International Journal of Computer Applications (0975 – 8887)
Volume 45– No.19, May 2012

Intrusion Detection System using Log Files and Reinforcement Learning

Bhagyashree Deokar, Ambarish Hazarnis
Department of Computer Engineering
K. J. Somaiya College of Engineering,
Mumbai, India

ABSTRACT

World Wide Web is widely accessed by people for accessing services, social networking and so on. All these activities of users are traced in different types of log files. Hence, log files prove to be extremely useful in understanding user behavior, improving server performance, improving cache replacement policy, intrusion detection, etc. In this paper, we focus on the intrusion detection application of log files. By analyzing drawbacks and advantages of existing intrusion detection techniques, the paper proposes an intrusion detection system that attempts to minimize drawbacks of existing intrusion detection techniques, viz. false alarm rate and inability to detect unknown attacks. To accomplish this, association rule learning, reinforcement learning and log correlation techniques have been used collaboratively.

General Terms

Log files, Association rules

Keywords

Association Rule Learning, Log Correlation, Log files, Reinforcement learning, Intrusion Detection Systems

1. INTRODUCTION

1.1 Intrusion Detection Systems (IDS)

Intrusion detection is defined as, "The task of detecting and responding to the computer misuse by detecting unauthorized access to a computer network [1]." Intrusion detection system is the system that collects information from a variety of systems and network sources and then analyses the information for signs of intrusion and misuse [1]. IDS are used to supervise and assay the activities of the user and the system, examine policy violation by user, to recognize patterns of typical attack, analysis of system configuration, file integrity and abnormal activity patterns.

a. Characteristics of IDS

Some of the characteristics of good intrusion detection systems [2] are

1. *Timeliness*: Ability to detect the attack when it is occurring or quickly after that.
2. *High probability of detection*: The IDS must be capable of detecting significant types of attack.
3. *Low False alarm rate*: The IDS should have a low false alarm rate.
4. *Scalability*: Ability to work with wide network involving large number of components.
5. *Low a priori information*: The IDS should require little or no advance information about potential attacks and their methods.

b. Types

Intrusion detection is of two types as explained below:

Signature detection

Signature detection is the method to detect attack patterns based on the similarity to the defined signatures of known attack.

Anomaly detection

In anomaly detection, normal behavior of the system or user is defined and when deviation from normal behavior is detected, attack is identified. There are very high chances of false alarms in case of anomaly detection.

1.2 Log Files for IDS

Log files contribute for increasing the strength of existing security measures by providing information about suspicious user behavior and the intrusion causing systems. However, existing intrusion detection systems using log files still face problems in detecting unknown attacks. Log correlation and other data mining concepts of clustering and association rules can be used to subsidize this problem.

1.3 Paper Structure

In this paper, section 2 states different types of log files based on their locations. Section 3 deals with concept of log correlation for efficient analysis of log files for intrusion detection. Bottom up and top down approach of log correlation are also explained. Section 4 gives general approach of existing intrusion detection system. Section 5 analyses the pros and cons of existing intrusion detection techniques. Section 6 proposes Intrusion Detection System using Log Files & Reinforcement Learning to minimize drawbacks of existing intrusion detection techniques and to detect known as well as unknown attack.

2. TYPES OF LOG FILES BASED ON THEIR LOCATION

2.1 Server side log files

A web server log file is a text file that consists of activities performed on the web server. These log files collect and store following types of data: Date, Time, Client IP Address, Referrer, User Agent, Service Name, Server Name, Server IP, etc. [5]. Hence, these files are useful for analyzing server performance and search engine optimization.

2.2 Client side log files:

Client side log files contain data collected from client side by the execution of a script on the client side machine. This script is send by the server along with the web document. Page tagging is one of the most widely used methods for client side data collection [8].

2.3 Proxy side log files

Proxy server is present between the client machine and server machine. It reduces burden of web server by serving the request for the pages which are available with it. It eliminates the drawback of web server's log files which allows analyzing only the behavior of user for one particular site only [6].

2.4 Firewall side log files

Firewall logs only those events which are denied by the system. As such, examination of firewall log files is vital for intrusion detection. It is also necessary to measure the strength of security measures implemented at firewall using Firewall auditing [4].

2.5 Network side log files

Network side log files can be obtained from network components such as from network firewall, routers and packet filters for analysis. It has an edge over other log files because it does not involve legal issues about violation of privacy of user's data [12].

2.6 System side log files

System side log files manage the information generated by the kernel and the system utilities. Detailed information about the activities of operating system is captured in system log files. Further, they emancipate the programmer from task of writing log files.

3. LOG CORRELATION

All the activities of the user about accesses to various applications are stored as a separate record in the log files. Hence**,** log files are an invaluable resource of information for attack detection. Tracking of each and every activity in the different log files allows us to gain almost complete information about all aspects of attack and its behavior. Such knowledge helps in identifying new as well as existing types of attack. However, when an attack occurs, it is necessary to check the appropriate affected areas. Many a times data in log files extracted from only one source results in false alarms [3]. Moreover, there is a high probability that if attack is collaborative, then it will not be detected by relying on log information provided by only one source. As such correlating data from log files of multiple sources helps us in achieving effective decision about detection of attack. It also facilitates reduction of false alarms due to the knowledge from various sources. Two different approaches based on log correlation as stated in [3] for attack detection are:

3.1 Top down approach

Top down approach initiates its processing when intrusion detection system notifies it about the detected attack. It analyses behavior of detected attack and identifies locations of relevant log files which are affected. By correlating and analyzing these affected log files, signature of the attack is prepared and can be used to identify same type of attack in the future. Analysis of affected log files is an important factor for deciding at which locations preventive measures should be taken.

