Implementation of CM-SPADE Algorithm In Building Denial of Service Detection System Model Using Snort

Eddy Prasetyo Nugroho[1], Rani Megasari[2], Enjun Junaeti[3], Samekto Rinekso Pribadi[4]
{eddypn@upi.edu[1], ranimegasari@upi.edu[2], enjun@upi.edu[3], samekto.rinekso@student.upi.edu[4]}

Computer Science Education Departement, Faculty of Science and Mathematics Education, Universitas Pendidikan Indonesia, Jl.Dr. Setiabudi no. 229 Bandung[1,2,3,4]

Abstract. Along with the increasing use of computers, business activities and data storage also use it. The increasing importance of the computer raises challenges and risks, especially in security of the computer system and network. Various ways have been done to improve computer network security, one of which is using the Intrusion Detection System (IDS). To maximize the IDS detection function that is able to detect unknown attacks, a data mining approach is used to create an IDS that uses anomaly-based detection techniques. In this study, a sequential pattern mining technique, the CM-SPADE algorithm, is used to generate IDS rules that can detect DoS attacks. Modeling and rules are made by applying the CM-SPADE algorithm to the KDD Cup 1999 data set. The results reveal that the implementation of the CM-SPADE algorithm is able to produce IDS rules that can detect DoS attacks with an accuracy rate of 97.976%.

Keywords: Intrusion Detection System, data mining, CM-SPADE, KDD Cup '99 Dataset, Snort.

1 Introduction

The use of information systems can not be separated from the use of computer network systems. The computer network system is used as a medium of communication between various users to exchange data more effectively and efficiently. The use of computer network systems is certainly not without flaws and risks. One of the big risks faced in computer network systems is the security of data stored in information systems.

Although the security rules have been widely applied, it's not stopping various intruder to find weaknesses and gaps in the security rules that have been applied. This is reinforced by the opinion that no matter how great the security applied, the stored data still becomes insecure [1].

Types of attack in a computer network can be categorized into four categories, and the most common and often type of attack used to intrude a network system is a DoS attack. DoS attacks are a type of attack where hackers make computing processes or memory resources on the server too busy and too full to serve the actual service request [2].

There are various ways that can be done to be able to detect DoS attacks, including using a protocol [3], applied scheduling using probability technique [4], statistical feature [5], and the most famous technique is using Intrusion Detection System (IDS).

IDS mainly function as early attack detector on network system so that those who manage the network system can get information when an attack occurs, this is reinforced by the

MSCEIS 2019, October 12, Bandung, Indonesia
Copyright © 2020 EAI
DOI 10.4108/eai.12-10-2019.2296332

opinion of a journal that IDS will monitor activities in a computer network system continuously, analyze data on these activities, and will notify the system administrator or network administrator in the event of an attack or suspicious activity [6].

One of the commonly used IDS applications is Snort. Snort is a signature-based IDS created in 1998 by Martin Roesch. Snort is open source and free. However, because it is anomaly-based, SNORT is unable to detect unknown types of attacks. To maximize the detection function, an IDS that is capable of carrying out the detection process of unknown types of attacks is needed. This triggered the rapid development of the use of data mining techniques in making IDS that applied anomaly-based detection.

Anomaly-based IDS is inseparable from data mining techniques. IDS is able to regularly find network behavior by analyzing data traces of activities that have been occurred using a data mining approach. As quoted from a journal, that there are two advantages gained from using the data mining approach on IDS, namely (1) IDS will be able to generate network activity models automatically, and (2) the data mining approach can be used to build IDS for various type of computing environment [7].

2 Literature Review

2. 1 Intrusion Detection System

Intrusion can be defined as any type of action that aimed to jeopardize the integrity, confidentiality, and availability of resources. IDS is designed to be able to detect all kinds of intrusion activities or all kinds of intrusion attempt which conducted in a computer system, by monitoring and analyzing network packages or system audit records, and then sending alert signals to the system administrator in real-time.

An optimal IDS is an IDS that has a high-level detection of system threats and also has relatively few frequencies of false alarms [1]. IDS monitors network packet data or system audit records, so IDS is associated with large data samples. Therefore, the application of data mining techniques makes it possible to exploit large data to get the rules based on existing characteristics and patterns. If implemented as to monitor computer network systems, data mining techniques can be used to detect intrusions, attacks and/or anomalies [6]. IDS has two different types of detection methods, which are signature-based and anomaly-based.

