Intrusion Detection using Text Processing Techniques with a Binary-Weighted Cosine Metric

Sanjay Rawat[1], V. P. Gulati[2], Arun K. Pujari[3] and V. Rao Vemuri[4]

[1]Intoto Softwares (I) Pvt Ltd
Uma Plaza, Nagarjuna Hills, Punjagutta, Hyderabad-500082, INDIA
sanjayr@intoto.com

[2]TCS, Deccan Park, Hyderabad, INDIA
vp.gulati@tcs.com

[3]AI Lab, Dept. of Computer and Information Sciences
University of Hyderabad, Hyderabad-500046, INDIA
akpcs@uohyd.ernet.in

[4]Department of Computer Science, University of California
Davis, CA 95616, USA
rvemuri@ucdavis.edu

Abstract: This paper introduces a new similarity measure, termed Binary Weighted Cosine (BWC) metric, for anomaly-based intrusion detection schemes that rely on using sequences of system calls. The new similarity measure considers both the number of shared system calls between two processes as well as frequencies of those calls. The k nearest neighbor (kNN) classifier is used to categorize a process as either normal or abnormal. The proposed BWC metric enhances the capabilities of simple kNN classifier significantly -especially in the context of intrusion detection. The experimental results obtained from 1998 DARPA Data, are very promising and show that the proposed scheme results in a high detection rate and low false positive rate.

Keywords: Intrusion Detection, Anomaly Detection, System Calls, kNN Classifier, Binary-Weighted Cosine Metric.

1 Introduction

With the widespread use of networked computers for critical systems, computer security is attracting increasing attention and intrusions have become a significant threat in recent years. Intrusions are attempts at compromising the confidentiality and integrity by bypassing the security mechanisms of a computer or network. Intrusion detection is the process of monitoring the events in a computer or a network and analyzing them for signs of intrusions. With the rapid growth of attacks on computers, intrusion detection systems (IDS), which are software or hardware products that automate this monitoring and analysis process [2], have become a critical component of security architecture.

According to one study [1], an IDS can be thought of as consisting of an Audit Collection/Storage Unit, Processing Unit and an Alarm/Response unit. The Audit Collection/Storage Unit collects data that is to be analyzed for signs of intrusion. The Processing Unit analyzes the data received from Audit Collection/Storage Unit to find the intrusions. The Alarm/Response Unit triggers an alarm on detecting an intrusion and it may execute defensive action too. Based on the various ways of managing these units, different types of IDS are proposed in the literature. On the basis of audit data, there are two types of IDS. The network-based systems collect data directly from the network that is being monitored, in the form of packets [17] and the host-based systems collect data from the host being protected [2]. IDS can also be classified, based on processing unit, into two types – misuse-based systems and anomaly-based systems. While the first keeps the signatures of known attacks in the database and compares new instances with the stored signatures to find attacks, the second learns the normal behavior of the monitored system and then looks out for any deviation in it for signs of intrusions. In the present work, we propose a novel intrusion detection method for host-based systems. A team of researchers at University of California, Davis has conducted a series of experiments on anomaly-based IDS using various methods, viz. kNN with cosine metric [15] and Robust Support Vector Machine [12], among others. While the former method adopts a simple approach and is easy to understand, the second method shows better results in terms of false positive rate and detection rate but with the added complexity of using SVMs. The proposed work draws inspiration from the work of Liao and Vemuri [15] on an anomaly-based intrusion detection system. The motivation behind the proposed work is to develop a method that is as simple to understand as the kNN method and yields results that are comparable to those obtained by using RSVM. By means of a series of experiments, we show that the proposed method (with the modified cosine metric) yields significantly better results than the kNN method with the original cosine metric.

The starting point of the method is the observation that any normal execution of a process follows a pattern and hence the normal behavior of a process can be profiled by a set of sequences of system calls. Any deviation in this pattern of system calls is termed an intrusion in the framework of anomaly-based IDS. The problem of intrusion detection thus boils down to detecting anomalous sequences of system calls, which are measurably different from the normal behavior. We propose a new scheme in which we measure the similarity between two processes using a metric that considers two factors - occurrence of system calls, which are common in the two said processes and the frequency of all system calls in the processes. We say that two processes have a common system call if during execution, both of the processes call that system call at some point of time i.e. that system call is present in the system call trace of both the processes. More precisely, let $\mathbf{P_1} = \{\mathbf{s^1_i}\}$ and

$\mathbf{P_2} = \{\mathbf{s_i^2}\}$ be the sets of unique system calls, invoked by the two processes P_1 and P_2 respectively, $s_i^j \in S$, where S is the set of all system calls. A system call $s_k \in \mathbf{P_1} \cap \mathbf{P_2}$ is said to be common to P_1 and P_2.

Due to the way it is constructed, we term this similarity metric Binary Weighted Cosine (BWC) metric. Adopting the method proposed by Liao and Vemuri, we make use of k-nearest neighbor scheme with our new similarity measure, thus gracefully extending their result, which did not consider the presence of common system calls. We illustrate that the similarity metric used by Liao and Vemuri may result in erroneous conclusions (for intrusion detection problems) and the BWC metric overcomes this shortcoming. The major contributions of our work are (a) the introduction of a novel similarity measure to capture both the commonality and frequency of occurrence of system calls and (b) the presentation of a comparative analysis to justify the use of kNN classifier for an improved detection method. We corroborate our claims of better IDS by experimental analysis.