3.2 Bottom up approach

Bottom up approach initiates its processing by analyzing the log records soon after they are logged in the log files. After observation of anomalies in one log file, relevant log files are analyzed and correlated. This facilitates the detection of unknown attacks. However, analysis of log files is time consuming because of its millions of entries. Data mining and machine learning algorithms can be used for the analysis of log files at a faster rate.

4. GENERALIZED APPROACH

General approach followed by existing intrusion detection systems is as shown in figure (1).
Step 1: Pre-processing performs the extraction of necessary fields in the log files along with the filtering of noise.
Step 2: Log correlation determines which log files should be correlated.
Step 3: Pattern analysis of correlated log files compares the log file scenario with knowledgebase of normal behavior. If required, pattern analysis is also implemented with respect to the signature database.
Step 4: The event is categorized as normal or attack or anomaly depending on the analysis performed in above step.

5. PROS AND CONS OF EXISTING ID TECHNIQUES

Most widely used intrusion detection techniques are anomaly and signature detection.

5.1 Anomaly Detection

Anomaly detection has advantage of detecting unknown attacks as it works on the basis of differences in the normal behavior of user and user's current behavior [9]. If that difference is large then, an attack is detected. Anomaly detection, however, has its own limitations as described below:

- There are some activities which users do not perform on the regular basis. Those activities may be categorized by system as an anomaly, leading to false alarms.
- If intrusive activities are performed in a way that will imitate normal behavior, then anomaly detection system will not be able to detect such stealthy attacks [7].
- Selecting the right parameters of the log file is crucial for the process of anomaly detection; missing important parameters makes it difficult to distinguish attacks from normal activities. On the other hand, having unimportant intrusion related features could introduce "noise" into the models and thus, affect the performance of detection systems [7].

5.2 Signature Detection

Signature detection, also known as misuse detection, builds patterns of well-known attacks or weak spots of the system and uses these patterns for identifying intrusions [9]. Signature detection, thus, gives accurate results for the attacks whose signature is known and has a very low false alarm rate as compared to anomaly detection. It also gives high performance as compared to anomaly detection because of explicit knowledge about the attack. Signature detection, however, suffers from the following drawbacks:

- Signature detection fails when any new type of attack occurs.
- Signature detection is also not able to detect attack which has slight variation with the available signatures of attacks.
- Signature detection needs explicit knowledge about the attack scenario. Moreover, this knowledge building requires analysis by the humans which is time consuming and error prone [7].

Hence, to overcome the drawback of signature detection in detecting new attacks, the existing systems use it along with anomaly detection.

6. PROPOSED SYSTEM

The primary objectives of the proposed system are:

- The reduction of false alarm rate

- Improvement in the ability of the system to detect unknown attacks

To achieve these goals, concepts of log correlation, reinforcement learning and association rule learning are put to use.

6.1 Real-Time IDS

Intrusion detection system can be of two type i.e. real time IDS and interval based IDS [10]. Real time IDS perform online analysis of events when they are occurring and thus, prevent the attack. Interval based IDS, on the other hand, works on periodic basis and thus, cannot help in preventing attacks while they are transpiring. Hence, we have adopted real-time IDS in our proposed approach. As a result, processing on logged event will start immediately as soon as it is recorded in the log files according to the Bottom up Approach of log correlation.

Figure 2 represents the architecture of the proposed intrusion detection system. It consists of Processing Units at different locations of log files as described in section 2 and a Central Unit.

6.2 Processing Unit (PU)

The Processing Units present at different locations of log files perform the task of capturing the newly logged event which is the initiating step of the entire process. Every Processing Unit consists of XML converter, Feature extractor, Comparator and Knowledge base of normal behavior. Content of knowledge base of normal behavior will vary depending on the location to which it belongs. Following part explains the individual components of Processing Unit:

6.2.1 Record of log event

It represents the new event traced in log file which is the main input source for the Processing Unit. Figure 3 represents the structure of log file.

6.2.2 XML converter

XML converter converts each new event traced in log file into XML format because of following advantages of XML over text files:

- XML files follow a structured format [11].
- XML format is more readable by machine [11].
- XML gives better performance.

Figure 4 shows the XML format of log file.

6.2.3 Feature Extractor

Feature Extractor performs feature extraction from the XML format of log event. Feature extraction process is necessary to identify fields from the log event which have high information gain value and collect them for further processing.

6.2.4 Knowledge base of normal behavior

It consists of patterns of normal behavior that can occur in the log files.

