Journal of Theoretical and Applied Information Technology
31st January 2018. Vol.96. No 2
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ALERT CORRELATION USING SUPPORT VECTOR MACHINE FOR MULTI INTRUSION DETECTION SYSTEMS

[1]XIAOYUN YE, [2]MYUNG-MOOK HAN
[1] Department of Computer Science, Gachon University, Seongnam, South Korea
[2] Department of Computer Science, Gachon University, Seongnam, South Korea
E-mail: [1]yxysun@gmail.com, [2]mmhan@gachon.ac.kr

**Corresponding author: Myung-Mook Han*

ABSTRACT

This paper presents a new alert correlation model for multiple intrusion detection systems. Based on the analysis of the complex relationship between the alert information of the intrusion detection system, an alert fusion model is proposed and used to alert correlation. The SVM algorithm has an advantage in the multi-dimensional classification, which can further reduce the influence of false positives and false negatives. The experimental results show that the alert fusion model has high accuracy and low false positive.

Keywords: *Alert Correlation, Intrusion Detection System(IDS), Support Vector Machine (SVM)*

1. INTRODUCTION

In the network intrusion detection and network alert collection always use different types of intrusion detection systems. Multiple intrusion detection systems have a greater advantage in the detection rate than single intrusion detection system, but we need face with the following disadvantages:

(1) Different intrusion detection system using different alert formats, which brings some difficulties to the alert analysis.

(2) A large number of duplicate alert data and redundancy data is not conducive to our analysis.

(3) Different intrusion detection systems have their limitations. This disadvantage can make a large of false positives and false negative.

(4) Some intrusion behaviors are related to each other and need to be analyzed.

Data fusion was first used in military affairs. JDL (Joint Directors of Laboratories) data fusion working group set up by the US Department of Defense proposed a general data fusion model – JDL model [1][2]. Bass proposed an intrusion detection data fusion model based on JDL [3]. This model interprets the distributed intrusion detection task as a synthesis problem of multiple sensor data under the hierarchical model. The fusion of intrusion detection data can be understood as several levels of the data extracted. In this hierarchical model, the intrusion detection data source from the data (Data) to information (Information) and then to knowledge (Knowledge) three logical abstraction level. This model provides a good idea for the application of data fusion in intrusion detection, but only proposes the functional level and the processing function requirements that the layers should meet. The paper does not propose a concrete implementation scheme.

SVM algorithm has good performance on a small sample, using this advantage in our model can correct identification and classification different type of alert information for alert fusion.

To overcome these disadvantages, this paper proposes an alert information fusion model based on the research of data fusion technology. It can deal with the alert flow and integrate the alerts from heterogeneous intrusion detection system according to the complex relationship among alert information, and use SVM algorithm to identify the different alert information in the same attack scene. Base on this model we present the experimental results.

The remainder of this paper is structured as follows: Section 2 presents the description of our related works. We will introduce the proposed model in detail, and Section 3 presents the description of our detection model. Section 4 presents the experimental results. Finally, Section 5 concludes and outlines future work.

2. RELATED WORK

The complex relationship between intrusion events determines that there will be complex relationships between intrusion detection system alarms, and different fusion methods are adopted according to the different needs of their relationship. There are three types of relationship: temporal relations, concurrency relations, and synergistic relations.

(1) Temporal relations: alarms that are triggered by an intrusion event that satisfies a temporal relationship can be considered to satisfy a temporal relationship.

(2) Concurrency relations: alarms that are triggered in the same period. That alarms' relationship is called concurrency relationship.

(3) Synergistic relations: There are multiple attacks from different attackers or attack sources. There is cooperation between attacks to achieve some attack intention. There are synergies between the alerts triggered by the attacks. These attacks or concurrency for a common goal, or the existence of a sequence and dependencies.

According to the above characteristics, we learn from the general process of JDI model shown in Fig.1.

But this model is not complete; it can't be applied to the actual information fusion. Such as the process of merge single-to-multiple alerts and merge multiple-to-single alerts, the same source IP may contain multiple attack information, if the attackers use many meaningless attacks to hide the true attack, we can't find them. Intrusion detection systems also have many false positive, so we need to handle that carefully. That disadvantages can make any help for our next analysis. We need to make some change for the model.