Section 2 gives a brief survey on anomaly-based schemes to understand the different approaches. Section 3 presents some background and definitions that are used in the construction of our proposed scheme. We explain the scheme proposed by Liao and Vemuri [15] in section 4. Section 5 describes the proposed scheme in detail. Experimental results have been shown in the section 6. Analysis of the experimental results and a comparative study of BWC, RSVM and kNN with cosine metric are provided in the section 7. We conclude our work in section 8.

2 Related Work

Anomaly-based IDSs work on the assumption that an attack differs from normal activity. However, anomaly-based systems have high false positive rate [2]. Hence, a lot of research is being done in the area of anomaly-based intrusion detection [1]. A pioneering work [6], aimed at reducing the false positives, describes a model for detecting computer abuse by monitoring the system's audit records. In this approach, profiles of subjects (users) are learnt and statistical methods are used to calculate deviations from the normal behavior. Lane and Brodly [13] propose another approach that captures a user's behavior. A database of sequences of UNIX commands that normally a user issues, is maintained for each user. Any new command sequence is compared with this database using a similarity measure. Though the scheme gives good results, it is rather tedious to profile all the users, especially in big organizations. Moreover, since the behavioral pattern of users is not very stable, such models may give a high false positive rate. In another approach [8][9][11], normal behavior of processes is captured because programs show a stable behavior over the period of time under normal execution. In this approach, short sequences of system calls are used to profile a process. A similar approach is followed by Lee et al [14], but they make use of a rule learner RIPPER, to form the rules for classification. Artificial neural networks have also been used for anomaly detection [10] due to their ability to learn behavior and generalize. In this approach, they use the Leaky Bucket Algorithm to capture temporal locality. There has been some work on the notion of a mimicry attack, which allows sophisticated attackers to cloak their intrusion to avoid detection by the IDS. In this context it has ben shown that an abnormal activity, in particular a malicious sequence of system calls, can be converted into a normal-looking sequence [19].

In a very recent study [3], neural networks are used with the *Soundex algorithm* which is designed to change feature selection and variable length system call data into a fixed length learning pattern. In the approach, the variable length sequential system call data is transformed into a fixed length behavior pattern using the Soundex algorithm and neural network is learnt by using the back-propagation algorithm. The proposed method and N-gram technique are applied for anomaly intrusion detection of system call using Sendmail Data of UNM to demonstrate its performance.

The thrust of this paper is to build upon a recently published method proposed by Liao and Vemuri [15][16]. In this method, based on the kNN Classifier, each system call is treated as a *word* and a collection of system calls during the execution of a process is treated as a *document*. The system call trace of the process is converted into a vector and cosine similarity measurement is used to calculate the similarity among processes. The proposed scheme in the present paper also follows an analogous approach by using a kNN classifier, except that a modified similarity measure - the BWC metric - is used instead. In another study at the University of California, Davis [12], the Robust Support Vector Machine (RSVM) has been applied to anomaly-based IDS. The emphasis of this RSVM study is on exhibiting the effectiveness of the method in the presence of noisy data. We show comparative results of our scheme with this RSVM method also.

3 Feature Vector and Similarity Measure

Let S (say, $Card(S) = m$) be a set of system calls made by all the processes under normal execution. From all the normal processes a matrix $A = [a_{ij}]$ is formed, where a_{ij} denotes the frequency of i^{th} system call in the j^{th} process. We also form a matrix $B = [b_{ij}]$ where, $b_{ij} = 1$, if i^{th} system calls is present in the j^{th} process, otherwise $b_{ij} = 0$. Thus the binary representation of process P, namely Pb_j, is defined by the $m-$ dimensional vector $Pb_j = [0, 1]^m$ as a column in B. For example,
Let $S = \{$`access audit chdir close creat exit fork ioctl`$\}$.
Let the system call traces of two normal processes be
$P_1 =$ `access close ioctl access exit`, and $P_2 =$ `ioctl audit chdir chdir access`. Then we have:

$$A = \begin{array}{c} \begin{array}{cc} P_1 & P_2 \end{array} \\ \begin{pmatrix} 2 & 1 \\ 0 & 1 \\ 0 & 2 \\ 1 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} \end{array} \qquad B = \begin{array}{c} \begin{array}{cc} Pb_1 & Pb_2 \end{array} \\ \begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} \end{array}$$

The rows of A (and B) correspond to the elements of S in the same order and columns of A (and B) correspond to processes P_1 and P_2. Thus the first entry in A is calculated by counting the frequency of system call `access` in the process P_1 that is 2. Similarly the first entry of the second column of A is calculated by counting the frequency of the system call `access` in the process P_2, which is 1, and so on. Similarly the first entry of the first column of B is 1 because the system call `access` is present in P_1 whereas the second entry is 0, which shows that the system call `audit` is absent in P_1.

We now define similarity measures, which we use in our scheme to calculate the similarity between processes.

