assume that sequences of web pages requested by users would exhibit similar pattern. For example, those who visit course pages are most likely to start their web surfing at the main page for the course or the instructor's personal homepage. They are then likely to view syllabus and homework pages subsequently. Repeatedly requesting specific URL or CGI and trying to bypass authentication pages, when required, could reasonably be considered as being rare, anomalous and possibly hostile. Examples of the former include web scanner software exploiting vulnerable CGI programs and Code Red worm requesting several pages specifically related to the IIS web server as follows.

```
GET /scripts/root.exe?/c+dir
GET /MSADC/root.exe?/c+dir
GET /c/winnt/system32/cmd.exe?/c+dir
GET /d/winnt/system32/cmd.exe?/c+dir
```

Session anomaly detection (SAD), as shown in Fig. 2, extracts web sessions from web access logs and builds profiles of "normally" requested page sequences (PFDB) based on which likelihood of specific request sequences are computed. Anomaly score is computed, and the sessions exceeding the threshold values are labelled abnormal. Parameters used in the analysis are site-dependent and therefore can be customized.

Extraction of web sessions is not as straightforward as it appears conceptually because session boundaries are poorly defined. The problem becomes even more complex when clients' IPs are dynamically assigned. In order to identify the beginnings and the ends of various web sessions, we decided to include all the page requests made from the same IP for the duration of Δ min. The value can be customized at each site, and we chose 30 min as the default value.

A session consists of sequence of pages requested as recorded in the web access log. When analyzing the web logs, one must carefully differentiate the logical access sequences and the physical access sequences. That is, there is a difference, due to page cache mechanism, between the sequence of pages requested by a web browser and therefore logged in the access log of web server and the sequence of pages a user actually views with a web browser. The former is referred to as the physical access sequence, and the latter is called the logical access sequence. Since some requests, especially those made by forward and back buttons, on the pages in the cache do not result in explicit page request, it appears that anomalous web session detection based on logical access sequence is infeasible. Therefore, we limit our analysis based on the physical access sequence.

Another factor to consider is that web pages can be subject to frequent updates and that sequence of pages visited in a session are most likely to be related in semantics and directly linked. That is, it is highly unlikely for users to type (as opposed to click) the URL of the pages to be visited next as browser input. Furthermore, some pages (e.g., authentication page) must first be visited before other pages (e.g., for fee) may be legally requested. Therefore, it makes sense to take web page topology into consideration when analyzing web sessions for anomaly.

We implemented a software to scan web pages stored in a server and analyze links among them. By storing the site topology as an XML document, as shown in Fig. 3, the same information can be used by other tools such as web usage mining tool (Cooley et al., 1999). Leaf-level pages, such as figures with jpg or gif extensions, are excluded in the topology to reduce the number of nodes to be analyzed. <*node*> and <*edge*> tags represent each unique page and link among them, respectively.

We assume that the web logs from the Apache web server are stored in the common format as shown in Fig. 4. One must note that, in current implementation of most web server software, web access logs are written *after*, not before, the web server responses have been sent to the client. Therefore, there are differences in time, however small they might be in practice, between the time of the attack and data being stored in the log.

Each web session is represented as a sequence of pages requested. See Table 1 for an example. For example, the first session in the table requested three distinct pages, numbered 142, 322, 75, respectively. Should a user request web pages that were excluded in the current web topology database, they are assigned unique ID dynamically. Such strategy is needed because web pages are subject to frequent changes and updates.

Size of a web session, regardless of its definition, is different in length, and there is a need to