In machine learning, support vector machines (SVMs, also support vector networks [4]) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked for belonging to one of two categories; an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

We use some open source intrusion detection system. Such as Snort IDS, Bro IDS, and OSSEC.

Snort is a free and open source network intrusion prevention system (NIPS) and network intrusion detection system (NIDS) [5] created by Martin Roesch in 1998 [6]. Snort is now developed by Sourcefire, of which Roesch is the founder and CTO [7], and which has been owned by Cisco since 2013[8][9].

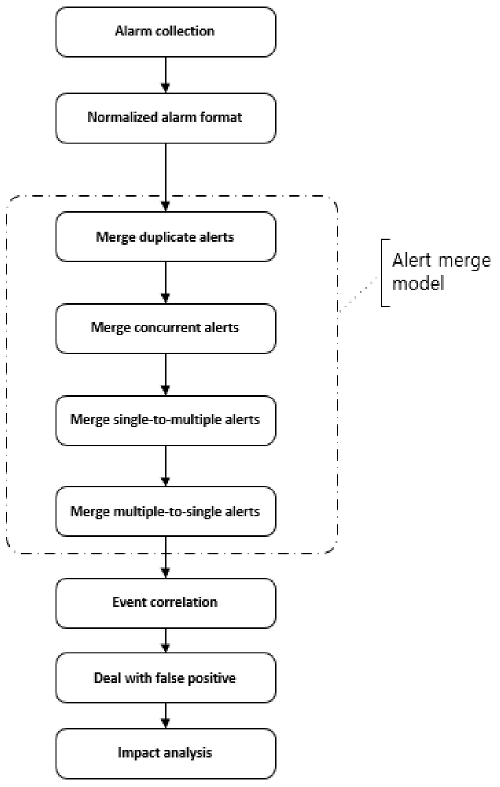

Figure 1: General Process Of JDI Model

Bro While focusing on network security monitoring, Bro provides a comprehensive platform for more general network traffic analysis as well. Well ground in more than 15 years of research, Bro has successfully bridged the traditional gap between academia and operations since its inception. Today, it is relied upon operationally in particular by many scientific environments for securing their cyberinfrastructure. Bro's user community includes major universities, research labs, supercomputing centers, and open-science communities [13].

OSSEC is a free, open-source host-based intrusion detection system (HIDS). It performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, time-based alerting, and active response. It provides intrusion detection for most operating systems, including Linux, OpenBSD, FreeBSD, OS X, Solaris, and Windows. OSSEC has a centralized, cross-platform architecture allowing multiple systems to be easily monitored and managed [14]. For example, when we handle the process of alert fusion, if the false positive is present in the alert data, the result of our fusion will be poor, and the result will be unreliable. We need a mechanism to reduce this condition. According to the characteristics of SVM, we use it to filter the false positive in the alert database.

The data set will be divided into two parts: training data set and test dataset. Training data set is an attack-free data set. In training data set we only need four features: SIP (source IP address), SPT (source port), DIP (destination IP address), DPT (destination port). Training dataset's structure is shown in below:

T = {SIP, SPT, DIP, DPT}

When we replay the test dataset, we got an alert database. Table.1 gives the attributes in the alert database and their meanings.

Our proposed model uses SVM algorithm to make up for the disadvantage of the model. We will talk about that in next chapter. Every record in the alert database is shown in the following format:

R = {ST, ET, IID, AID, AC, SIP, SPT, DIP, DPT, TS}

3. PROPOSED MODEL

The following describes the implementation of the main components of our alert fusion system based on our model shown in figure 2.

The processing flow is as follows:

(1) Replay the DARPA dataset with the multi intrusion detection systems (Snort, Bro, OSSEC).
(2) Normalized alert's format using Table.1.
(3) Training SVM Classifier with the attack-free dataset.
(4) Merge duplicate alerts.
(5) Merge concurrent alerts.
(6) Merge single-to-multiple alerts using the SVM classifier to filter the false positive alerts.
(7) Merge multiple-to-single alerts using the SVM classifier to filter the false positive alerts.
(8) Put all data into alert correlation database

The alert merge model process in our proposed model is shown below:

Merge duplicate alerts: In this step, we focused on processing of duplicate alerts from the same intrusion detection systems. First, we set a Time window (2 seconds) to process the alert data.