3.1 Binary Similarity Measure

We define a similarity score $\mu(Pb_i, Pb_j)$ between any two processes Pb_i and Pb_j (i^{th} and j^{th} columns of B) as follows:

$$\mu(Pb_i, Pb_j) = \frac{\displaystyle\sum_{l=1}^{m} (Pb_i \wedge Pb_j)_l}{\displaystyle\sum_{nl=1}^{m} (Pb_i \vee Pb_j)_l} \qquad (1)$$

where the summation runs over l, which is a subscript on the elements of the processes Pb_i and Pb_j and m is the length of each process vector.

It may be noticed that $0 \leq \mu \leq 1$. The value of μ increases when there are more common system calls between the two processes (due to the numerator) and value of μ decreases when the number of uncommon system calls, is more than the common ones (due to the denominator) in Pb_i and Pb_j.

3.2 Frequency Similarity Measure

Another similarity score, known as cosine similarity measure $\lambda(P_i, P_j)$ between the processes P_i and P_j (i^{th} and j^{th} columns of A) is defined as follows [4]:

$$\lambda(P_i, P_j) = \frac{P_i \cdot P_j}{\parallel P_i \parallel \cdot \parallel P_j \parallel} \tag{2}$$

where $\parallel X \parallel = \sqrt{X \cdot X}$.
It may be noted that eqn. 2 represents the same similarity measure as used by Liao and Vemuri [15].

3.3 Binary Weighted Cosine Metric

We define our new similarity measure, termed as *Binary Weighted Cosine* (BWC) metric, $Sim(P_i, P_j)$ as follows:

$$Sim(P_i, P_j) = \mu(Pb_i, Pb_j) \cdot \lambda(P_i, P_j) \tag{3}$$

The motive behind multiplying μ and λ is that $\lambda(P_i, P_j)$ measures the similarity based on the frequency and $\mu(Pb_i, Pb_j)$ is the weight associated with P_i and P_j. In other words, $\mu(Pb_i, Pb_j)$ tunes the similarity score $\lambda(P_i, P_j)$ according to the number of similar and dissimilar system calls between the two processes. Therefore, the similarity measure $Sim(P_i, P_j)$ takes frequency and the number of common system calls into consideration while calculating similarity between two processes.

The following section summarizes the scheme based on kNN-classifier [15]. We also identify some cases in which the kNN classifier leads to erroneous conclusions.

4 k-NN Classifier with Cosine Metric

An approach based on *kNN classifier* is proposed by Liao and Vemuri [15] where the frequencies of system calls used by a program (process), rather than their temporal ordering, are used to define program's behavior. Their approach draws an analogy between text categorization and intrusion detection, such that each system call is treated as a word and a set of system calls generated by a process as a document. The processes under normal execution are collected from the DARPA data and thereafter converted into vectors (as described earlier), consisting of the frequencies of the system calls made by them during the normal execution. From all the normal processes, a matrix $A = [a_{ij}]$ is formed, where a_{ij} denotes the frequency of i^{th} system call in the j^{th} process. In order to categorize a new process P into either normal or abnormal class, the process P is first converted into a vector. The *kNN classifier* then compares it with all the processes A_j in A to determine the k nearest neighbors, by calculating the similarity $CosSim(P, A_j)$, using the cosine formula given by

$$CosSim(P, A_j) = \frac{P \cdot A_j}{\parallel P \parallel \cdot \parallel A_j \parallel} \tag{4}$$

The average similarity value of the k nearest neighbors is calculated and a threshold is set. When the average similarity value is above the threshold, process P is considered as normal, and if not, abnormal. Since the similarity measure, given by equation 4, considers only the frequencies of the system calls appearing in the processes, sometimes it may produce erroneous results while calculating similarity as illustrated below.
Consider the sequence of system calls associated with the following two processes P_1 and P_2
P_1 = `open close close close close access access access access`
P_2= `open ioctl mmap pipe access login su su audit audit`
Let us consider a third process P given by the sequence of calls as
P= `open close ioctl mmap pipe pipe access access login chmod`
The similarity score of the new process P with P_1 is:

$$CosSim(P, P_1) = 0.6048$$

and the similarity score of the new process P with P_2 is:

$$CosSim(P, P_2) = 0.5714$$

We observe that there are only three common system calls out of eight between P and P_1 and six common system calls out of eight between P and P_2. Intuitively, P_2 is more similar to P than P_1, but the similarity measures, calculated above indicate the contrary. This is due to the frequent occurrence of `close` and `access` in P_1, and absence of `close` in P_2. The above example makes it clear that while calculating the similarity score, there is no weight accorded to processes having more number of common system calls. We believe that while calculating the similarity score, if we include a factor that depends on the number of common calls, such anomalous results can be avoided. Therefore, in our scheme, we define a similarity measure that depends not only on frequencies of system calls but also on the number of common system calls between the processes.

With this background and definitions, we describe, in the next section, the proposed method for anomaly intrusion detection.