6.2.5 Comparator

It is responsible for comparison of information extracted by Feature Extractor with the knowledge base of normal behavior. If comparator found match for the event, it should wait until a new event is logged in log file. If no match is found for the event, the comparator should then, hand over the logged event to the Central Comparator Unit after converting it to association rule format with the help of association rule converter.

6.3 Central Unit

Central Unit co-ordinates between PUs of log files at different locations for implementing log correlation in order to detect known, unknown and collaborative attacks. Central Unit consists of Association Rule Database, Central Comparator, Feedback Unit and Association Rule learner.

6.3.1 Association Rule Database

Initially, Association Rule Database consist of two tables viz. association rules for well-known attacks(ARKA) and association rules for unknown attacks (ARUA).

Association Rules for Well-Known Attacks (ARKA):

Association rules for known attack types are stored in the ARKA table in the form of events along with their respective log files. Consider the case when particular event E_1 containing anomalies is logged in log file L_1, consequently its next dependent event E_2 containing anomalies is logged in L_2 and E_3 in $L_2$and at last E_4 in L_3. Hence, association rules formed will be $E_1 (L_1) => E_2 (L_2)$, $E_2 (L_2) => E_3 (L_2)$ *and* $E_3 (L_2) => E_4 (L_3)$. These rules are stored in a structured format as shown in table 1. “Part 1” column of table represents the antecedents and “Part 2” column of table represents the precedents. Many other possible combinations of different association rules exist based on the attack type. When considerable amount of association rules are satisfied then, prediction of attack is possible.

Association Rules for Unknown Attacks(ARUA):

Association rules for unknown attacks are stored in the table ARUA. These association rules represent the different combinations of log files for correlation and are utilized to deal with events which are neither found in knowledge base of normal behavior nor in the ARKA table. In other words, in case of unknown attacks these rules aid in deciding which log files should be referred for correlation to confirm the subsequent anomalies and hence to detect attack. These types of rules are defined in the table shown below. Rule ID will uniquely identify each rule. Part 1 represents antecedent component of association rule. Part 2 represents precedent component of association rule. For example-If any abnormalities are present in Web server log file then Central Comparator will also check for anomalies in System log file as per the rule with Rule ID 1 specified in ARUA table 2 and further if system log files also contain anomalies then by Rule ID 3 Central Comparator will check firewall log files for anomalies. The rule checking can be continued until considerable amount of rules are satisfied to confirm an attack or no valid rule exists. ‘Strength’ column represents the effectiveness of rule. Rules which are having high strength value are checked earlier by Central Comparator.

6.3.2 Central Comparator

It is responsible for comparison of rules provided by different PUs with the association rules of known attacks stored in ARKA. If match is not found then, Central Comparator will refer the rules specified in ARUA table to determine the correlated files which are then, requested from the PUs. These files are then, scanned by the Central Comparator for anomalies or signs of attacks to send feedback to the Feedback Unit.

