Accessing Rarity of System Call Patterns for Anomaly Detection

Mr. Kulkarni Sagar S.[1] Prof. Kahate Sandip A.[2]
[1]Student [2]Assistant Professor
[1,2]Department of Computer
[1,2]SP COE, Otur, Pune, India

Abstract— Today computer security is important field research because more and more security threats arising in daily basis. Many researches focus on use intrusion detection system so as to prevent these intrusions. But most of the intrusion detection system suffers from high FPR and TPR. This is because of incomplete training and strategy used by them for detecting suspicious behaviour. This paper presents a host based intrusion detection that uses natural language concepts and rarity index to find anomalous system call trace. Also proposed system uses approximate pattern matching over mismatched sequence to defend against incomplete training. Evaluation of proposed is carried using publicly available UNM intrusion detection dataset. The results support the proposed system.

Key words: Anomaly Detection; Intrusion Detection; Semantic Theory

I. INTRODUCTION

The widespread use Internet technology brings many opportunities and challenges for users. In order to face these challenges many more techniques and technologies were developed and intrusion detection system is one of them [5]. Intrusion detection system is used to detect activities that violate security of the system to steal/damage important data. The use of intrusion detection system increased over years and techniques it uses are getting much more complex. Large number of researches on intrusion detection system focuses on packet level data but it is more effective if we enhance the end point security.

Intrusion detection system categorized into Host-based or Network-Based based. This categorization is based on where it is installed and data it uses to find intrusion [1]. Also particular IDS can be subdivided into misuse based detection and anomaly based detection. A misuse based detection uses signature of attacks to detect intrusion and such system have high detection rate with nearly zero FPR. But misuse based detection suffers from high FNR, it means anomalous patterns can be categorized as normal. This happens when intrusion sequence is not present in IDS database. While anomaly detection uses normal profile and any significant deviation with respect to normal profile is considered as anomaly. Anomaly based detection have high DR but it suffers from high FPR, because of incomplete normal profile [1][2].

Number of approaches used for intrusion detection some uses machine learning, data mining, statistical analysis tools [2][3][10]. Some system used network traffic, system log files, system parameters, audit logs and system call traces for intrusion detection. Here a host based intrusion detection system that uses system call patterns for anomaly detection is proposed. It also addresses problems associated with anomaly based intrusion detection system and provides solution.

The rest of paper is organized as follows: Section 2 covers literature review. Section 3 presents theories which supports proposed system. Section 4 contains system design, Section 5 gives algorithms, Section 6 experimental result and Section 7 contains concluding remarks.

II. LITERATURE

Forrest et. al. [4] proposed model uses fixed length patterns for anomaly detection. In paper [5] Forrest et. al proposed extension to their earlier work that uses hamming distance measure so as to solve incomplete training profile problem. Warrender et. al [6] proposed model [STIDE] uses Locality Frame Count that aggregates mismatches occurred in LFC window and applied threshold. Another model proposed by Warrender et. al [6] uses frequency of system call patterns to train their model. According to them rare sequences are suspicious, so they removed patterns that are less frequent. In [7] Wespi et. al proposed a model variable length patterns to profile normal behavior of process. These patterns are generated using Teiresias pattern matching algorithm. The total mismatches occurred are then used for detecting intrusion. In [8] Vardi et. al proposed model accesses rarity of system call patterns to flag the system call trace as anomalous. According to them frequency count is not a good measure to access popularity of system call sequence, so they used another measure to find rarity. They applied their concept over STIDE [6] in which LFC is set as rarity threshold. n [10] Liao et. al model uses text categorization technique. They represent program behavior by frequencies of system calls executed. Each system call is treated as a word and the collection of system calls over each program execution as a document. These documents are then classified using k-NN classifier, a popular method in text categorization. Gedion et. al [11] proposed model uses semantic analysis of system call patterns. Relationship between patterns is accessed and strength of this relation is used to find out anomaly. But time required for training is too high also FPR increases as detection rate reaches above 80%. In [12] Zhang et. al. model uses variable length system call patterns and average hamming distance to find anomalous sequence. There are some other researches [13][14] that uses system calls with their arguments. But none of these approaches able get high detection rate with reduced FPR.

The proposed system uses combination of different approaches to increase the detection rate with reduced FPR. The variable length patterns are extracted as it proved to be effective in previous researches, relationship between different words is determined using semantic theory, the rarity index model is used because frequency based measure is not accurate to find out popularity of different system call patterns and problem of incomplete training sequences is solved using hamming distance measure. Following section gives detailed description of proposed system.