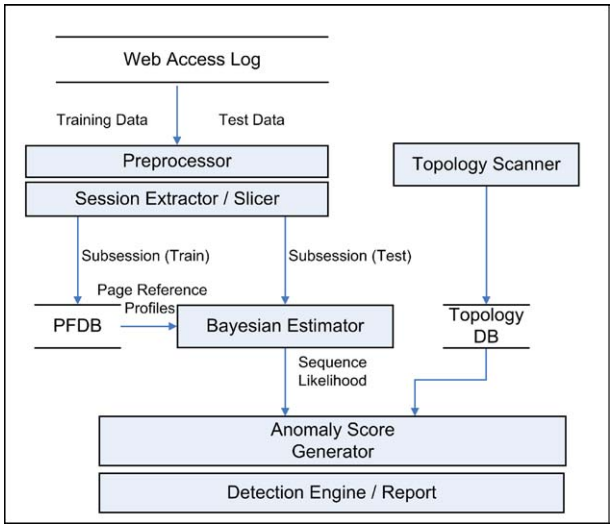

Figure 2 The architecture of session anomaly detection (SAD).

```
<?xml version="1.0"?>
<!-- KAIST Web Scanner(KWS) Ver-0.1 -->
<html xmlns="http://salmosa.kaist.ac.kr/~kane/KWS" xml:lang="en">
<node id="1" label="http://salmosa.kaist.ac.kr/" weight="200"> </node>
<node id="2" label="http://salmosa.kaist.ac.kr/menu.html" weight="200"> </node>
<node id="3" label="http://salmosa.kaist.ac.kr/main.html" weight="200"> </node>
<edge source="1" target="3" weight=0 label=""> </edge>
<edge source="2" target="3" weight=0 label=""> </edge>
                    ...                    ...
```

Figure 3 The example of web topology.

[Web Log Format] (apache)

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referrer}\" \"%{User-Agent}\"" combined

%h : Source IP, %l : Client's identity, %u : User Name in authentication, %t : Time
%r : Request URI, %s : Return status code, %b : Size of transferred data
%{Referrer} : Previous URI, %{User-agent} : Browser information

[Web Log Example]

218.148.212.32 - - [23/Jul/2002:14:48:02 +0900] "GET/~ekjee/images/intro_lbulle t.gif HTTP/1.1" 200 261 "http://salmosa.kaist.ac.kr/~ekjee/intro.htm" "Mozilla/4.0(compatible; MSIE 6.0; Windows NT 5.1; Q312461)

Figure 4 The web log format and example of logs.

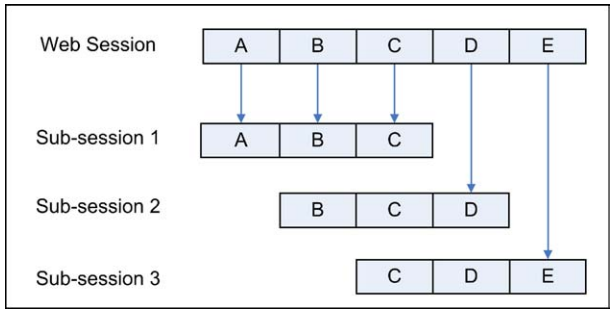

Figure 5 Web sub-session.

apply uniform and consistent analysis variable length data (see Table 1 for an example). We chose to further divide each session into sub-sessions each with fixed length. Fig. 5 illustrates how a web session containing 5 pages is decomposed into three sub-sessions of length 3.

As mentioned earlier, we adopted the Bayesian parameter estimation technique proposed by Friedman and Singer (1999). The likelihood of each event is estimated based on previous probability distribution, and empirical analysis demonstrates Friedman and Singer's estimator to be effective even when processing unbounded alphabets or event sets. Because the number of web pages resident at a site is unbounded and changes dynamically, Bayesian parameter estimation, whose formula is given below, is considered an appropriate choice.

$$C(D,L)=\sum_{k=k^0}^{L}\frac{k^0\alpha+N}{k\alpha+N}P(k|D) \tag{1}$$

$$P(X^{N+1}=i|D)=\begin{cases}\frac{\alpha+N_i}{k^0\alpha+N}C(D,L) & \text{if} \quad i\in\Sigma^0\\ \frac{1}{n-k^0}(1-C(D,L)) & \text{if} \quad i\notin\Sigma^0\end{cases} \tag{2}$$

In Eqs. (1) and (2), $C(D,L)$ denotes the probability that the previous event would happen again and $P(X^{N+1}=i|D)$ is the estimate, based on training data, how likely the same sequence of pages will be requested. For example, in SAD-Bayes-3, with sub-session window size 3, $P(page_\alpha, page_\beta, page_\gamma)$ represents the probability that a user requests pages $page_\alpha$, $page_\beta$ and $page_\gamma$ in sequence. In Friedman and Singer's estimator, whose definition is given in formula (2), N_i is the frequency of a user visiting $page_\gamma$ after requesting $page_\alpha$ and $page_\beta$. N is the frequency of that user visiting $page_\alpha$ and $page_\beta$. L represents the number of total pages visited, and K_O is the number of kinds of event sets, ($page_\alpha$, $page_\beta$) appearing in training data. In the anomaly calculation, the lower the probability is, the higher the anomaly score is assigned and more anomalous the sub-session is considered to be.