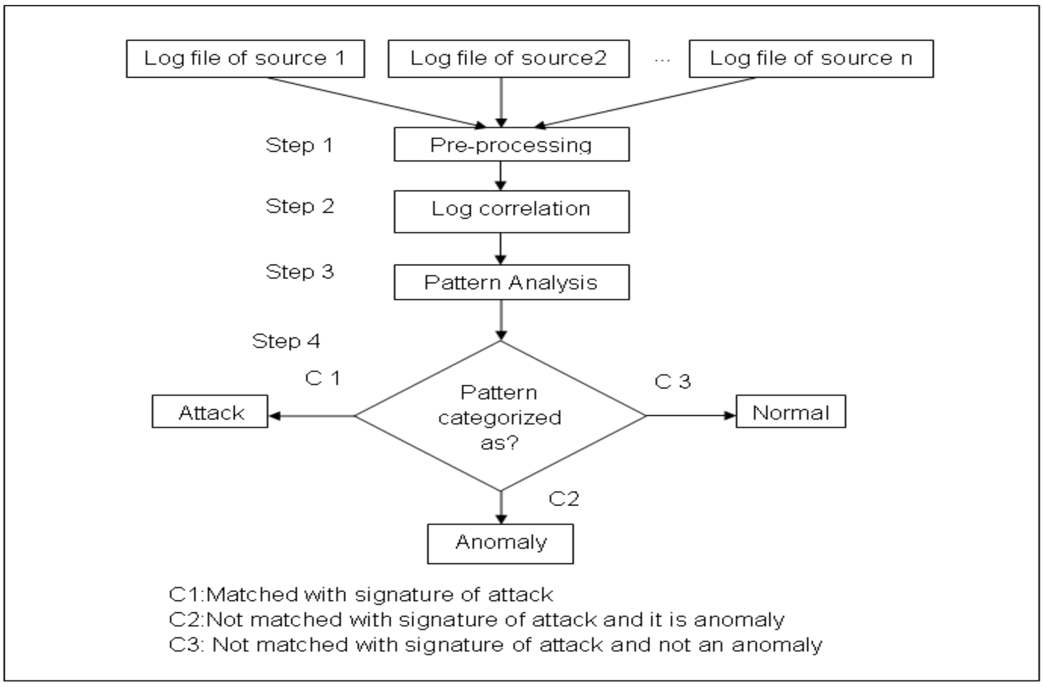


Fig 1: General Overview of existing intrusion detection system

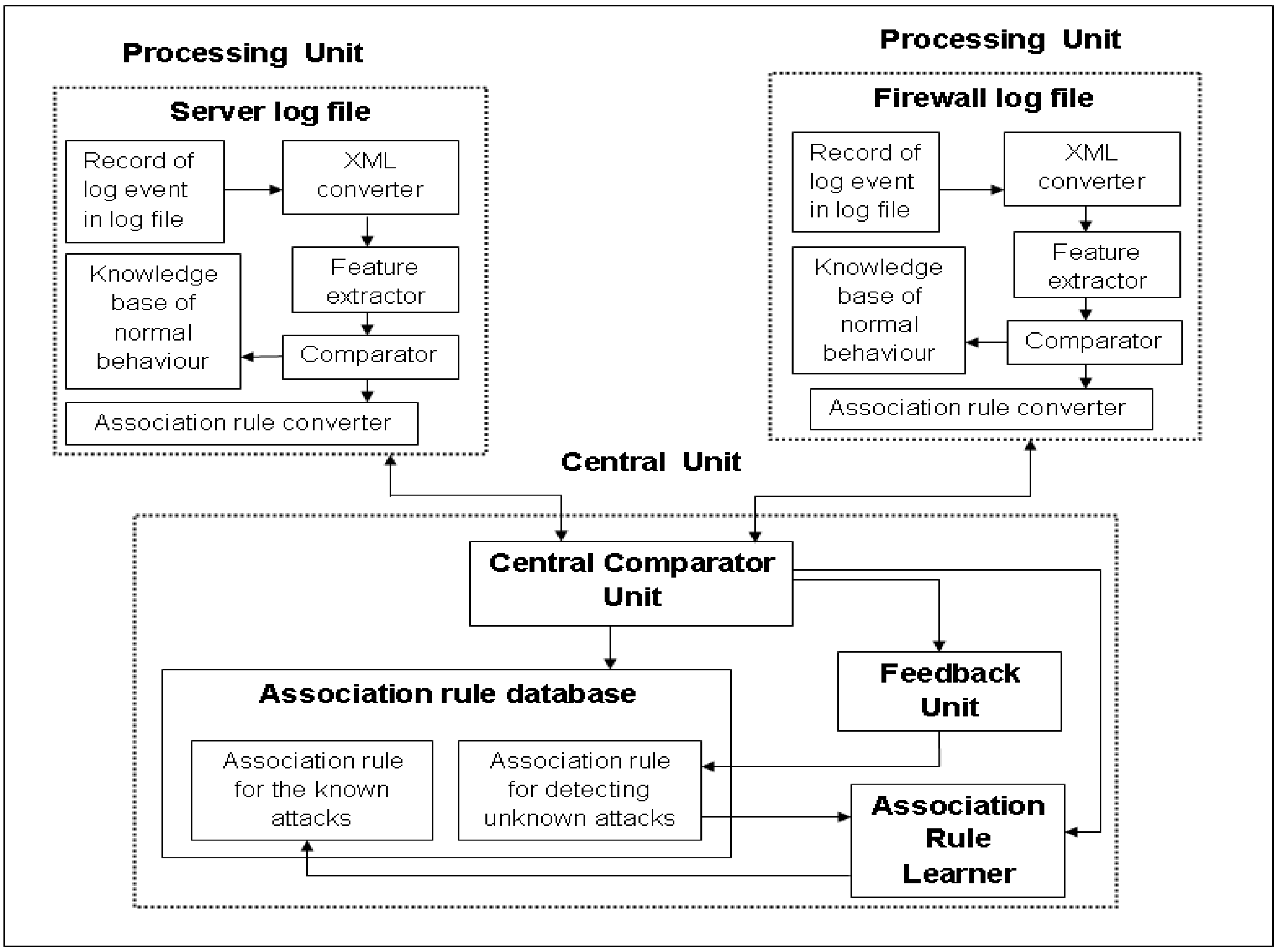
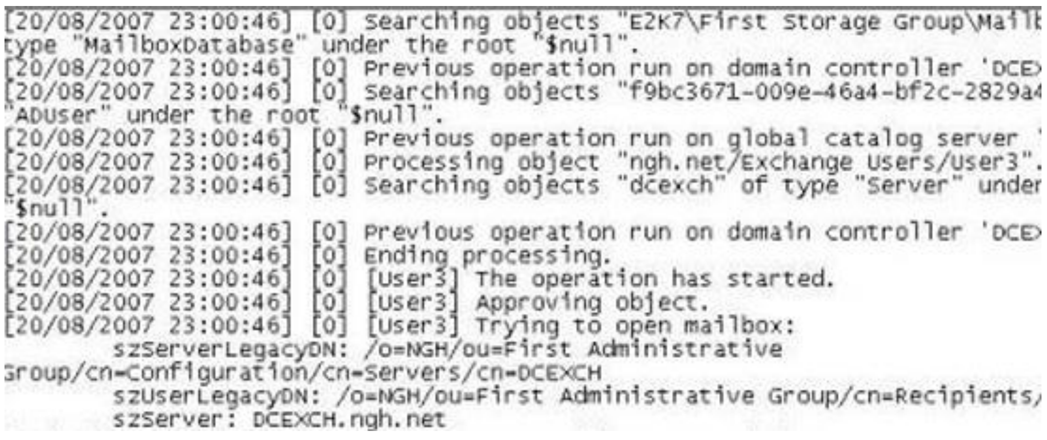
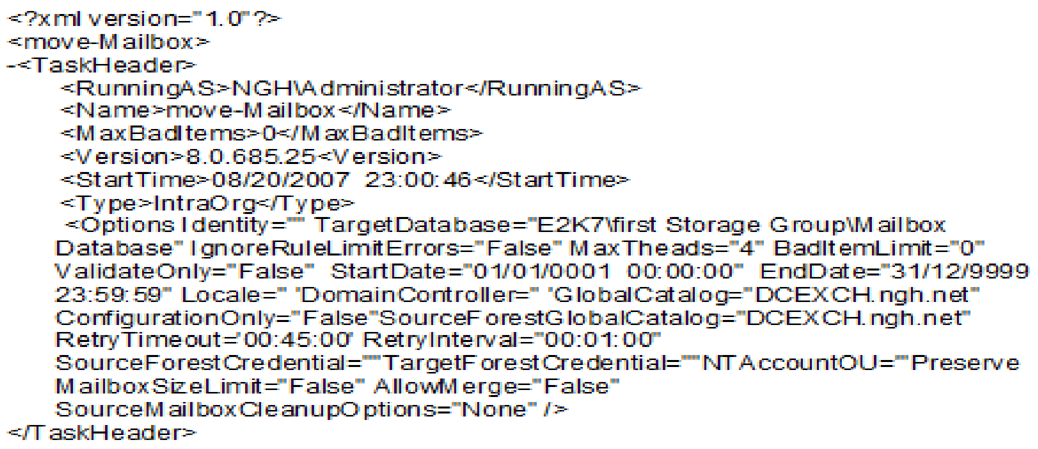