III. PRELIMINARIES

A. Accessing Relationship between Patterns:

We know that, program in execution is called process and mostly these program written in higher level languages that follows some grammar. Thus programs written in those programming languages also follows similar grammar. So sequences generated after execution of these programs also follows similar grammar as that of program. Also most of programs uses inbuilt functions from programming languages, this means every time a inbuilt function is called it will generate nearly similar set of system call patterns. To model system call sequences in natural language, consider each system call as a 'letter', then collection of continuous system calls becomes a 'word' and collection of such words becomes a 'phrase' [11]. Theoretically, if we have complete set of unique words (system call patterns) and unique phrases, then we can check validity of the phrases generated after execution of program. But for complex programs it is tedious task to cover all possible paths of execution. Also patterns generated must be able to cover entire normal profile. This problem is solved using variable length patterns. Also in case of incomplete training hamming distance used for forming approximate pattern.

B. Need of Accessing Rarity Index:

According to Warrender et. al [6] rare sequences are suspicious, but deciding rarity using frequency measure is not useful. The proposed system uses rarity index measure that depends on number of training traces of program and number of training traces containing specified pattern. Rarity index is calculated as follows,

$$\delta(C;U) = \begin{cases} -\frac{1}{U}\sum_{u=1}^{U} I_{Ru}, & \text{if } \sum_{u=1}^{U} I_{Ru}(C) \geq 0 \\ 1, & \text{if } \sum_{u=1}^{U} I_{Ru}(C) = 0 \end{cases}$$

Where I is indicator function

$$\delta(I_{Ru}(C)) = \begin{cases} 1, & \text{if C is one of the subsequence of } R_u \\ 0, & \text{if C is not one of the subsequence of } R_u \end{cases}$$

According to this model each sequence rarity is in between -1 to 1. If sequence is not present in training samples then it gets value 1. Otherwise rarity index value remains in between -1 to 0.The smaller the (C) is the more training traces contains the sequence and less abnormal the sequence is. Negative rarity index indicates that the sequence has been seen in training traces while positive sequence indicates that the sequence is new and can be abnormal [8].

IV. IMPLEMENTATION

Architecture of proposed system is shown in figure 4.1. The proposed system functions in two phases, training and testing. At training time normal system call traces of program to be monitored are given to system. From these training samples maximal patterns are extracted then, phrases seen in normal training data is built with their rarity index. In testing phase, test traces are given for anomaly detection. The explanation of functioning is given below

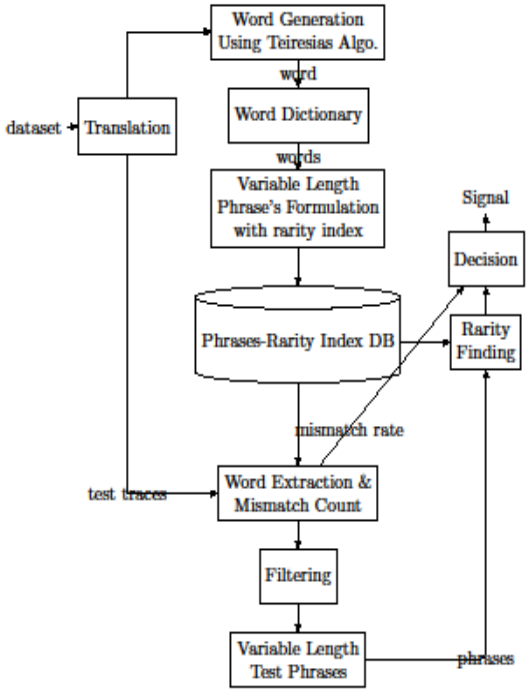

Fig. 1: System Architecture

A. Pattern Extraction:

Assume m, l as maximum, minimum pattern length to be extracted, T = $\{t_1,t_2,\ldots t_n\}$ where $t_1,t_2,\ldots t_n$ are training traces, w is sliding window where, l <= |w| <= m, . Initially set |w| = l and slide window across all the traces extracting all patterns of length |w|. Then store these extracted patterns with their occurrence count. If w<=m then increases |w| = |w| + 1 and do same procedure. Proposed system using Teiresias algorithm to find maximal patterns. According to the definition p is pattern if |p| >= 2, where |p| is length of pattern and (count of p) >= 2. Then Assume p and q as patterns, where |p|= |q|-1. A pattern p is maximal only if, p is not a subsequence of any other pattern q whose count is greater than that of p. If p is not maximal then delete p and repeat it for all patterns. B.

B. Formation of Phrases with Rarity:

Necessity of formation of phrases is that it stores information about relationship pattern. If phrases formed are popular, then it means that mismatches occurred while forming given phrase must be low. Consequently rare phrases formed can be result of incomplete training and in such case mismatch rate should be high. Formation of phrases is carried out by doing additional pass over training samples. Firstly adjacent maximal patterns in each training traces is extracted. By sliding window of different length over these extracted consequent maximal patterns (words) we get phrases of different length. Normally phrases of length 1 to 4 are extracted, because longer phrases do not represent true nature of training data. This step is equivalent to algorithm 5.3.traces.