2.2 DoS Attacks

Simple tools that are capable of producing and sending packages from one source to one destination were used in DoS technology many years ago [8]. Generally, DoS attacks take the form of attacks that send large numbers of packets to a destination that causes excessive use of resources (CPU and memory) and network bandwidth.

2.3 Data Mining Approach on IDS

Data mining is a series of processes carried out to explore and obtain an added value from a large collection of data in the form of hidden knowledge that is invisible. There are two main techniques used in data mining, namely clustering techniques and classification techniques. The clustering technique is a technique used to group large data sets based on their similarity.

While in classification techniques, a pattern of data distribution in a large set of data will be found based on the target class that we have determined previously.

The implementation of the data mining approach is fundamental in IDS because IDS itself deals with large amounts of data. One of the data mining techniques that have been used is the clustering technique, such as the use of the ant colony algorithm called ANTIDS (ANT Colony based Intrusion Detection System). ANTIDS are able to guide agents to find patterns of an object in an adaptive way [9].

In addition, there is also research that combines two data mining techniques, such as a combination of SVM with ANT colony called CSVAC (Combining Support Vector with Ant Colony). CSVAC algorithm is able to minimize the use of training data sets, but still able to add new data points to the training set dynamically [10].

Besides CSVAC, there is also research that combines two data mining techniques, which combine K Means with RBF Kernel Function called KMSVM. In the KMSVM method only a few features are selected, so that data that has been measured can be organized into a group, and data that cannot be measured are organized into other groups [6].

2.4 CM-SPADE Algorithm

SPADE Algorithm [10] uses the vertical id-list format, this allows the formation of patterns and calculation of support for each sequence can be done without doing too much of the process of reading database that could cause a burden on the system. However, there is a weakness of the SPADE algorithm, which is due to the use of the generate-candidate-and-test approach. This approach could generate a lot of sequence patterns that don't appear very often or don't even appear at all in the database.

Based on the weakness of the SPADE algorithm, a modification of the algorithm called CM-SPADE is proposed, which will be used in this study. CM-SPADE provides a solution to SPADE's weaknesses based on the study of item co-occurrences [11]. In CM-SPADE, a new data structure called the Co-occurrence MAP (CMAP) is introduced. CMAP is a small and compact structure, that can also be built with a one-time database reading process.

The CMAP structure is used to carry out the process of pruning candidates generated by the SPADE algorithm. CM-SPADE will increase the speed of the process and reduce memory usage by applying an additional checking stage to candidates based on the data structure stored in CMAP [12]. This enables a faster process of removing less promising candidates and reduces the number of candidates who appear repeatedly (redundant candidates).

2.5 KDD Cup 1999 Data Set

KDD Cup is an annual competition that focuses on data mining and knowledge discovery material [13]. KDD Cup which was held in 1999 discussed intrusion detection on computer networks. The main task of the competition is to create an intrusion detector that is able to distinguish between 'bad' connections, which are intrusions or attacks on the network, and 'good' connections, which are normal connections [14].

The dataset provided in the KDD Cup ‘99 contains a collection of data in the form of various types of attacks simulated in military networks in military environments. There are forty-two different features in this dataset.

2.6 Snort IDS

Snort is an intrusion detection system and intrusion prevention system that was first created in 1998 by Martin Roesch. Snort is open source and also free.

Snort is composed of several components [15], which is 1) Packet Capture Library, 2) Packet Decoder, 3) Preprocessor, 4) Detection Engine, dan 5) Output.

3 Experiment

The experiment was carried out through three stages to measure the performance of the IDS model that was developed using the CM-SPADE algorithm. Three main stages which consist of:

3.1 Data Collection

The KDD Cup '99 data set are used as training data set and testing data set. The KDD Cup ‘99 data set was used because in 1999 the KDD Cup competition was focused on network security. Therefore, the data is very suitable to be used in building an IDS rule.

The data set is obtained from the KDD Cup official page. In this study, KDD Cup '99 training data set is used for the rules development process, and KDD Cup '99 data testing is used for testing the performance of the algorithm. KDD Cup '99 training data set consists of 4,898,431 instances, each of which has 42 attributes.