Processing Unit
Server log file
Record of log event in log file
XML converter
Knowledge base of normal behaviour
Feature extractor
Comparator
Association rule converter
Processing Unit
Firewall log file
Record of log event in log file
XML converter
Knowledge base of normal behaviour
Feature extractor
Comparator
Association rule converter
Central Unit
Central Comparator Unit
Association rule database
Association rule for the known attacks
Association rule for detecting unknown attacks
Feedback Unit
Association Rule Learner

Fig 2:Intrusion Detection System Using Log Files & Reinforcement Learning



```
[20/08/2007 23:00:46] [0] Searching objects "E2K7\First Storage Group\Mailb
type "MailboxDatabase" under the root "$null".
[20/08/2007 23:00:46] [0] Previous operation run on domain controller 'DCE>
[20/08/2007 23:00:46] [0] Searching objects "f9bc3671-009e-46a4-bf2c-2829a4
"ADUser" under the root "$null".
[20/08/2007 23:00:46] [0] Previous operation run on global catalog server '
[20/08/2007 23:00:46] [0] Processing object "ngh.net/Exchange Users/User3".
[20/08/2007 23:00:46] [0] Searching objects "dcexch" of type "Server" under
"$null".
[20/08/2007 23:00:46] [0] Previous operation run on domain controller 'DCE>
[20/08/2007 23:00:46] [0] Ending processing.
[20/08/2007 23:00:46] [0] [User3] The operation has started.
[20/08/2007 23:00:46] [0] [User3] Approving object.
[20/08/2007 23:00:46] [0] [User3] Trying to open mailbox:
        szServerLegacyDN: /o=NGH/ou=First Administrative
Group/cn=Configuration/cn=Servers/cn=DCEXCH
        szUserLegacyDN: /o=NGH/ou=First Administrative Group/cn=Recipients/
        szServer: DCEXCH.ngh.net
```

Fig 3: Log File



```
<?xml version="1.0"?>
<move-Mailbox>
-<TaskHeader>
    <RunningAS>NGH\Administrator</RunningAS>
    <Name>move-Mailbox</Name>
    <MaxBadItems>0</MaxBadItems>
    <Version>8.0.685.25<Version>
    <StartTime>08/20/2007  23:00:46</StartTime>
    <Type>IntraOrg</Type>
     <Options Identity="" TargetDatabase="E2K7\first Storage Group\Mailbox
   Database" IgnoreRuleLimitErrors="False" MaxTheads="4" BadItemLimit="0"
   ValidateOnly="False"  StartDate="01/01/0001  00:00:00"  EndDate="31/12/9999
   23:59:59" Locale=" 'DomainController=" 'GlobalCatalog="DCEXCH.ngh.net"
   ConfigurationOnly="False"SourceForestGlobalCatalog="DCEXCH.ngh.net"
   RetryTimeout='00:45:00' RetryInterval="00:01:00"
   SourceForestCredential=""TargetForestCredential=""NTAccountOU=""Preserve
   MailboxSizeLimit="False" AllowMerge="False"
   SourceMailboxCleanupOptions="None" />
</TaskHeader>
```