Define new alert A_{new}, Alert a (A_a) and Alert b (A_b) form database. Use the format we mentioned earlier. Then we can get:

$$A_{new} = \{ST_{new}, ET_{new}, IID_{new}, AID_{new}, AC_{new}, SIP_{new}, SPT_{new}, DIP_{new}, DPT_{new}, TS_{new}\}$$

$$A_a = \{ST_a, ET_a, IID_a, AID_a, AC_a, SIP_a, SPT_a, DIP_a, DPT_a, TS_a\}$$

$$A_b = \{ST_b, ET_b, IID_b, AID_b, AC_b, SIP_b, SPT_b, DIP_b, DPT_b, TS_b\}$$

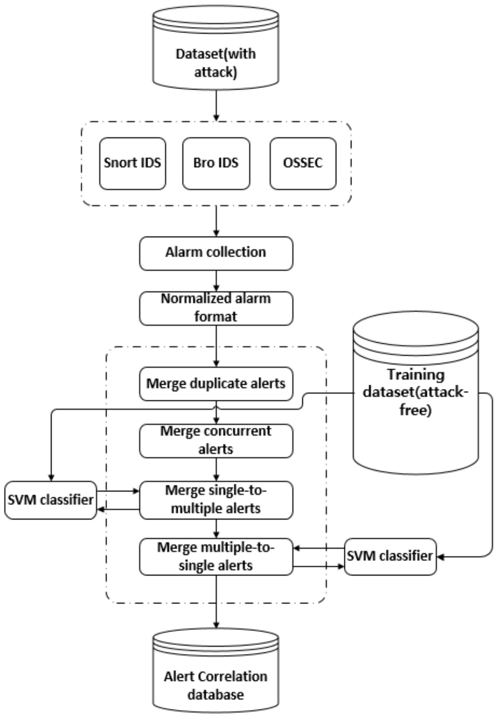

Figure.2 Proposed Model

We set the time window is 2 seconds, then we need to find an A_b that $ST_b < ST_a + 2s$ and has the same SIP, SRT, DIP, DPT as A_a. If we can find the A_b, then we can get A_{new} with:

$$\begin{aligned} ST_{new} &= ST_a \\ ET_{new} &= ET_b \\ AID_{new} &= AID_a \\ AC_{new} &= AC_a + AC_b \\ SIP_{new} &= SIP_a \\ SPT_{new} &= SPT_a \\ DIP_{new} &= DIP_a \\ DPT_{new} &= DPT_a \\ TS_{new} &= TS_a \end{aligned}$$

In this condition, all alerts came from the same intrusion detection system, so they all have the same IID, so we don't need to change that.

Merge concurrent alerts: We focused on processing of concurrent alerts from the different intrusion detection systems. In this step, we need to find out the alerts that alerted by the different intrusion detection systems at the same time. We don't need the SVM algorithm in this step. It can be easily processed with the script program.

Defined new alert A_{new}, Alert a (A_a) and Alert b (A_b). Use the format we mentioned earlier. Then we can get:

$$A_{new} = \{ST_{new}, ET_{new}, IID_{new}, AID_{new}, AC_{new}, SIP_{new}, SPT_{new}, DIP_{new}, DPT_{new}, TS_{new}\}$$

$$A_a = \{ST_a, ET_a, IID_a, AID_a, AC_a, SIP_a, SPT_a, DIP_a, DPT_a, TS_a\}$$

$$A_b = \{ST_b, ET_b, IID_b, AID_b, AC_b, SIP_b, SPT_b, DIP_b, DPT_b, TS_b\}$$

We set the time window is 2 seconds, then we need to find an A_b that $ST_b < ST_a + 2s$ and has the same SIP, SRT, DIP, DPT as A_a. If we can find the A_b, then we can get A_{new} with:

$$\begin{aligned} ST_{new} &= ST_a \\ ET_{new} &= ET_b \\ AID_{new} &= AID_a \\ IID_{new} &= IID_a + IID_b \\ AC_{new} &= AC_a + AC_b \\ SIP_{new} &= SIP_a \\ SPT_{new} &= SPT_a \\ DIP_{new} &= DIP_a \\ DPT_{new} &= DPT_a \\ TS_{new} &= TS_a \end{aligned}$$

In this condition, all alerts came from different intrusion detection system, so they don't have the same IID, so we need to change that. For example, $IID_{new} = IID_a + IID_b$ can be changed like $IID_{new} = \{IID_a, IID_b\}$, but not the sum of value.