5 Proposed Scheme

As discussed in section 3, the matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ are constructed using normal processes and the set S. For every new process P, if P contains a system call that is not in S, the process P is classified as abnormal; if not, it is first converted into a vector for further processing. The binary equivalent Pb of this vector is calculated. Next, the similarity score $\lambda(P, P_j)$ is calculated for every normal vector P_j by using equation 2. If $\lambda(P, P_j) = 1$, P is classified as normal. Otherwise, using equations 1 and 3, the values of $\mu(P, P_j)$ and $Sim(P, P_j)$ are calculated. Values of $Sim(P, P_j)$ are sorted in descending order and the k nearest neighbors (first k highest values) are chosen. We calculate the average value (Avg_Sim) of the k nearest neighbors. The kNN classifier categorizes the new process P as either normal or abnormal according to the rule given below.

```
If Avg_Sim > Sim_Threshold, classify P as normal,
           otherwise P is abnormal.
```

where Sim_Threshold is a predefined threshold value for similarity score. The pseudo code for the proposed scheme is shown in Figure 1.

Using the above-defined scheme, if we re-calculate the similarity of process P with the processes P_1 and P_2, given in section 4, using the equations 1, 2 and 3, we get

$$Sim(P, P_1) = 0.2268 \text{ and } Sim(P, P_2) = 0.3429$$

The results obtained above validate our hypothesis empirically.

6 Experimental Setup and Results

We use BSM audit logs from the 1998 DARPA data [5] for training and testing of our algorithm. As we wish to compare our proposed method with that of Liao and Vemuri, we use the same data set that is used in their scheme. For completeness, we summarize the whole process of extracting the system call data below.

After analyzing the training data, we extract the 50 unique system calls that appear in the training data. All the 50 system calls are shown in table 1. For each day of data, a separate BSM file is provided with the 'BSM List File'. Each line of this file contains the information about one session such as time, service, source IP and destination IP. A '0' at the end of the line shows that the session is normal and the presence of a '1' at the end of the line declares the session intrusive. All the intrusive sessions are labeled with the name of the attacks launched during the sessions. We

```
Given a set of processes and system calls $S$, form the matrices
$A = [a_{ij}]$ and $B = [b_{ij}]$
for each process $P$ in the test data do
    if $P$ has some system calls which does not belong to $S$ then
        $P$ is abnormal; exit.
    else then
        for each process $A_j$ in the training data $A$ do
            calculate $Sim(P, A_j)$;
            if $Sim(P, A_j)$ equals 1.0 then
                $P$ is normal; exit.
        end do
        find first $k$ highest values of $Sim(P, A_j)$;
        calculate Avg_Sim for $k$ nearest neighbors so obtained;
        if Avg_Sim is greater than Sim_Threshold then
            $P$ is normal;
        else then
            $P$ is abnormal;
end do
```

Figure 1: Pseudo code of the proposed scheme

Table 1: List of 50 unique system calls

```
access, audit, auditon, chdir, chmod, chown, close,
creat, execve, exit, fchdir, fchown, fcntl, fork, fork1,
getaudit, getmsg, ioctl, kill, link, login, logout, lstat,
memcntl, mkdir, mmap, munmap, nice, open, pathconf, pipe,
putmsg, readlink, rename, rmdir, setaudit, setegid,
seteuid, setgid, setgroups, setpgrp, setrlimit, setuid,
stat, statvfs, su, sysinfo, unlink, utime, vfork
```

Table 2: List of 55 attacks used in testing data set

1.1_it_ffb_clear, 1.1_it_format_clear, 2.2_it_ipsweep, 2.5_it_ftpwrite, 2.5_it_ftpwrite_test, 3.1_it_ffb_clear, 3.3_it_ftpwrite, 3.3_it_ftpwrite_test, 3.4_it_warez, 3.5_it_warezmaster, 4.1_it_080520warezclient, 4.2_it_080511warezclient, 4.2_it_153736spy, 4.2_it_153736spy_test, 4.2_it_153812spy, 4.4_it_080514warezclient, 4.4_it_080514warezclient_test, 4.4_it_175320warezclient, 4.4_it_180326warezclient, 4.4_it_180955warezclient, 4.4_it_181945warezclient, 4.5_it_092212ffb, 4.5_it_141011loadmodule, 4.5_it_162228loadmodule, 4.5_it_174726loadmodule, 4.5_it_format, 5.1_it_141020ffb, 5.1_it_174729ffb_exec, 5.1_it_format, 5.2_it_144308eject_clear, 5.2_it_163909eject_clear, 5.3_it_eject_steal, 5.5_it_eject, 5.5_it_fdformat, 5.5_it_fdformat_chmod, 6.4_it_090647ffb, 6.4_it_093203eject, 6.4_it_095046eject, 6.4_it_100014eject, 6.4_it_122156eject, 6.4_it_144331ffb, test.1.2_format, test.1.2_format2, test.1.3_eject, test.1.3_httptunnel, test.1.4_eject, test.1.5_processtable, test.2.1_111516ffb, test.2.1_format, test.2.2_xsnoop, test.2.3_ps, test.2.3_ps_b, test.2.5_ftpwrite, test.2.4_eject_a, test.2.2_format1