Fig 4: XML format of Log File

Table 1: ARKA

Part 1	Part 2
(E_1 (L_1))	(E_2 (L_2))
(E_2 (L_2))	(E_3 (L_2))
(E_3 (L_2))	(E_4 (L_3))
(E_1 (L_2))	(E_2 (L_1))

Table 2: ARUA

Rule ID	Part 1	Part 2	Strength
1	Web server log file	System log file	0
2	Network log file	Firewall log file	10
3	System log file	Firewall log file	0
4	Web server log file	Network log file	5

6.3.3 Feedback Unit (FU)

Depending on the Central Comparator's feedback to Feedback Unit, it will update the strength of the rules specified in ARUA by the concept of Reinforcement learning. If log file specified in a rule doesn't contain anomalies or any signs of attack then, FU will penalize the Central Comparator by decreasing strength of rule which is selected by it. On the other hand, if log file specified in rule contains anomalies or any signs of attack then, FU will award the Central Comparator by increasing strength of rule which is selected by it. By giving reward in terms of increased value of strength, it motivates the Central Comparator to choose the files which are more appropriate to search for traces of attack.

6.3.4 Association Rule Learner

It collects information about anomalies in log files from the Central Comparator. This information is used to create association rules for unknown attacks in same format as specified for ARKA table.

7. WORKING

1. *Start.*
2. *Set the matched rule count (Count) to zero. Threshold identifies the value of Count that confirms an attack. It can be set by the administrator of the system.*
3. *Check if the count of matched rules has crossed the threshold value.*
    a. *If yes, then, notify the administrator about the attack and Stop.*
    b. *If no, then, go to step 4.*
4. *Processing Unit checks whether any new event is logged in log file.*

   a. *If no new event is found then wait till a new event arrives.*
   b. *If a new event is found then, go to step 5.*
5. *XML converter converts the recorded event into XML file format.*
6. *Feature extraction is performed on this XML file to extract the required information from logged event.*
7. *Comparator will compare the log record features extracted in above step with the knowledge base of normal behaviour.*
8. *Convert the log entry into association rule format for comparing with rules in the Association Rule Database.*
9. *This rule is handed over to Central Comparator for processing the below steps.*
10. *Central Comparator will search ARKA table of Association Rule Database for match of association rule handed over to it in step 9.*
11. *Check whether a match is found.*
    a. *If the match is found in ARKA table then, go to step 12.*
    b. *If the match is not found in ARKA table then, go to step 15.*
12. *Perform Log Correlation as explained in Section 3 and Association Rule Database of Central Unit by checking for transitivity of current matched rule in the ARKA table.*
13. *Increase the count of Count based on the number of rules matching form step 12.*
14. *Go to step 3.*
15. *Central Comparator will search ARUA table of Association Rule Database to find the correlated files using the log-file's name as the antecedent. This log file is the one which contains the logged event from step 4.*
16. *Check whether a match is found.*
    a. *If the match is not found in ARUA table go to step 4 to process the subsequent log events.*
    b. *If the match is found in ARUA table go to step 17.*
17. *Select rule 'R' which is having the highest strength among the matched rules.*
18. *The descendant part of the rule which is also a log file name is requested from the respective processing unit Central Comparator will process this file for attack or anomaly detection.*
    a. *If no discrepancies are encountered go to step 19.*
    b. *If discrepancies are encountered go to step 21.*
19. *Central Comparator Unit will direct the Feedback Unit to decrease the strength of rule R.*
20. *Go to step 4.*
21. *Association Rule Learner will create a new rule in the format of ARKA table for the identified attack using the discrepancies found.*
22. *The rule created in step 8 and step 21 will be added to ARKA table.*
23. *Central Comparator Unit will direct the Feedback Unit to increase the strength of rule R.*
24. *Perform Log Correlation as explained in Section 3 and Association Rule Database of Central Unit by checking for transitivity of current matched rule in the ARKA table.*
25. *Increase the count of Count based on the number of rules matching from step 24.*
26. *Go to step 3.*

8. CONCLUSION AND FUTURE WORK

Our paper proposed an Intrusion Detection System which implements log correlation, reinforcement learning and association rule learning collaboratively to identify known and unknown attacks effectively. Use of Reinforcement Learning for intrusion detection helps to detect unknown attack by motivating comparator with the rewards to identify correct log files for confirmation of attack. It exemplifies the benefits of integrating various artificial intelligence techniques with the Intrusion Detection Systems. It also provides scope for advancements in efficient pattern matching algorithms for accurate results. Implementation of proposed system will be best way to understand real-time issues that are not possible to realize during designing phase of the proposed system.