Merge single-to-multiple alerts: In this part, we focused on processing of single-to-multiple alerts from the intrusion detection systems. First, we set a Time window (120 seconds); the connection for the alerts are at least two connections. This step and next step we use SVM algorithm to get the classification boundary, because, one scene may contain different types of attack. The experimental results will show the advantage for this improvement.

Define new alert A_{new}, Alert a (A_a) and Alert b (A_b) form database. And define a scene use the format we mentioned earlier. This phase is characterized by some alerts containing the same source IP address and port, but with different destination IP addresses and ports. Then, we can use scenes to merge these alert as an integrated part.

A single-to-multiple scene's structure is shown in below:

$$Scene_n = SIP, SPT \left\{ \begin{array}{l} attack\ type\ 1 \begin{Bmatrix} \{ST_1, ET_1, DIP_1, DPT_1\} \\ \{ST_2, ET_2, DIP_2, DPT_2\} \end{Bmatrix} \\ attack\ type\ 2 \begin{Bmatrix} \{ST_3, ET_3, DIP_3, DPT_3\} \\ \{ST_4, ET_4, DIP_4, DPT_4\} \end{Bmatrix} \\ \cdots \\ \cdots \\ attack\ type\ n \begin{Bmatrix} \{ST_{n-1}, ET_{n-1}, DIP_{n-1}, DPT_{n-1}\} \\ \{ST_n, ET_n, DIP_n, DPT_n\} \end{Bmatrix} \end{array} \right\}$$

We set the time window is 120 seconds, then we can get some scenes in this step. If we found another scene with the same scene structure and the same SIP and SPT, we can merge that scene. For example, there are two scenes: $Scene_a$ and $Scene_b$ $Scene_b$'s start time is later than $Scene_a$, they have the same structure.

$$Scene_a = SIP, SPT \left\{ \begin{array}{l} attack\ type\ 1 \begin{Bmatrix} \{ST_1, ET_1, DIP_1, DPT_1\} \\ \{ST_2, ET_2, DIP_2, DPT_2\} \end{Bmatrix} \\ attack\ type\ 2 \begin{Bmatrix} \{ST_3, ET_3, DIP_3, DPT_3\} \\ \{ST_4, ET_4, DIP_4, DPT_4\} \end{Bmatrix} \end{array} \right\}$$

$$Scene_b = SIP, SPT \left\{ \begin{array}{l} attack\ type\ 1 \begin{Bmatrix} \{ST_1, ET_1, DIP_1, DPT_1\} \\ \{ST_2, ET_2, DIP_2, DPT_2\} \end{Bmatrix} \\ attack\ type\ 2 \begin{Bmatrix} \{ST_3, ET_3, DIP_3, DPT_3\} \\ \{ST_4, ET_4, DIP_4, DPT_4\} \end{Bmatrix} \end{array} \right\}$$

$$Scene_{new} = SIP, SPT \left\{ \begin{array}{l} attack\ type\ 1 \begin{Bmatrix} \{ST_1, ET_1, DIP_1, DPT_1\} \\ \{ST_2, ET_2, DIP_2, DPT_2\} \end{Bmatrix} \\ attack\ type\ 2 \begin{Bmatrix} \{ST_3, ET_3, DIP_3, DPT_3\} \\ \{ST_4, ET_4, DIP_4, DPT_4\} \end{Bmatrix} \end{array} \right\}$$

We will merge $Scene_a$ and $Scene_b$. Before the merge process, we need use the SVM classifier to filter the false positive alerts. And then we need to get $Scene_{new}$ with:

$$Scene_{new}.attack\ type\ 1.ST_1 = Scene_a.attack\ type\ 1.ST_1$$
$$Scene_{new}.attack\ type\ 1.ST_2 = Scene_a.attack\ type\ 1.ST_2$$

$$Scene_{new}.attack\ type\ 2.ST_3 = Scene_a.attack\ type\ 2.ST_3$$
$$Scene_{new}.attack\ type\ 2.ST_4 = Scene_a.attack\ type\ 2.ST_4$$
$$Scene_{new}.attack\ type\ 1.ET_1 = Scene_b.attack\ type\ 1.ET_1$$
$$Scene_{new}.attack\ type\ 1.ET_2 = Scene_a.attack\ type\ 1.ET_2$$
$$Scene_{new}.attack\ type\ 2.ET_3 = Scene_a.attack\ type\ 2.ET_3$$
$$Scene_{new}.attack\ type\ 2.ET_4 = Scene_a.attack\ type\ 2.ET_4$$