C. Patterns Matching:

Initially one system call s is extracted from test trace T. After extraction of system call one of the condition holds,

1) If there are no matching patterns then considers it as mismatch.

2) There are set of patterns starting with s. In such case pattern matching continues and pattern with longest match is selected. If there are no patterns that can be found, in such case start position is marked as mismatch and pattern matching continues from skipped event. The skipped sequence is monitored for possible pattern that can be generated (approximate pattern matching).
3) If there is any mismatched sequence having length>=2 then hamming distance is used to replace mismatched pattern. The matched patterns are temporarily stored in sequential manner with their rarity index, mismatches occurred while constructing them. The phrases are built from these extracted patterns.

D. Decision Engine:

The decision is based on the number of mismatches occurred while forming given phrase and average rarity of phrase.

V. ALGORITHM

A. Algorithm for Fixed-Length Pattern Extraction:

```
procedure GETFIXEDWORDS(syscalltraces)
    for all traces do
counter = 1
fixedWordDic←empty
        for system call in traces do
            word = systemcall + nextcountercalls
            if word is in fixedWordDic then
                increment count of word
            else
                add word to fixedWordDic
            end if
            counter = counter + 1
        end for
      end for
   return fixedWordDic
end procedure
```

B. Algorithm for Maximal Pattern Extraction:

```
procedure GETVARIABLEWORDS(fixedWordDic)
for all words in fixedWordDic do
    currWord = word
    if currWord is subString of anyotherWord then
            currWordCnt ←count of  currWordCnt
            otherWordCnt ←count of otherWordCnt
            if currWordCnt <= otherWordCnt then
               delete currWord from fixedWordDic
            end if
    end if
end for
    varWordDic←fixedWordDic
    return varWordDic
end procedure
```

C. Algorithm Phrase Generation:

```
procedure GETPHRASES(syscalltraces,initWordDic)
for all traces do
   counter←2
   wordPos←0
    for words in trace do
        word = +nextcounterwords
        if phrase in phraseDictionary then
            Increment count of Phrase
    else
            Add phrase to phraseDictionary
    end if
      counter = counter + 1
    if counter > MAXW then
           exit
    end if
 end for
```

VI. EXPERIMENTAL RESULTS

Evaluation Proposed system is evaluated using UNM intrusion detection dataset. UNM dataset contains live normal data for different programs. Also different intrusive sequences including buffer overflow, trojan programs, denial of service attack [6]. For each process there is mapping file and normal as well as intrusive data given in different file. Each line of normal and intrusive file consist of 2 fields, PID and system call number and there can be multiple system call traces in one file [6]. The mapping of system call to their names can be done using mapping file. The system call traces for process ps, login, ftp, lpr extracted. Table 5.1 gives information about data extracted for each process.

Process Name	Total Normal Traces Extracted	Number of Normal Training Traces	Number of Test Traces	
			Normal	Attack
Login	9	4	5	12
ftp	7	4	3	5
Lpr	5	3	2	1001
ps	24	20	4	26

Table 1: Extracted Processes

For process ps and login intrusive sequence is Trojan attack that allows to get 'back-door' entry to system. The lpr program there is single lprcp symbolic link intrusion that consists of 1001 print jobs. Detection of anomaly in any of these 1001 traces considered as successful detection of intrusion [5][6].

Before extracting variable length words pre-processing technique was applied over dataset to remove redundant system call. For each program maximal patterns of length 2 - 6 are extracted and phrases of length 2 - 4 are formed. Table 2 shows word and phrase dictionary size statistics,

Process Name	Variable Length Word Dictionary Size (MAX length 6)	Phrase Dictionary Size (1 to 6)
Login	286	582
ftp	357	1728
lpr	154	259
ps	117	405

Table 2: Information of Dictionary Size

To test the performance of system, the test traces include normal and attack traces of each process. Each trace is selected and we note mismatches occurred in trace locally and generate phrases of length 1 to 6 from these extracted words with corresponding rarity-index noted of each phrase. The local average rarity-index of phrases with mismatch

count is used for decision purpose. The TABLE 3 shows the thresholds chosen for particular process.

A. Experiment 1:

This experiment is carried out to validate that rare sequences are anomalous and test combined use of rarity index of patterns and mismatch rate occurred while given phrase. The phrase of length 1 is used and maximum mismatches allowed while forming given phrase is given as threshold. According to hypotheses if phrase is rare then mismatch rate will go high, so combined use of rarity with mismatch can be used for anomaly detection. Also this experiment does not use hamming distance for replacing mismatched pattern with approximate one. Following graph shows single anomalous system call trace for process ps with rarity and mismatches occurred. The positive Y-axis consist of number of mismatches occurred and negative Y-axis shows rarity index, while X-axis consist of number of phrases extracted along trace. The detection rate and FPR for processes ps, login, lpr is shown in figure 5.2.