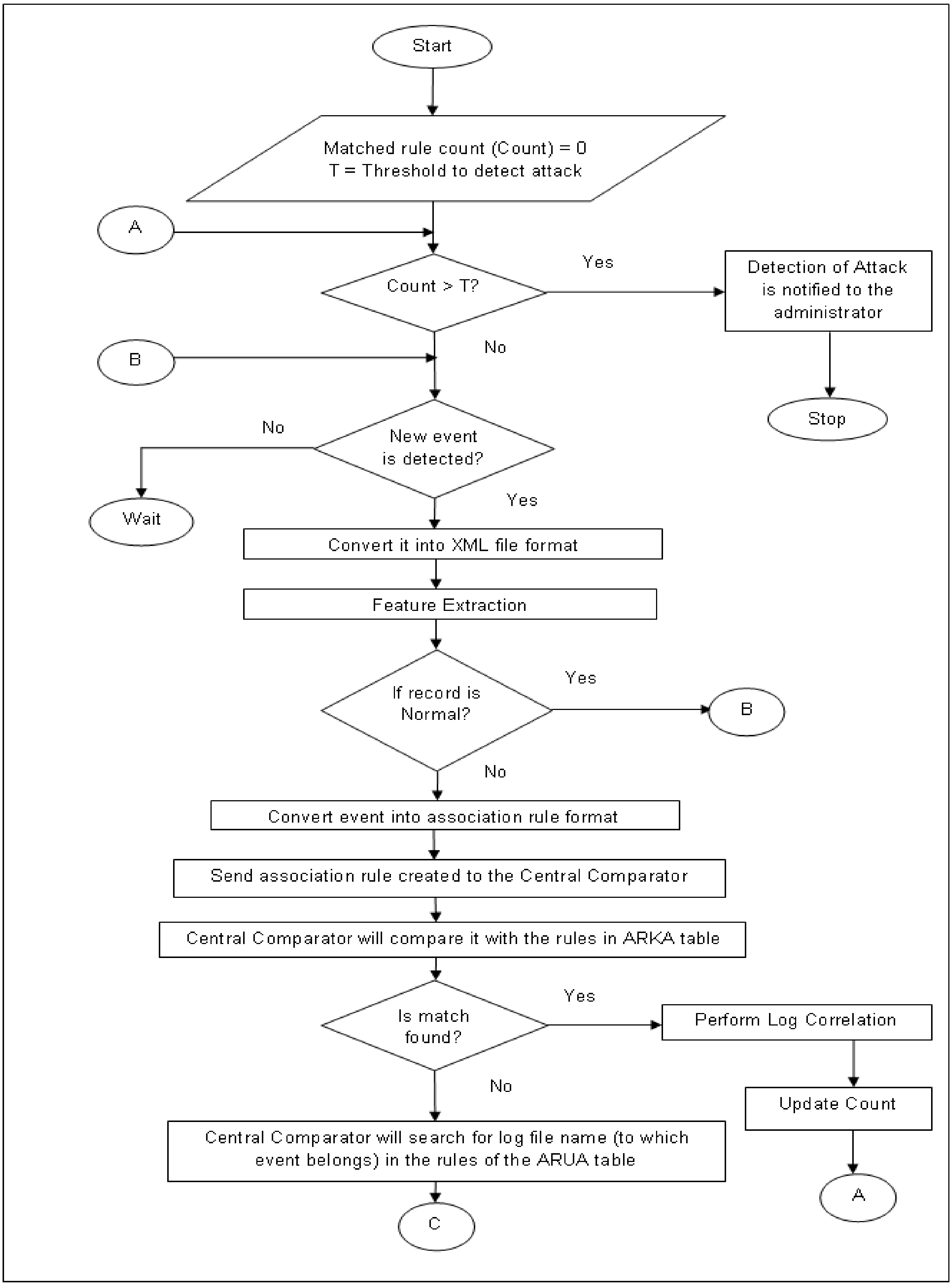

Start
Matched rule count (Count) = 0
T = Threshold to detect attack
A
Count > T?
Yes
Detection of Attack is notified to the administrator
Stop
No
B
New event is detected?
No
Wait
Yes
Convert it into XML file format
Feature Extraction
If record is Normal?
Yes
B
No
Convert event into association rule format
Send association rule created to the Central Comparator
Central Comparator will compare it with the rules in ARKA table
Is match found?
Yes
Perform Log Correlation
Update Count
A
No
Central Comparator will search for log file name (to which event belongs) in the rules of the ARUA table
C