Merge multiple-to-single alerts: In this part, we focused on processing of multiple-to-single alerts from the intrusion detection systems. First, we set a Time window (200 seconds); the connection for the alerts are at least 500 connections. This step we also used the SVM algorithm.
A multiple-to-single scene's structure is shown in below:

$$Scene_n = \left\{\begin{array}{c} attack\ type\ 1\left\{\begin{array}{l}\{ST_1, ET_1, SIP_1, SPT_1\}\\ \{ST_2, ET_2, SIP_2, SPT_2\}\end{array}\right\} \\ attack\ type\ 2\left\{\begin{array}{l}\{ST_3, ET_3, SIP_3, SPT_3\}\\ \{ST_4, ET_4, SIP_4, SPT_4\}\end{array}\right\} \\ \cdots \\ \cdots \\ attack\ type\ n\left\{\begin{array}{c}\{ST_{n-1}, ET_{n-1}, SIP_{n-1}, SPT_{n-1}\}\\ \{ST_n, ET_n, SIP_n, SPT_n\}\end{array}\right\}\end{array}\right\} DIP, DPT$$

We set the time window is 120 seconds, then we can get some scenes in this step. If we found another scene with the same scene structure and the same SIP and SPT, we can merge that scene. For example, there have two scenes: $Scene_a$ and $Scene_b$. $Scene_b$'s start time is later than $Scene_a$, they have same structure.

$$Scene_a = \left\{\begin{array}{l} attack\ type\ 1\left\{\begin{array}{l}\{ST_1, ET_1, SIP_1, SPT_1\}\\ \{ST_2, ET_2, SIP_2, SPT_2\}\end{array}\right\} \\ attack\ type\ 2\left\{\begin{array}{l}\{ST_3, ET_3, SIP_3, SPT_3\}\\ \{ST_4, ET_4, SIP_4, SPT_4\}\end{array}\right\}\end{array}\right\} DIP, DPT$$

$$Scene_b = \left\{\begin{array}{l} attack\ type\ 1\left\{\begin{array}{l}\{ST_1, ET_1, SIP_1, SPT_1\}\\ \{ST_2, ET_2, SIP_2, SPT_2\}\end{array}\right\} \\ attack\ type\ 2\left\{\begin{array}{l}\{ST_3, ET_3, SIP_3, SPT_3\}\\ \{ST_4, ET_4, SIP_4, SPT_4\}\end{array}\right\}\end{array}\right\} DIP, DPT$$

$$Scene_{new} = \left\{\begin{array}{l} attack\ type\ 1\left\{\begin{array}{l}\{ST_1, ET_1, SIP_1, SPT_1\}\\ \{ST_2, ET_2, SIP_2, SPT_2\}\end{array}\right\} \\ attack\ type\ 2\left\{\begin{array}{l}\{ST_3, ET_3, SIP_3, SPT_3\}\\ \{ST_4, ET_4, SIP_4, SPT_4\}\end{array}\right\}\end{array}\right\} DIP, DPT$$

We will merge $Scene_a$ and $Scene_b$. Before the merge process, we need use the SVM classifier to filter the false positive alerts. And then we need to get $Scene_{new}$ with:

$$Scene_{new}.attack\ type\ 1.ST_1 = Scene_a.attack\ type\ 1.ST_1$$
$$Scene_{new}.attack\ type\ 1.ST_2 = Scene_a.attack\ type\ 1.ST_2$$
$$Scene_{new}.attack\ type\ 2.ST_3 = Scene_a.attack\ type\ 2.ST_3$$
$$Scene_{new}.attack\ type\ 2.ST_4 = Scene_a.attack\ type\ 2.ST_4$$
$$Scene_{new}.attack\ type\ 1.ET_1 = Scene_b.attack\ type\ 1.ET_1$$
$$Scene_{new}.attack\ type\ 1.ET_2 = Scene_a.attack\ type\ 1.ET_2$$
$$Scene_{new}.attack\ type\ 2.ET_3 = Scene_a.attack\ type\ 2.ET_3$$
$$Scene_{new}.attack\ type\ 2.ET_4 = Scene_a.attack\ type\ 2.ET_4$$

In the step of merge single-to-multiple alerts and merge multiple-to-single alerts, we use SVM to make a classifier to divide the data into normal data and intrusion data. We will show the experiment results in the next chapter.