make use of the BSM commands `auditreduce` and `praudit` and a couple of shell scripts to extract the data that can be used in our algorithm. On analyzing the entire set of BSM logs (list files), we locate five days which are free of any type of attacks - Tuesday of the third week, Thursday of the fifth week and Monday, Tuesday and Wednesday of the seventh week. We choose the first four days of data for creating the normal training data set and the fifth day's data for creating the normal test data. There are about 2000 normal sessions reported in the four days of data. We extract the processes occurring during these days and our training data set consists of 606 unique processes. There are 412 normal sessions on the fifth day and we extract 5285 normal processes from these sessions. We use these 5285 normal processes as testing data. In order to test the detection capability of our method, we incorporate 55 intrusive sessions into our testing data. Table 2 lists these attacks. A number in the beginning of the name denotes the week and day followed by the name of the session (attack). For example, the attack name 3.1_it_ffb_clear means that the attack was launched in the 3^{rd} week, on the 1^{st} day viz. Monday and the name of the attack is ffb (in clear mode). The table should be read horizontally from left to right and top to bottom. These attack sessions consist of almost all types of attacks launched on the victim Solaris machine (in the simulated DARPA setup) during seven weeks of training period and two weeks of testing period and that can be detected using BSM logs. An intrusive session is said to be detected if any of the processes associated with this session is classified as abnormal. Thus detection rate is defined as the number of intrusive sessions detected, divided by the total number of intrusive sessions. We perform the experiments with k = 5, 10 and 15. Table 3 shows the results for k = 5 and 10. The first column in the table shows the threshold values used in the experiments. Entries in column two are the rate of false positives, which is equal to the number of normal processes detected as abnormal divided by the total number of normal processes. Column three details the detection rate as defined above. We have not shown the results for k = 15 as we do not find any significant difference from the case of k = 10. In order to make a valid comparison, we repeat the experiment with Liao and Vemuri's scheme and table 4 summarizes the results for k =10. Notice that the false positive rate is very high at a detection rate of 100%. Satisfactory explanation of this behavior is conjectural, at best, at this point. Based on some preliminary experiments, not reported in this paper, we are tempted to believe that the attack data sets perhaps contained some processes that show a high degree of similarity with normal processes when the original cosine metric was used.

Table 3: False Positive Rate vs Detection Rate for k= 5, 10 for BWC scheme

Threshhold value	k = 5		k = 10	
	False positive rate	Detection rate	False positive rate	Detection rate
0.52	0.00	0.36	0.00	0.36
0.55	0.00	0.36	0.007	0.38
0.60	0.0003	0.38	0.009	0.76
0.65	0.007	0.52	0.020	0.83
0.70	0.011	0.85	0.024	0.87
0.74	0.022	0.89	0.049	0.94
0.78	0.048	0.96	0.055	0.96
0.80	0.048	0.96	0.10	1.00
0.84	0.052	0.96	0.14	1.00
0.86	0.077	0.96	-	-
0.89	0.087	1.00	-	-

7 Discussion And Analysis

We show the comparative results of BWC scheme with that for Liao and Vemuri and Hu et al. As can be seen in the figure 2, the ROC for k = 5 tends to be lower than that for k = 10, therefore we draw ROC curves for BWC scheme at k = 5 for comparison with Liao and Vemuri method, which is illustrated in figure 3. The ROC curve is a graph between detection rate and false positive rate. For each threshold value we get the detection rate and false positive rate.

The BWC scheme achieves a detection rate of 100% at a false positive rate of 8% whereas Liao and Vemuri scheme achieves 100% detection rate at 39% false positive. If we look carefully at table 3 for k = 5, we find that from threshold value 0.78

Table 4: False Positive Rate vs Detection Rate for k= 10 for Liao and Vemuri scheme

Threshold	False Positive Rate	Detection Rate
0.50	0.00	0.34
0.55	0.0009	0.34
0.70	0.001	0.36
0.78	0.001	0.74
0.85	0.002	0.74
0.88	0.031	0.76
0.90	0.034	0.78
0.93	0.035	0.81
0.95	0.035	0.83
0.97	0.044	0.92
0.98	0.091	0.96
0.99	0.33	0.96
0.992	0.36	0.98
0.994	0.39	1.00

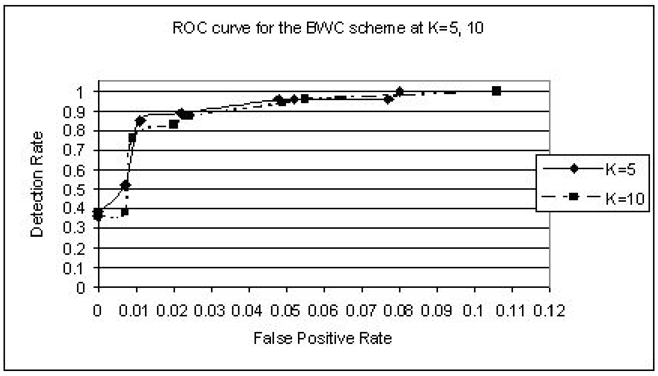

Figure 2: ROC curve for BWC Scheme at k =5, 10

to 0.86, the detection rate is constant. When we investigate, we find that there are two attacks included in the testing data viz. 4.5_it_162228loadmodule and 5.5_it_fdformat_chmod, which could not be detected by BWC method whereas Liao and Vemuri scheme detected them at a lower threshold. On consulting the document on DARPA attack description, which provides various details about the attacks, we find that though the attack 4.5_it_162228loadmodule was launched, it failed to compromise the system. In other words, the processes associated with the attack behaved normally. Therefore it is not a surprise that BWC method could not detect the attack because there indeed was no attack. In case of the second attack 5.5_it_fdformat_chmod, we find that this attack is launched in various stages and the included instance was in stage 2. Therefore, we may think that attack has not manifested and thus may not be detected. In fact, these events validate that the proposed BWC method checks more closely and rigorously the similarity between the processes. This is attributed to the binary similarity metric as its value increases with the number of system calls, common to the processes being compared. We repeat the whole experiment after removing these two attacks from our testing data set and the results are shown in the Table 5. It can be seen in Table 5 that detection rate reaches 100% with false positive rate as low as 4% only. With this reduced test data set also, Liao and Vemuri scheme gives almost the same results as with the previous data. This could be due to the reason that their scheme detected these attacks at early stage at low threshold with original test data set. Therefore even after removing these attacks, it does not improve in performance.