Table 1 The example of web session

Length	Pages
3	142 322 75
1	555
7	1 2 3 4 13 14 107
5	1 2 3 8 82
2	13 15
4	96 75553 1 1

When conducting anomaly detection, some degree of false alarm is inevitable. Valid users may choose to use the web, or any other software, in a way different from the established profile, and profile itself must evolve. Therefore, one must evaluate the critical factors contributing to false alarms. To learn more about it, we repeated the experiment with different size of sub-sessions. If the size of sub-session is increased, it is more likely that short web sessions end up unexamined when usage profile is developed, resulting in partial and less accurate profile. In addition, some of the hostile web attacks requiring less page requests than the threshold value may not be detected. Therefore, smaller sub-session size is not necessarily ideal. In addition to overhead associated with profile generation analysis, anomaly score computation would require more computational resource, thereby impacting the feasibility of real-time anomaly detection.

As a web session may consist of multiple sub-sessions, one must also decide how anomaly scores of various sub-sessions are to be combined into the anomaly decision of the entire session. Two obvious options taking the maximum and average scores exist, and there are trade-offs involved. If one were to take the maximum anomaly score of sub-sessions as the session anomaly score, even slight variation from typical usage patterns would trigger an alarm. While providing a possibility that new attacks are more likely to be detected, it would also increase the number of false alarms. If one were to take

the average of the anomaly scores, anomaly detection process can be made less sensitive to errors at the cost that some attacks may go undetected.

Finally, it seems essential to take web site topology into anomaly decision. Just because certain page sequences did not appear in the training data, it makes little sense to always label it as anomalous and hostile if such sequences seem feasible given the relationship among hyperlinks present in the web pages. Likewise, it seems reasonable to adjust the anomaly score if visiting the same and previously observed page sequence would now include laborious activity (e.g., explicit typing of several URLs) on the part of the user given the current topology.

Experimental design and results

Ideally, one must perform an empirical analysis on detection of anomalous web sessions by developing a web usage profile collected over an extended period of time. One then would evaluate effectiveness of the proposed technique by running tests on "live" data containing "genuine" web attacks. Unfortunately, there are no publicly available data on web attacks that can be used as benchmark tests against which effectiveness of various approaches can be quantitatively measured. Occurrence of web attacks can usually be confirmed only after conducting careful and often manual analysis on web traffic, and the exact timings of real-world web attack cannot be predicted in advance. Publicly available evaluation data on IDS is an alternative, but the technique could be "optimized" to provide the best performance. Therefore, an experiment on static data is not objective enough.

As an alternative, we chose to conduct a controlled experiment in which access logs obtained from a university laboratory web server were used as training data. Web logs were collected during a one month period, and data were manually analyzed to eliminate logs generated by well-known attacks such as Nimda worms. The web server used in the case study supports about 40 users most of whom are graduate students. There were 19,643 unique source ip addresses from which 34,154 user sessions were extracted using the criteria mentioned earlier.

During a week of test period that did not overlap the training period, web logs generated from user activities were collected. In addition, for about 3 h, experiment administrators intentionally executed Whisker (Rain Forest Puppy) software version 1.4 as a means of artificially generating anomalous web requests. Whisker has 12 different modes, and about 400 scans are made in each mode including vulnerable CGI program scanning, password guessing, and attempts to evade IDS. Some are dependent on the operating systems and web servers being used. There were 13,415 user sessions initiated from 5247 different source IPs. While test was being carried out, we also monitored the network activities using Snort version 1.8.7 with all six signature groups related to web enabled.

The objectives of the experiment was to empirically evaluate if the proposed approach was capable of detecting the activities of Whisker and the Nimda worm, without having to rely on attack signatures, as being anomalous. We were also interested in determining the expected rate of false alarms and how different parameters (e.g., subsession size, size of profile database, threshold value, session anomaly decision logic, etc) influenced performance.

Results from our experiment are as follows:

Accuracy of anomaly detection. The Bayesian estimation technique detected 91% of all the Whisker scans and other page sequences that were previously unseen as being anomalous. Some Whisker scans, especially in modes 9 and 11, did not leave any logs and could not have been detected by the proposed approach. If we were to exclude such scans, accuracy increases to nearly 99%. Only the Whisker scans requesting '/index.html' page at the root directory were considered normal since such requests were previously observed during the training period. Similar results were obtained with Nimda worms. On the same data, Snort, signature-based network IDS, detected only about 36%, on the average, of the simulated attacks. Our controlled experiment convincingly demonstrates that the proposed approach is effective enough to be used in practice.