Table 1: Output Format.

Attributes name	Meaning
ST	Alert's start time
ET	Alert's end time
IID	ID of alerted IDS
AID	Alert's unique ID
AC	Number of alerts
SIP	Source IP
SPT	Source Port
DIP	Destination IP
DPT	Destination Port
TS	Alert's timestamp

3. EXPERIMEN

The computer environment is shown in Table.2.

Table.2 Computer Environment

CPU: Intel Core i5 2.5Ghz Memory: 8G OS: Ubuntu 14.04

This paper used the DARPA [17] as our training data set and test data set. DARPA data set's format is TCP dump (pcap). The advantage of this format is that it can be replayed. There is a software called TCP replay that can replay the data stream from that time. It is used for testing the ability of intrusion detection systems. The data are stored in a database. The available fields are shown in Table .1.

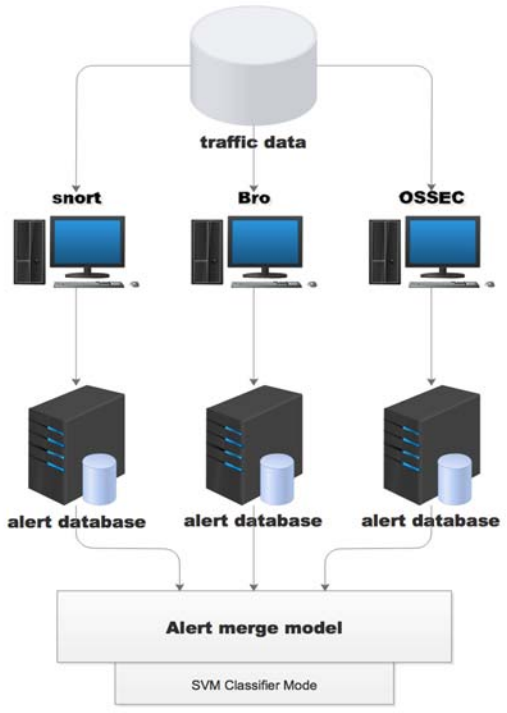

Figure.3 The Structure Of Computers And Database

We use three computers with the same hardware and system shown in Figure 3. Each computer has a different IDS environment and alert database. Use this method; we can get more accurate alert start time and end time.

The test dataset contains 888139 connection form DARPA dataset. We divided the data set into ten parts, take the average of the test results, remove the unreasonable maximum and minimum. This can accurately measure the system's detection ability. We used the general JDL model and the proposed model to process the same alerts data from the intrusion detection systems. The alert collection results are shown in Figure. 4. We got about 5716 alerts; there are 896 real attack alerts in this data. Also, we can get the 3754 normal data from the original model and 3528 normal data from the proposed model.

We can use formula 1 and 2 to calculate the fusion rate and the accuracy rate. The experimental results are shown in Table. 3.

There are two formulas, used to calculate the fusion rate and accuracy rate.

$$FR = \frac{BAC-AAC}{BAC} \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

$$AR = \frac{TP+TN}{TP+TN+FP+FN} \ldots\ldots\ldots\ldots\ldots (2)$$

We use FR and AR to represent fusion rate and the accuracy rate, use BAC to represent the alert count before fusion process, use AAC to represent the alert count after fusion process. The meaning of TP, TN, FP and FN are true positive, true negative, false positive, and false negative. RA means the real attack in the data set.