3.2 Data Preprocessing

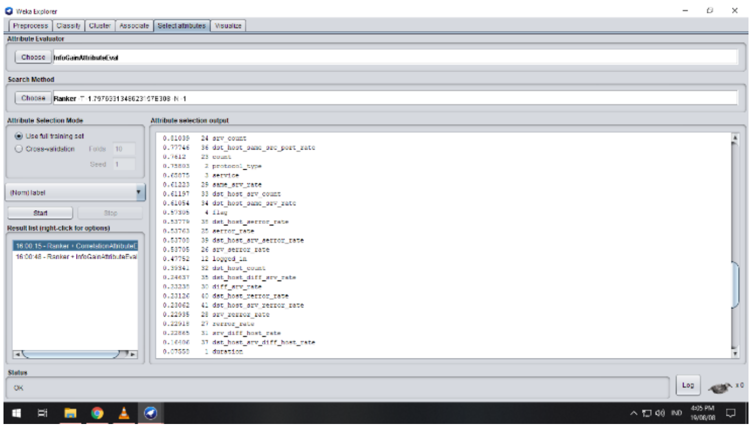
After the data is obtained, the next process is to preprocess the data. Data preprocessing is a process/step taken to make raw data into quality data. In this study, the data preprocessing process will be carried out using the Weka application.

Data preprocessing needs to be done because not all raw data obtained will be directly suitable for immediate application in the algorithm. Preprocessing data has a role to prepare the data so that the data can be processed by the algorithm smoothly and get good performance. Data preprocessing is carried out through 4 stages, which is: feature selection, discretize, remove unused data, and replace attribute value.

Feature selection. Is an activity that generally carried out in preprocessing and it aims to select features that have a significant effect and remove features that have insignificant or even no effect in a modeling activity or analyzing data.

In this research, the correlation attributes are used in the feature selection process. Correlation attribute is one of a few popular techniques used in doing the feature selection. The correlation attribute technique will measure how much correlation of information provided by a feature of the features that are sought.

After going through the feature selection process using the correlation attribute, a feature ranking is obtained, which is then used to determine the importance of the feature. The ranking of features can be seen in **Figure 1**.


Attribute Evaluator: InfoGainAttributeEval

Search Method: Ranker -T -1.7976931348623157E308 -N -1

Attribute selection output

```
0.81039   24 srv_count
0.77746   36 dst_host_same_src_port_rate
0.7612    23 count
0.75803    2 protocol_type
0.65875    3 service
0.61223   29 same_srv_rate
0.61197   33 dst_host_srv_count
0.61054   34 dst_host_same_srv_rate
0.57305    4 flag
0.53779   38 dst_host_serror_rate
0.53763   25 serror_rate
0.53708   39 dst_host_srv_serror_rate
0.53705   26 srv_serror_rate
0.47752   12 logged_in
0.39341   32 dst_host_count
0.24637   35 dst_host_diff_srv_rate
0.23238   30 diff_srv_rate
0.23126   40 dst_host_rerror_rate
0.23062   41 dst_host_srv_rerror_rate
0.22935   28 srv_rerror_rate
0.22918   27 rerror_rate
0.22865   31 srv_diff_host_rate
0.16406   37 dst_host_srv_diff_host_rate
0.07558    1 duration
```

Result list: 16:00:15 - Ranker + CorrelationAttributeE; 16:00:48 - Ranker + InfoGainAttributeEval

Status: OK

Fig. 1. Attribute Ranking.

After the ranking is decided, it is determined that the lower limit of ranking that will be used is 0.3. Thus, from a total of 41 attributes (with the exception of the label attribute), 15 attributes with the highest level of importance were chosen. The 15 selected attributes can be seen in Table 1.

Table 1. Selected Feature.

No	Attribute	No	Attribut
1	protocol_type	9	same_srv_rate
2	service	10	dst_host_count
3	flag	11	dst_host_srv_count
4	logged_in	12	dst_host_same_srv_rate
5	count	13	dst_host_same_srv_rate
6	srv_count	14	dst_host_serror_rate
7	serror_rate	15	dst_host_srv_serror_rate
8	srv_serror_rate		

Discretize. Some of the attributes that remain after performing feature selection are varied in value. Attributes numbers 4 through 15 in Table 1 have a large range of values. This will later complicate the algorithm in conducting the data mining process. Therefore, after the feature selection process, a discretizing process will be carried out on the data.