False alarms. To measure the impact subsession size had on the rate of false alarms, we repeated the same experiment with different parameter values. When the subsession size of 2, 3, and 4 pages are considered, 14.6%, 13.2%, and 28.9% of sessions were erroneously flagged as being anomalous. Based on such results, we were unable to conclusively determine how an organization must decide on the sub-session size. In the case study, 75% of sessions, according to our session definition, had three web logs or less. With the sub-session size 4, anomaly decision had to rely on partial, therefore inaccurate,

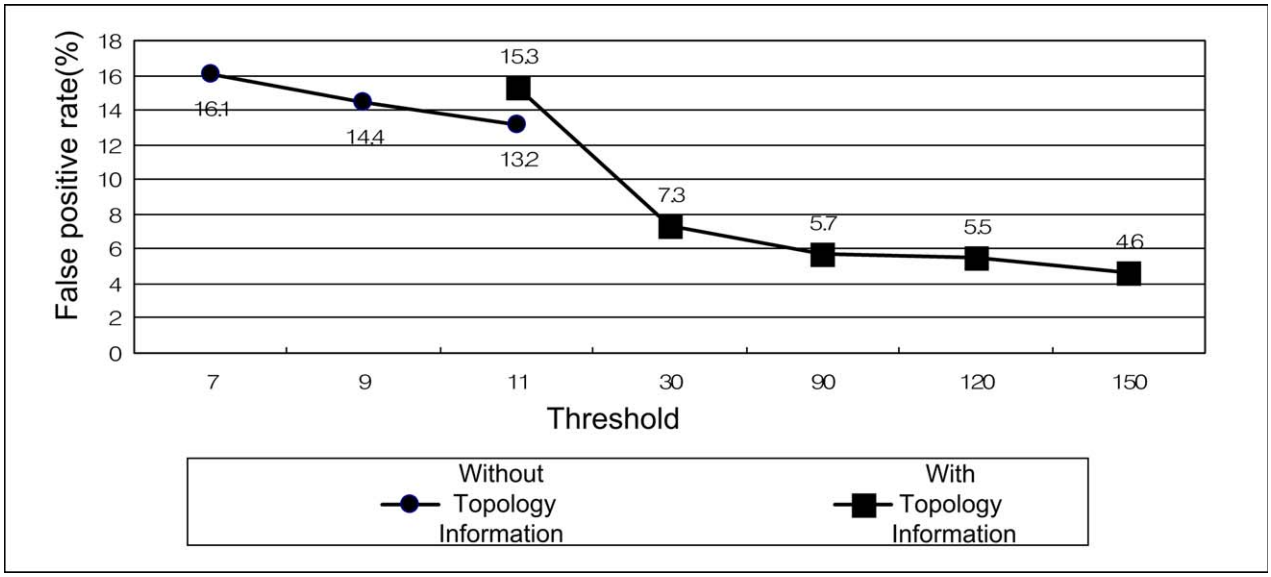

Figure 6 Comparison of the false positive rates of weighted version using topology. The detection rate is about 100%.

profile data. As the size of the sub-session increased, so did the rate of false alarms. As for the optimal value, it is apparently site-dependent, and security administrators need to tune parameters to best fit each site.

Learning time. There was little variation on the learning time in that all the training took about the same time (e.g., 40 min). Therefore, training overhead is unlikely to become a major factor in determining if near-real-time (or real-time) anomaly detection can be achieved because, if needed, training and updating normal usage profile can be conducted on separate and dedicated processors. Rather, it is the number of web logs generated during the test period that determine if real-time or near-real-time anomaly detection is feasible or not. As for the period during which normal usage profile is established, as expected, the longer the period, the more accurate the anomaly detection observed. Although the exact rate is site-dependent, we were able to reduce false alarm by 1% when we doubled the training period from 2 weeks to 4 weeks. That is, the longer the training period, the more accurate the profile that can be derived, and therefore the less the false alarms.

Anomalous session decision logic. As discussed earlier, if a session consists of multiple sub-sessions, one must decide how to compute the overall anomaly score of the session based on the anomaly scores of sub-sessions. For example, one can choose to select the minimum, average, or maximum. In our experiment, maximum method returned higher rate, or 21%, of false alarms than the average method which produced 13.2%.