Fig 5(a): Flowchart representing working of the system

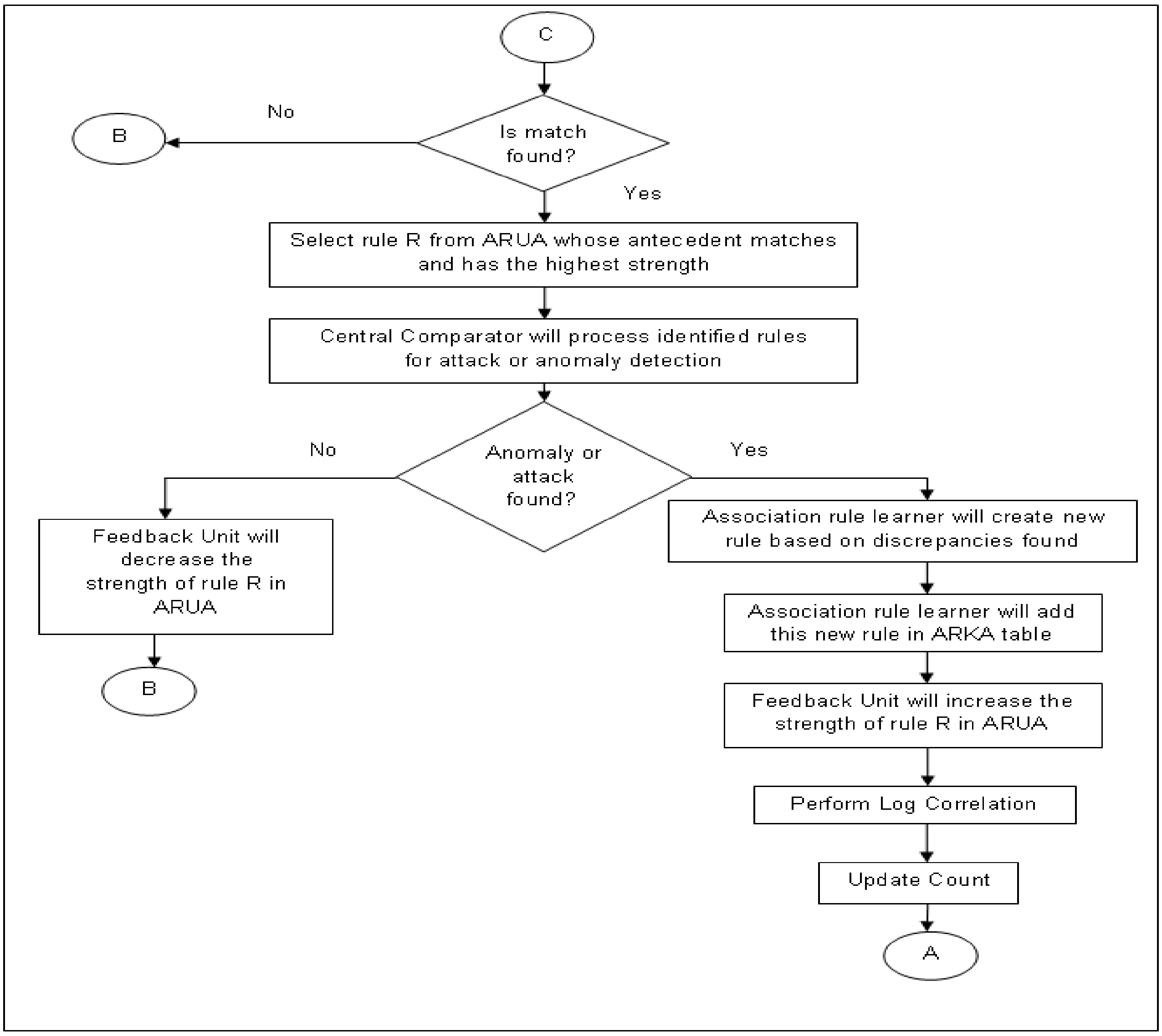

C
Is match found?
No
B
Yes
Select rule R from ARUA whose antecedent matches and has the highest strength
Central Comparator will process identified rules for attack or anomaly detection
Anomaly or attack found?
No
Yes
Feedback Unit will decrease the strength of rule R in ARUA
B
Association rule learner will create new rule based on discrepancies found
Association rule learner will add this new rule in ARKA table
Feedback Unit will increase the strength of rule R in ARUA
Perform Log Correlation
Update Count
A

Fig 5(b): Flowchart representing working of the system

9. REFERENCES

[1] Tesink, S. (2007). Improving Intrusion Detection Systems through Machine Learning. Group, (07).

[2] Cramer, M. L., Cannady, J., & Harrell, J. (1996).New Methods of Intrusion Detection using Control-Loop Measurement. Information Systems Security, 1-10.

[3] Abad, C., Taylor, J., & Rowe, K. (n.d.). Log Correlation for Intrusion Detection : A Proof of Concept Systems Research.

[4] Paper, W. (n.d.). Firewalls – Overview and Best Practices. Information Systems.

[5] <http://www.bruceclay.com/analytics/logfiles.htm> "It represents information about web server log files."

[6] Kerkhofs, J., &Pannemans, D. (2001). Web Usage Mining on Proxy Servers : A Case Study.

[7] Ning, P., & Carolina, N. (n.d.). Intrusion Detection Techniques. Bernoulli.

[8] Booth, D., & Jansen, B. J. (n.d.). A Review of Methodologies for Analyzing Websites, 141-162.

[9] Zhang, C., Zhang, G., & Sun, S. (2009). A Mixed Unsupervised Clustering-Based Intrusion Detection Model. 2009 Third International Conference on Genetic and Evolutionary Computing, 426-428. Ieee. doi:10.1109/WGEC.2009.72

[10] <http://www.windowsecurity.com/articles/ids-part2-classification-methods-techniques.html> "It represents classification, methods and techniques for intrusion detection system."

[11] Salama, S. E., I. Marie, M., El-Fangary, L. M., & K. Helmy, Y. (2011). Web Server Logs Preprocessing for Web Intrusion Detection. Computer and Information Science, 4(4), 123-133. doi:10.5539/cis.v4n4p123

[12] Brugger, S. T. (n.d.). Data Mining Methods for Network Intrusion Detection, V, 1-35.