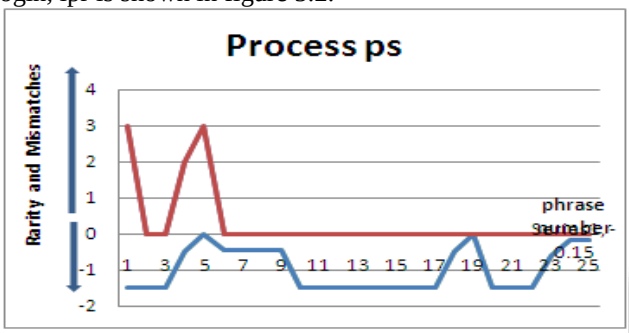

Fig. 2: Relation between Rarity and Mismatch

Process Name	Threshold on Mismatch Rate	Threshold on Rarity Index	Detection Rate	FPR
Login	3	-0.2	98.33%	4%
ftp	12	-0.15	93.00%	2.6%
lpr	3	-0.5	100%	0%
ps	9	-0.15	96.18%	4.30 %

Table 3: Experiment 1 Result

B. Experiment 2:

This experiment is carried out to test effect of choosing longer phrase length over detection rate. The experimental settings are similar as shown in table 5.1. In this experiment patterns are extracted from test trace and phrase of length 2 is constructed. The rarity of each phrase of length 2, is summed up with rarity its participant phrases of length 1 and average is taken. The rarity of participant's phrases is determined so as to find support count. The results are shown in table,

Process Name	Threshold on Mismatch Rate	Threshold on Rarity Index	Detection Rate	FPR
login	3	-0.4	97.45%	3.1%
ftp	12	-0.15	93.00%	2.6%
lpr	4	0.3	100%	1 %
ps	9	-0.15	96.18%	4.30%

Table 4: Experiment 1 Result

VII. Conclusion

From experimental results it is validated there is relation between rare sequences and anomaly. We also tested use rarity index and mismatch count for anomaly detection. Also result shows that use of relation for anomaly detection is beneficial. It has been observed that patterns generated using Teiresias algorithm is incomplete because it looses some patterns whose count=1 even after it may be result of incomplete training problem and it needs to be corrected. The initial training time is acceptable.

References

[1] John McHugh, Alan Christie, and Julia Allen, "The Role of Intrusion Detection Systems, IEEE SOFTWARE,SEP 2000.
[2] Mehdi Bahrami and Mohammad Bahrami, "An overview to Software Architecture in Intrusion Detection System", Soft Computing And Software Engineering (JSCSE),2011.
[3] Herve Debar, "An Introduction to Intrusion-Detection Systems", IBM Research, 2011.
[4] S. Forrest,S. A. Hofmeyr and A. SoMayaji, "A sense of self for Unix Processes", IEEE Symposium, May 1996.
[5] S. Forrest,S.A. Hofmeyr and A. SoMayaji, "Intrusion Detection Using Sequences of System Calls", IEEE Symposium, May 1996.
[6] C. Warrender, S. Forrest, and B. Pearlmutter,"Detecting intrusions using system calls: alternative data models", Proceedings of the 1999 IEEE Symposium,1999.
[7] John Andreas Wespi and Herv Debar, "An intrusion detection system based on the teiresias pattern discovery algorithm", Proceedings of EICAR, 1998.
[8] Wen-Hu Ju and Yehuda Vardi, "Profiling Unix Users And Processes Based On Rarity of Occurrences Statistics with Applications to Computer Intrusion Detection", Fourth Aerospace Computer Security Applications Conference, October 1988.
[9] UNM intrusion detection dataset available at http://www.cs.unm.edu/~immsec/systemcalls.htm
[10] John Y. Liao, V. R. Vemuri, "Use of K-Nearest neighbor classifier for intrusion detection", Computer Security,2002.
[11] G. Creech and J. Hu. ,"A Semantic Approach to Host-based Intrusion Detection Systems Using Contiguous and Dis-contiguous System Call Patterns", IEEE Transactions on Computers, 2014.
[12] Ye Du, Ruhui Zhang, and Youyan Guo, "A Useful Anomaly Intrusion Detection Method Using Variable-length Patterns and Average Hamming Distance", Journal of Computers, Aug 2010.
[13] Firkhan Ali,and Yee Yong Len, "Development of Host Based Intrusion Detection System for Log Files", IEEE Symposium on Business,2011.
[14] Federico Maggi and Matteo Matteucci," Detecting Intrusions through System Call Sequence and Argument Analysis", IEEE Transaction On Dependable And Secure Computing, Aug 2007.