Web site topology. When anomaly scores were adjusted based on topology structure as shown in Fig. 6, we were able to reduce the rate of false alarm by 4.6% when we incorporated the web topology into the decision logic.

Conclusions

In this paper, we demonstrated that the Bayesian estimation method is effective in determining anomalous web sessions. The proposed approach, despite some degree of false alarms, offers promise that new attacks could be detected without having to know their characteristics in advance. Such a technique would play a key role in further securing web service when used in conjunction with misuse detection systems. Unfortunately, the rate of false alarm is still too high for the technique to be used in an environment where availability of human resource is limited.

There are several topics worthy of further research, and they are as follows:

Handling of dynamic pages. In this work, we only focused on static pages and ignored arguments included when issuing requests. Examples include CGI or php parameters. The more information we utilize in the analysis, the higher accuracy we can reasonably expect. However, brute-force approach in the parameter analysis is unlikely to be effective

since parameter values and file names are highly unpredictable even in "normal" usage of web servers.

Additional features. Only the page sequences and their frequency are analyzed in the proposed approach. If semantic analysis, based on the contents of the web pages, could be performed, one can expect to further enhance accuracy in anomaly detection.

Web session identification. Currently, we used time duration to divide the web logs into sessions. We need to extend the proposed approach to properly deal with dynamic allocation of IP addresses.

References

Apap F, Honig A, Hershkop S, Eskin E, Stolfo S. Detecting malicious software by monitoring anomalous windows registry accesses. In: Proceedings of Recent Advance in Intrusion Detection (RAID 2002); 2002. p. 36—53.

Cooley Robert, Mobasher Bamshad, Srivastava Jaideep. Data preparation for mining world wide web browsing patterns. Knowledge Inf Syst 1999;1(1):5—32.

Friedman N, Singer Y. Efficient Bayesian parameter estimation in large discrete domains. Advances in neural information processing systems 11. Cambridge, Mass: MIT Press; 1999.

Griffiths Tom. Bayesian smoothing through text classification. Available from http://nlp.stanford.edu/courses/cs224n/2001/gruffydd.

Krugel Christopher, Toth Thomas, Kirda Engin. Service specific anomaly detection for network intrusion detection. In: Proceedings of Symposium on Applied Computing; 2002.

Lippmann Richard, Haines, Joshua W, Firied David J, Korba Jonathan, Das Kumar. Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. In: Proceedings of DARPA Information Survivability Conference and Exposition; 2000.

Mackie Andrew, Roculan Jensenne, Russell Ryan, Van Velzen Mario. Nimda worm analysis. Securityfocus.com Incident Analysis Report; 2001.

Mahoney Matthew V, Chan Philip K. PHAD: packet header anomaly detection for identifying hostile network traffic, Florida Tech. Technical Report CS-2001-4; 2001.

Patton Samuel, Yurcik William, Dos David. An Achilles' heel in signature-based IDS: squealing false positives in SNORT. In: Proceedings of Recent Advance in Intrusion Detection (RAID 2001); 2001.

Power R. CSI/FBI computer crime computer survey. Comput Secur Issues Trends 2002;8(1).

Ptacek Thomas H, Newsham Timothy N. Insertion, evasion, and denial of service: eluding network intrusion detection; 1998.

Rain Forest Puppy. A look at whisker's anti-IDS tactics. Available from http://www.wiretrip.net/rfp/pages/whitepapers/whiskerids.html.

Roesch M. Snort—lightweight intrusion detection for networks. In: Proceedings of USENIX LISA 1999; 1999.

Sanghyun Cho received the BS degree in Computer Science from Korea University, Korea, and the MS degree in Electrical Engineering and Computer Science from Korea Advanced Institute of Science and Technology (KAIST), Korea. He is a member of Certified Information Security Auditor (CISA) and Certified Information System Security Professional (CISSP). He is currently a doctoral candidate at the Department of Electrical Engineering and Computer Science from KAIST, Korea. His current research interests include network security, anomaly detection and computer forensics.

Sungdeok Cha received the BS, MS and PhD degrees in Information and Computer Science from the University of California, Irvine, in 1983, 1986, and 1991, respectively. From 1990 to 1994, he was a member of the technical staff at Hughes Aircraft Company, Ground Systems Group, and the Aerospace Corporation, where he worked on various projects on software safety and computer security. In 1994, he became a faculty member of the Korea Advanced Institute of Science and Technology (KAIST), Electrical Engineering and Computer Science Department. His research interest includes software safety, formal methods, and computer security.