Another set of experiments is carried out to show the effectiveness of the BWC scheme for noisy data. This part of the section aims to attest the idea that an approach as simple as kNN classifier can be used to produce results that are comparable to the results obtained by more complex method such as the RSVM.

The Robust Support Vector Machine (RSVM) has been applied

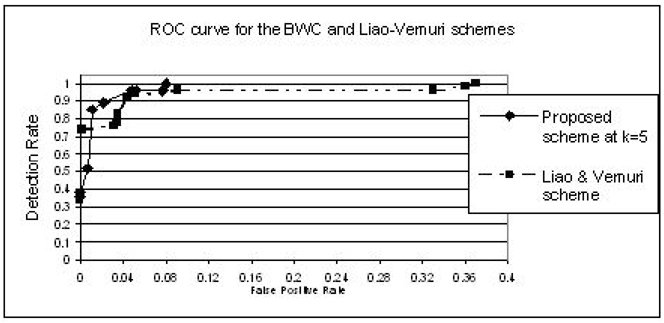

Figure 3: ROC curve for BWC Scheme and Liao and Vemuri scheme

Table 5: False Positive Rate vs Detection Rate for k= 5, 10 for BWC scheme after removing two attacks

Threshold value	k = 5		k = 10	
	False positive rate	Detection rate	False positive rate	Detection rate
0.52	0.00	0.37	0.00	0.37
0.55	0.00	0.37	0.007	0.39
0.60	0.0003	0.39	0.009	0.79
0.65	0.007	0.54	0.020	0.86
0.70	0.011	0.88	0.024	0.90
0.74	0.022	0.92	0.049	0.98
0.78	0.048	1.00	0.055	1.00

to anomaly-based IDS. The emphasis is to exhibit the effectiveness of the method in the presence of noise in data. The experiments have been performed on the same BSM DARPA'98 data set that we discussed earlier. A detailed description can be found in [12]. Briefly, we mention below the data sets used for the above experiments and the results thus obtained.

There are two training sets - clean data set and noisy data set. The clean data set contains 300 normal processes and 12 abnormal processes with correct labels. The noisy data set consists of 300 normal processes and 28 abnormal processes. Of these 28 abnormal processes, 16 have been mislabeled as normal (and thus noisy) and the remaining 12 are correctly labeled as abnormal. The testing data set contains 5285 normal processes and 22 intrusive sessions from the two-week testing data of DARPA. We reproduce the SVM and RSVM results [12] in table 6 for ready reference. It has been

Table 6: False Positive Rate vs Detection Rate of RSVM and SVM under clean and noisy data

Method	Clean Data		Noisy Data	
	False positive rate	Detection rate	False positive rate	Detection rate
RSVM	3.0%	100%	8.0%	100%
SVM	14.2%	100%	100%	100%

noticed by Eskin et al [7] that in practical situations the normal data outnumbers the intrusion data by a factor of 100:1. If we look at DARPA data we find that the frequency of intrusive sessions is very low compared to that of normal sessions. In other words, if we collect normal data in a real working environment and if the data is contaminated by the presence of some intrusive data, the proportion of intrusive data will be very small. Whereas the above noisy data set used by Hu et al is not in this proportion. We also find it difficult to decide the appropriate ratio of normal and abnormal processes in the data. We, therefore, choose a middle path and performed the experiments on a data set in which we try to maintain the ratio of normal and abnormal processes as 100:2 so that proportion of normal and abnormal processes is near to the realistic scenario (namely, 100:1) and we retained the 100:5 proportion for the RSVM experiments. Our training data set consists of 622 normal processes (606 normal + 16 abnormal processes, mislabeled as normal). Out of these 622 processes, 606 are the same as those used in earlier experiments. The testing data set consists of 22 intrusive sessions launched over the two-weeks of testing period in DARPA setup and 5285 normal processes. Table 7 shows the experimental results with the aforementioned data sets. It can be

Table 7: False Positive Rate vs Detection Rate of BWC scheme under noisy data

Threshold value	$k = 5$		$k = 10$	
	False positive rate	Detection rate	False positive rate	Detection rate
0.52	0.00	0.00	0.00	0.00
0.60	0.0003	0.00	0.007	0.45
0.65	0.006	0.36	0.020	0.50
0.70	0.017	0.59	0.03	0.72
0.74	0.029	0.68	0.05	0.90
0.78	0.054	0.68	0.06	1.00
0.80	0.055	0.72	0.08	1.00
0.84	0.059	0.90	0.14	1.00
0.86	0.083	0.90	-	-
0.89	0.09	1.00	-	-
0.92	0.32	1.00	-	-

seen from table 7 that the detection rate with $k = 10$, reaches 100% with a false positive rate of only 6%, whereas in case of RSVM the detection rate is 100% with a false positive rate of 8.0%. It should be noted that in case of RSVM number of training instances are 316 whereas they are 622 in our experiment. We also performed an experiment with 316 training instances (300 normal + 16 abnormal processes as described above) at $k = 10$ and we could get a detection rate of 100% with a false positive rate of 12%. These figures reveal that BWC scheme needs larger number of normal processes for training as compared to RSVM method.