Table.3 Proposed Model Results

Original model	Proposed model
TP: 631	TP: 773
FP: 461	FP: 131
FN: 265	FN:123
TN: 3293	TN: 3397
AR: 84.39%	AR: 94.34%

Table.4 General Jdl Model Fusion Results

Alert Count		Alert Count After fusion process	Fusion rate(average)
Snort	2083	1357	76.26%
Bro	1795		
OSSEC	1838		
Sum: 5716			

Table.5 Proposed Model Fusion Results

Alert Count		Alert Count After fusion process	Fusion rate
Snort	2083	1027	82.03%
Bro	1795		
OSSEC	1838		
Sum: 5716			

The fusion results of the original model and the proposed model are shown in Table. 4 and Table. 5. We can compare the fusion ability of the two models in the Figure. 5 and Figure. 6. Obviously, our model's fusion ability is better than the original model.

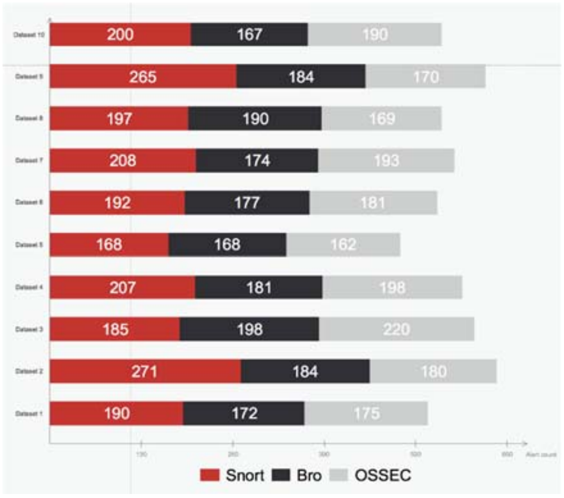

Figure.4 The Alert Collection Results (10 Data Sets)

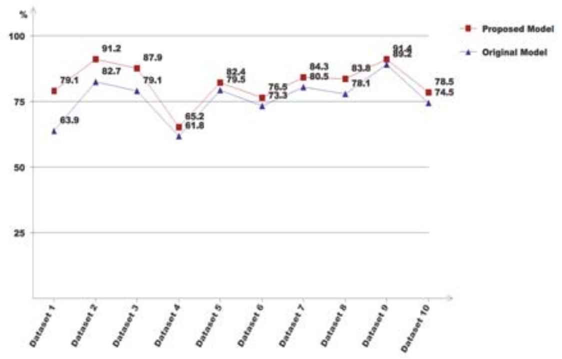

Figure.5 Alert Fusion Rate (10 Data Sets)

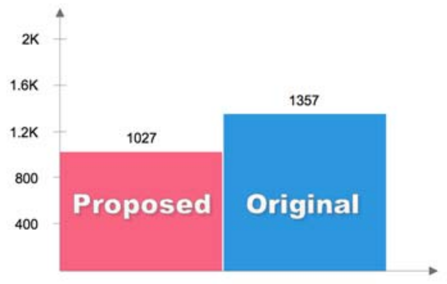

Figure.6 Alert Count After Fusion Process

Thanks to the SVM classifier, the number of false positives in our experimental results is significantly reduced. We can get the ten dataset's false positives shown in Figure. 7.

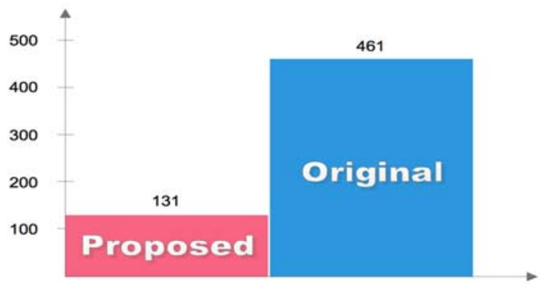

Figure.7 False Positive Count

The results show that our model for alert fusion improved the fusion rate from 76.26% to 82.03%, and also improved the accuracy rate from 84.39% to 94.34%. The results show our proposed model is better than the original model.

5. CONCLUSIONS

This paper presents a new alert correlation structure and uses SVM to filter and optimize the false positive of the IDS systems, which greatly improves the system's fusion ability and the detection accuracy. The simulation was done against DARPA data set and shows the performance of proposed approach with an improvement in false positive rate. We use the average value to measure the ability of the fusion system, which can more accurately reflect the actual situation of the system.