The discretize process used to group data into some small groups. This aims to simplify the algorithm in processing data. Because it is grouped, the processed data variant can be minimized.

Remove Unused Data. The process carried out next is to eliminate unused data. In the KDD CUP ‘99 data set, there are various kinds of labels that represent the types of connections that enter the system. Broadly speaking, connection types can be divided into 5, namely 1) normal, 2) DoS, 3) U2R, 4) Probes and 5) R21.

Because this study is focused on detecting DoS attacks, instances that have U2R, Probe and R21 labels need to be removed from the data. This aims to reduce the data that will be processed later.

After the data that labeled U2R, Probe and R21 were removed, the total amount of data was reduced to 4,856,149 instances.

Replace Attribute Value. The next process that was carried out is to change the value in each attribute. This had to be done so that the algorithm process can run with better performance, as well as facilitate the writer later in reading the pattern.

In this process, each value in the attribute will be changed to a nominal value starting from number 1, and each agency may not have the same value. In addition, values that have a lot of data variants, in this case, the attributes number 4 through 15 in Table 2, will be divided into three large groups. Namely data groups with low, medium and high values.

Replacing attribute value was carried out using the Rename Nominal Values function in weka. After the process was done, the values in the data are divided into 120 different values which spread over 16 attributes.

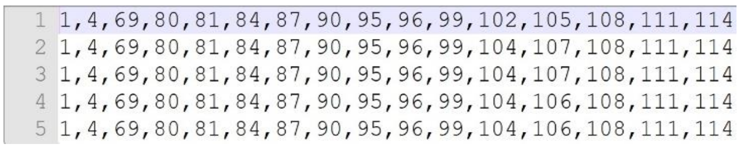
3.3 CM-SPADE Algorithm Implementation

The implementation of the CM-SPADE algorithm into the data set was carried out using the SPMF framework developed by Philippe Fournier Viger (http://www.philippe-fournier-viger.com/spmf/). The implementation consists of three stages, which is:

Turning the Data Set Into SPMF Format. The first thing to do when using SPMF is to change the data so that the data matches the SPMF format. This can be done by selecting the convert_a_transaction_database_to_SPMF_format option.

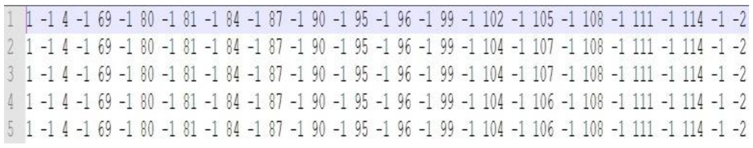
In this process, the data which was previously in the form of CSV will be converted into SPMF format data. This is done by removing the comma value used as a data separator in CSV and changing it with the number -1. So for data that has the SPMF format, each attribute will be separated by the number -1. In addition, each end of an instance will be given a value of -2. A value of -2 signifies the end of an instance, and the next value that is read by the system is the start of the next instance.

Examples of changes from CSV data into the SPMF format can be seen in **Figure 2** and **Figure 3**. **Figure 2** is an example of 5 instances of data that have gone through a preprocessing process. Because it is in the form of CSV, each attribute is separated by a comma symbol. Whereas **Figure 3** is an example of 5 institutions that are the same as **Figure 2**, but it has been changed to the SPMF format. Because they are in the SPMF format, each attribute is separated by the number -1, and at the end of each instance is represented by the number -2.



```
1,4,69,80,81,84,87,90,95,96,99,102,105,108,111,114
1,4,69,80,81,84,87,90,95,96,99,104,107,108,111,114
1,4,69,80,81,84,87,90,95,96,99,104,107,108,111,114
1,4,69,80,81,84,87,90,95,96,99,104,106,108,111,114
1,4,69,80,81,84,87,90,95,96,99,104,106,108,111,114
```

Fig. 2. CSV Format Data Sample.