We also obtained results for the clean training data set (consisting of 606 normal processes and 12 abnormal processes) with $k = 5$. Each new process is first compared with each of the 12 processes by calculating similarity scores. If the score equals one, the new process is classified as abnormal otherwise regular method described in section 5 is followed. Table 8 summarizes the results. With clean data RSVM achieves 100% detection rate with

Table 8: False Positive Rate vs Detection Rate of BWC scheme under clean data

Threshold	False Positive Rate	Detection Rate
0.60	0.0003	0.18
0.65	0.007	0.72
0.70	0.01	0.90
0.74	0.02	0.95
0.78	0.05	1.00

3% false positives, whereas the BWC scheme reaches a detection rate of 100% at 5% false positive rate. Therefore, from performance standpoint, the BWC scheme is slightly inferior to RSVM. Another criterion for performance measurement, mentioned in [12] is that an intrusion detection system should not produce false positive rate greater than 1% with as high detection rate as possible. Table 9 shows the detection rates of BWC, RSVM and SVM methods over the clean data set at false positive rate under 1%. In this scenario, the BWC method performs better than RSVM. From these observations, it can be inferred that our proposed scheme is able to perform well in the presence of noisy data, but the amount of training data required is more than in the case of RSVM method.

Table 9: Detection rate of BWC, RSVM and SVM at a fixed false positive rate of 1%

Method	False Positive Rate	Detection Rate
BWC	90.0%	1%
RSVM	81.8%	1%
SVM	81.8%	1%

If we look at the complexity, our method is simpler than the RSVM method.

In view of the above experiments and the analysis, it can be asserted that the proposed similarity metric performs better than the simple cosine similarity metric. It can also be noticed that minimum threshold value at which the proposed metric could identify all the abnormal processes (0.78) is lower than that in case of Liao and Vemuri scheme (0.994). It shows that separation boundary created by proposed similarity metric is much wider, thus facilitating to minimize the overlap between normal and abnormal processes.

8 Conclusions

All anomaly-based intrusion detection systems work on the assumption that normal activities differ from the abnormal activities (intrusions) substantially. In the case of IDS models that learn a program's behavior, these differences may manifest in the form of (a) the frequency of system calls, (b) the number of common system calls invoked by the processes and (c) the ordering of system calls used by the processes under normal and abnormal execution. Our BWC scheme considers the first two of these factors while classifying a new process as normal or abnormal. The new BWC metric has two factors. One factor calculates the similarity between the two processes based on the frequency of system calls by making use of the cosine metric. The second factor operates on the binary form of the process vectors and calculates the similarity between the two processes based on the number of the common system calls in those processes.

With the BWC metric, we obtain a detection rate of 100% at a false positive rate of 4%, which is a significant improvement over the performance of the scheme proposed by Liao and Vemuri, which makes use of kNN classifier with cosine metric. At a fixed false positive rate of 1%, WBC scheme outperforms with a detection rate of 90% as compared to the detection rate of 81.8% with other schemes. The experiments show that the proposed BWC scheme performs consistently even in the presence of noise in the training data and the results, thus obtained, are comparable to one of the recently published study with RSVM. We believe that the proposed method takes a very simple approach by using kNN classifier and is easier to understand and implement than the RSVM.

Acknowledgment

We thank Mr. Yihua Liao of University of California, Davis, US for providing help in getting DARPA training and testing data. The first author also thanks IDRBT for providing infrastructure and financial assistance to carry out this research. At the time of this work, the author was associated with IDRBT as research fellow and with University of Hyderabad as PhD scholar. Work of Prof. Vemuri is supported in part by AFOSR's grant number FA95500410159. Work of Prof. Pujari is supported by MIT (Govt. of India) grant no. 12(22)/04-IRSD.

References

[1] S. Axelsson. "Research in intrusion detection systems: A survey". Technical Report No. 98-17, Dept. of Computer Engineering, Chalmers University of Technology, Gteborg, Sweden, 1999.

[2] R. Bace, P. Mell. "NIST special publication on intrusion detection system". SP800-31, NIST, Gaithersburg, MD, 2001.

[3] B. Cha, B. Vaidya, S. Han. "Anomaly Intrusion Detection for System Call Using the Soundex Algorithm and Neural Networks". In Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC'05), pp. 427-433, 2005.

[4] Z. Chan, B. Zhu. "Some Formal Analysis of the Rocchio's Similarity-based Relevance Feedback Algorithm". Technical Report CS-00-22, Dept. of Computer Science, University of Texas-Pan American, Edinburg, TX, 2000.