6. FUTURE WORKS

Our proposed model has a limitation: the SVM's process speed is too slow. Our future work will be to investigate other architectures to see if we can achieve better performance of more effective and faster alert correlation. Other future work will be to more thoroughly investigate the algorithms currently being used in our research. And use another classification algorithm to improve the accuracy of classification.

ACKNOWLEDGEMENT:

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (NRF-2015R1D1A1A01060874).

REFERENCES

[1] Alan N. Steinberg, Christopher L. Bowman, Franklin E. White. Revisions to the JDL Data Fusion Model[A]. In Sensor Fusion: Architectures, Algorithms, and Applications, Proceedings of the SPIE[C]. Vol. 3719, 1999.

[2] Llinas J, Bowman C, Rogova G, et al. Revisiting the JDL data fusion model II[R]. SPACE AND NAVAL WARFARE SYSTEMS COMMAND SAN DIEGO CA, 2004.

[3] Bass T. Multisensor data fusion for next generation distributed intrusion detection systems[J]. 1999.

[4] Chandola, V.; Banerjee, A.; Kumar, V. (2009). "Anomaly detection: A survey". ACM Computing Surveys. 41 (3): 1–58.

[5] Helman, Paul, Liepins, Gunar, and Richards, Wynette, "Foundations of Intrusion Detection," The IEEE Computer Security Foundations Workshop V, 1992.

[6] http://bammv.github.io/sguil/index.html

[7] https://www.snort.org/license

[8] Jeffrey Carr (2007-06-05). "Snort: Open Source Network Intrusion Prevention". Retrieved 2010-06-23.

[9] eWeek.com Staff (2008-04-04). "100 Most Influential People in IT". Retrieved 2010-06-23.

[10] Larry Greenemeier (2006-04-25). "Sourcefire Has Big Plans For Open-Source Snort". Retrieved 2010-06-23.

[11] "Cisco to Buy Sourcefire, a Cybersecurity Company, for $2.7 Billion". The New York Times. Retrieved July 23, 2013.

[12] Doug Dineley; High Mobley (2009-08-17). "The Greatest Open Source Software of All Time". Retrieved 2010-06-23.

[13] https://www.bro.org/

[14] "SteelApp for Application Delivery Control & Scalability". riverbed.com.

[15] Cortes, C.; Vapnik, V. (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297. doi:10.1007/BF00994018.

[16] Nalini L. A Comprehensive Approach to Intrusion Detection Alert Correlation[J]. Adarsh Journal of Information Technology, 2015, 2(1): 69-71.

[17] MIT Lincoln Laboratory, Lincoln Lab Data Sets,http://www.ll.mit.edu/IST/ideval/data/data_index.html, 2000.

[18] A.K. Ghosh, J. Wanken, and F. Charron, "Detecting Anomalous and Unknown Intrusions against Programs," Proc. Ann. Computer Security Application Conf. (ACSAC '98), pp. 259-267, Dec. 1998.

[19] R. Gula, "Correlating IDS Alerts with Vulnerability Information," 2002.

[20] technical report, Tenable Network Security, Dec. 2002.

[21] J. Haines, D.K. Ryder, L. Tinnel, and S. Taylor, "Validation of Sensor Alert Correlators," IEEE Security and Privacy Magazine, vol. 1, no. 1, pp. 46-56, Jan. Feb. 2003.

[22] C. Kruegel and W. Robertson, "Alert Verification: Determining the Success of Intrusion Attempts," Proc. First Workshop the Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 2004), July 2004.

[23] C. Warrender, S. Forrest, and B.A. Pearlmutter, "Detecting Intrusions Using System Calls: Alternative Data Models," Proc. IEEE Symp. Security and Privacy, pp. 133-145, 1999.

[24] J. McHugh, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evalautions as Performed by Lincoln Laboratory," ACM Trans. Information and System Security, vol. 3, no. 4, Nov. 2000.

[25] H.S. Javitz and A. Valdes, "The NIDES Statistical Component Description and Justification," technical report, SRI Int'l, Mar. 1994.

[26] S.M. Bellovin, "Packets Found on an Internet," technical report, AT&T Bell Laboratories, May 1992.

[27] F. Cuppens and A. Miege, "Alert Correlation in a Cooperative Intrusion Detection Framework," Proc. IEEE Symp. Security and Privacy, May 2002.