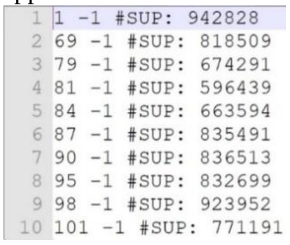
```
1 -1 4 -1 69 -1 80 -1 81 -1 84 -1 87 -1 90 -1 95 -1 96 -1 99 -1 102 -1 105 -1 108 -1 111 -1 114 -1 -2
1 -1 4 -1 69 -1 80 -1 81 -1 84 -1 87 -1 90 -1 95 -1 96 -1 99 -1 104 -1 107 -1 108 -1 111 -1 114 -1 -2
1 -1 4 -1 69 -1 80 -1 81 -1 84 -1 87 -1 90 -1 95 -1 96 -1 99 -1 104 -1 107 -1 108 -1 111 -1 114 -1 -2
1 -1 4 -1 69 -1 80 -1 81 -1 84 -1 87 -1 90 -1 95 -1 96 -1 99 -1 104 -1 106 -1 108 -1 111 -1 114 -1 -2
1 -1 4 -1 69 -1 80 -1 81 -1 84 -1 87 -1 90 -1 95 -1 96 -1 99 -1 104 -1 106 -1 108 -1 111 -1 114 -1 -2
```

Fig. 3. SPMF Format Data Sample.

Mining Process Using CM-SPADE Algorithm. After the data is changed to SPMF format, it means that the data is ready to be processed using the CM-SPADE algorithm. This is done by selecting the CM-SPADE option on SPMF.

Things to consider when using CM-SPADE is to determine the minimum support. Minimum support is the value that will determine which pattern will be taken by the algorithm. If the frequency of appearance of the pattern in the algorithm exceeds or equals minimum support, then the pattern will be a candidate for the formation of the next pattern. The higher the value of a pattern's support, the more patterns will appear in the data.

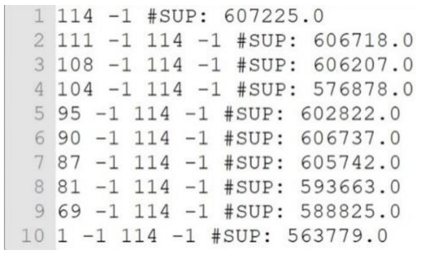
In this research, the minimum support was determined to 0.45. With that minimum value of support, the algorithm produces 742 different patterns. **Figure 4** shows an example of the first 10 patterns that emerge after mining. The #SUP value itself is a representation of the support value of each pattern that appears.



```
1 -1 #SUP: 942828
69 -1 #SUP: 818509
79 -1 #SUP: 674291
81 -1 #SUP: 596439
84 -1 #SUP: 663594
87 -1 #SUP: 835491
90 -1 #SUP: 836513
95 -1 #SUP: 832699
98 -1 #SUP: 923952
101 -1 #SUP: 771191
```

Fig. 4. Emerged Patterns Sample.

The purpose of this study is to look for patterns that can later be used to help predict the type of connection of instances. Patterns that do not have a label attribute value cannot be used in the next process. Therefore, mining results need to be filtered, so that only patterns that have labels are used. **Figure 5** shows the first 10 patterns that emerge after the filtering process was done.



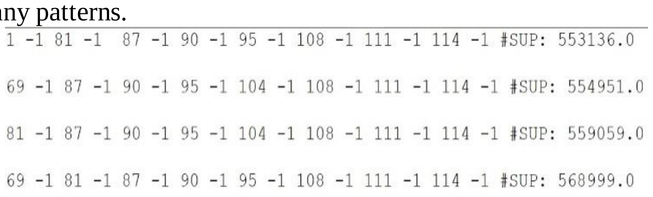
```
114 -1 #SUP: 607225.0
111 -1 114 -1 #SUP: 606718.0
108 -1 114 -1 #SUP: 606207.0
104 -1 114 -1 #SUP: 576878.0
95 -1 114 -1 #SUP: 602822.0
90 -1 114 -1 #SUP: 606737.0
87 -1 114 -1 #SUP: 605742.0
81 -1 114 -1 #SUP: 593663.0
69 -1 114 -1 #SUP: 588825.0
1 -1 114 -1 #SUP: 563779.0
```

Fig. 5. Emerged Pattern After Filtering Sample.

After the filtering process was done, the total number of patterns that were remained is 288 patterns.