[5] DARPA 1998 Data, MIT Lincoln Laboratory, http://www.ll.mit.edu/IST/ideval/data/data_index.html

[6] D.E. Denning. "An Intrusion-Detection Model". In Proceedings of the 1986 IEEE Symposium on Security and Privacy (SSP '86), IEEE Computer Society Press, pp. 118-133, 1990.

[7] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, S. Stolfo. "A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data",. in Applications of Data Mining in Computer Security, D. Barbara, S. Jajodia (eds.), Kluwer academics Publishers, pp. 77-102, 2002.

[8] S. Forrest, S. A. Hofmeyr, A. Somayaji, T. A. Longstaff. "A Sense of Self for Unix Processes". In Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy, Los Alamitos, CA, IEEE Computer Society Press, pp. 120-128, 1996.

[9] S. Forrest, S. A. Hofmeyr, A. Somayaji. "Computer Immunology", Communications of the ACM, 40(10), pp. 88-96, 1997.

[10] A. K. Ghosh, A. Schwartzbard. "A Study in Using Neural Networks for Anomaly and Misuse Detection". In Proceedings of the 8th USENIX security Symposium, Washington D C USA, pp. 141-151 1999.

[11] S. A. Hofmeyr, S. Forrest, A. Somayaji. "Intrusion Detection Using Sequences of System Calls", Journal of Computer Security, 6, pp. 151-180, 1998.

[12] Wenjie Hu , Y. Liao, V. Vemuri. "Robust Support Vector Machines for Anamoly Detection in Computer Security". In International Conference on Machine Learning, Los Angeles, CA. 2003.

[13] T. Lane, C. E. Brodly. "An Application of Machine Learning to Anomaly Detection". In Proceeding of the 20th National Information System Security Conference, Baltimore, MD, pp. 366-377, 1997.

[14] W. Lee, S. Stolfo, P. Chan. "Learning Patterns from Unix Process Execution Traces for Intrusion Detection". In Proceedings of the AAAI97 workshop on AI methods in Fraud and risk management, AAAI Press, pp. 50-56, 1997.

[15] Y. Liao, V. R. Vemuri. "Use of K-Nearest Neighbor Classifier for Intrusion Detection", Computers & Security, 21(5), pp. 439-448, 2002.

[16] Y. Liao, V. R. Vemuri. "Using Text Categorization Techniques for Intrusion Detection". In Proceedings USENIX Security 2002, San Francisco, US, pp. 51-59, 2002.

[17] B. Mukherjee, L. T. Heberlein, K. N. Levitt. "Network Intrusion Detection", IEEE Network, 8(3), pp. 26-41, 1994.

[18] G. Tandon, P. Chan. "Learning Rules from System Calls Arguments and Sequences for Anomaly Detection". In ICDM Workshop on Data Mining for Computer Security (DMSEC), Melbourne, FL, pp. 20-29, 2003.

[19] D. Wagner, P. Soto. "Mimicry Attacks on Host-Based Intrusion Detection Sytems". In Proceedings of the 9th ACM conference on Conputers and Communications Security (CCS'2002), Washington, DC, USA, 2002.

Author Biographies

Sanjay Rawat was born in Tundla, India in 1976. He has submitted his PhD thesis (computer science) to University of Hyderabad, Hyderabad, India. He was also associated with Institute for Development and Research in Banking Technology, Hyderabad as research fellow. He received his MSc in mathematics and Mphil in mathematics/cryptography. Presently he is working with Intoto Softwares (I) Pvt. Ltd, Hyderabad, India as Intrusion Analyst. His areas of interest are network security (Intrusion Detection) and cryptography.

Dr. V. P. Gulati recieved his PhD from the Indian Institute of Technology, Kanpur, India. Dr Gulati was a member of Technology Upgradation in Banking Sector Committee. He is a consultant to many Banks. Earlier, he was a Professor at National Institute of Bank Management (NIBM) Pune, in the area of Computer and Systems. At NIBM, he has conducted large number of Management Development Programs in the area of Information Technology and organised number of conferences on Financial Network, Electronic Fund Transfer, Electronic Data Interchange, and related issues. Currently, he is working with TCS, India as Consultant Advisor.

Prof. Arun K. Pujari is a professor and Dean of Computer Science dept. at University of Hyderabad, Hyderabad, India. Prior to joining UoH, he served at the Automated Cartography Cell, Survey of India, and Jawaharlal Nehru University, New Delhi. He recieved his PhD from the Inian Institute of Technology, Kanpur and MSc from Sambalpur University, Sambalpur. His areas of interest are Combinatorial Algorithms, Data Mining, Logic & Reasoning and Security.He has also undertaken several visiting assigments at the Institute of industrial sciences, University of Tokyo; International Institute of software Technology, United Nations University, Macau; University of Memphis, USA; and Griffith University, Australiya, among others.

Prof. V. Rao Vemuri holds a joint appointment as a professor in the departments of Computer Science and Applied Science of the University of California, Davis. He also holds a position as a Computer Scientist at the Lawrence Livermore National Laboratory. His research interests are in the application of machine learning techniques to a variety of applications including computer security. In his spare time he writes in English and Telugu to popularize science. He is the founding president of Eco Foundation and received the Distinguished Public Service Award form his university. He is a senior member of IEEE and a member of ACM.