Making IDS Rules. After the patterns that appear in the data have been obtained, then it is necessary to take a pattern that will later be used as a basis for intrusion detection rules.

The pattern that chosen is a pattern that contains previous patterns but still has a good support value. If looking for patterns that contain previous patterns, then in this study the longest patterns that have the highest support value are taken. **Figure 6** shows the selected patterns from many patterns.



```
1 -1 81 -1  87 -1 90 -1 95 -1 108 -1 111 -1 114 -1 #SUP: 553136.0

69 -1 87 -1 90 -1 95 -1 104 -1 108 -1 111 -1 114 -1 #SUP: 554951.0

81 -1 87 -1 90 -1 95 -1 104 -1 108 -1 111 -1 114 -1 #SUP: 559059.0

69 -1 81 -1 87 -1 90 -1 95 -1 108 -1 111 -1 114 -1 #SUP: 568999.0
```

Fig. 6. Longest Patterns with Highest Support.

From those four patterns, it can be concluded that normal data has the characteristics as shown in Table 2. If an instance in the system does not have the characteristics corresponding to Table 2, then it can be determined that the instance is a DoS connection type.

Table 1. Normal Connection Characteristic

Attribute Name	**Value**
flag	SF
count	<39.5
serror_rate	<0.165
srv_serror_rate	<0.335
same_srv_rate	>0.675
dst_host_same_srv_rate	>0.385
dst_host_serror_rate	<0.125
dst_host_srv_serror_rate	<0.045

4 Result

Performance testing is done by comparing the data from the learning process with data validation. After comparison, it will show how much accuracy that obtained.

The first thing to do is to collect testing data. In this research, the testing data used are KDD Cup ‘99 unlabeled testing data set and KDD Cup ‘99 corrected testing data set. Unlabeled data set was used as data that will store label that obtained from the learning process of the data mining process. So, the label of unlabeled is the result of implementing the model of existing data. While the corrected data is unlabeled data that already has a label that matches the actual label. Corrected data will be used as validation data for unlabeled data that has been through the implementation of the model that has been obtained. Testing data and corrected data have 311,029 agencies.

After the data is obtained, the next process that needs to be done is to clear the data first, or it can also be called data preprocessing. Just like the training data used to bring up the model at the research stage, testing data and corrected data also have labels that consist of five main categories of connection types, namely 1) normal, 2) DoS, 3) U2R, 4) Probe and 5) R21. Therefore, we need to eliminate instances that have the U2R, Probe and R21 labels so as not to interfere with the model performance testing process. The results of this cleaning caused the number of instances in the testing data set and corrected data set to be reduced to only 283,892 instances.

When instances other than DoS and normal are eliminated, after that we need to group all types of attacks into one, the DoS type. In testing and corrected data set, DoS attack types are divided into 6, namely 1) Back, 2) Land, 3) Neptune, 4) Pod, 5) Smurf and 6) Teardrop. But after going through the mining process, only a normal network type rules were obtained. This means that the rules are only able to detect 2 types of connections, which are only normal or intrusion, without being able to detail the type of incoming attack. So, all the labels are grouped into one type of attack, the DoS type.

When the data cleaning process has been carried out, then the testing data set will go through the process of implementing the rules. The implementation of the rules in the testing data set is done by filling in the label that was initially blank with the rules that were obtained from the mining process. After that, the label on the testing data set will have one of two values, namely normal or DoS.

When the testing data has a label, the next step is to test the performance of the model by comparing the label held by the testing data set with the actual label contained in the corrected data set.

From the results of experiments that have been conducted, the data that has gone through the preprocessing process is then used in the data mining process by implementing the CM-SPADE algorithm. After the algorithm is implemented and patterns that often appear are filtered, a rule that can be used to identify normal connection have emerged. The rules then were applied into the testing data set.

When the testing data set has a label, the next step is to test the performance of the rules by comparing the label held by the testing data set with the actual label contained in the corrected data. When comparing, the rules were able to guess the labels of 278,146 instances correctly, and guess the labels of 5745 instances incorrectly. Then it can be concluded that the rules have an accuracy of 97.976%, and only guessed incorrectly at 2.023%.

In this study, the authors used the KDD Cup 1999 data set as a training dataset and testing dataset. For future research, the authors suggest trying to use another network data set to find out the comparison between the generated model. Two of the examples for another dataset are the United States National Security Agency (NSA) dataset [17] and the UNSW-NB15 dataset [18].

5 Conclusion

This research has succeeded in building a model of the KDD Cup '99 data set by using the CM-SPADE algorithm. From this model, an IDS rule can be generated which can be used to identify the type of connection on a network, whether it is a normal connection or denial of service intrusion connection. The rules resulting from the implementation of the CM-SPADE algorithm have a high degree of accuracy when compared to other algorithms that have been applied using the KDD Cup '99 data set. Namely ANTIDS algorithm with an accuracy of 99.97%, CSVAC algorithm with an accuracy of 94.86%, and KMSVM algorithm with an accuracy of 93.33%.

References

[1] S. Elhag, A. Fernandez, A. Bawakid, S. Alshomrani dan F. Herrera, “On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on Intrusion Detection Systems,” Expert Systems with Applications, pp. 193-202, (2015).

[2] M. S. Hoque, M. . A. Mukit dan M. A. N. Bikas, “An implementation of intrusion detection system using a genetic algorithm,” International Journal of Network Security & Its Applications, (2012).

[3] M. Darwish, A. Ouda dan L. F. Capretz, “A cloud-based secure authentication (CSA) protocol suite for defense against Denial of Service (DoS) attacks,” Journal of Information Security and Applications 20, pp. 90-98, (2015).

[4] D. Seo, H. Lee dan A. Perrig, “APFS: Adaptive Probabilistic Filter Scheduling against distributed denial-of-service attacks,” Computer & Security 39, pp. 399-385, (2013).

[5]D. Gavrilis dan E. Dermatas, “Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features,” Computer Networks 48, pp. 235-245, (2005).

[6]U. Ravale, N. Marathe dan P. Padiya, “Feature Selection Based Hybrid Anomaly Intrusion Detection System Using K Means and RBF Kernel Function,” dalam International Conference on Advanced Computing Technologies and Applications, (2015).

[7]S. A. S., A. P., A. S. dan V. C., “An approach for IDS by combining SVM and ant colony algorithm,” International Journal of Research in Engineering and Technology, pp. 459-465, (2014).

[8]P. A. R. Kumar dan S. Selvakumar, “Detection of distributed denial of service attacks using an ensemble of adaptive and hybrid neuro-fuzzy systems,” Computer Communication, pp. 303-319, (2013).

[9]V. Ramos dan A. Abraham, “ANTIDS: Self Organized Ant-Based Clustering Model for Intrusion Detection System,” Soft Computing as Transdisciplinary Science and Technology, pp. 977-986, (2005).

[10]W. Feng, Q. Zhang, G. Hu dan J. X. Huang, “Mining Network Data for Intrusion Detection through Combining SVM with Ant Colony,” Future Generation Computer Systems, (2013).

[11]M. J. Zaki, “SPADE: An Efficient Algorithm for Mining Frequent Sequences,” Machine Learning, pp. 31-60, (2001).

[12]P. Fournier-Viger, A. Gomariz, M. Campos dan. R. Thomas, “Fast Vertical Mining of Sequential Patterns Using Co-occurrence Information,” Advances in Knowledge Discovery and Data Mining, pp. 40-52, (2014).

[13]B. Huynh, C. Trinh, H. Huynh, T.-T. Van, B. Vo dan V. Snasel, “An efficient approach for mining sequential patterns using multiple threads on very large databases,” Engineering Applications of Artificial Intelligence, pp. 242-251, (2018).

[14]“KDD CUP ARCHIVES,” 2018. [Online]. Available: https://www.kdd.org/kdd-cup.

[15]“KDD Cup 1999 Data,” 28 October 1999. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[16]H. Bintara, “Mengenal Snort Sebagai Network Intrusion Detection System (NIDS),” 16 February 2017. [Online]. Available: https://netsec.id/snort-nids/.

[17] “Cyber Research Center Data Sets,” West Point, [Online]. Available: https://westpoint.edu/centers-and-research/cyber-research-center/data-sets. [Accesed on 28th August 2019].

[18] N. Moustafa dan J. Slay, “The UNSW-NB15 Dataset Description,” UNSW Canberra, 14 November 2018. [Online]. Available: https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. [Accesed on 28th August 2019].