A Multiple Instance Learning Problem Approach Model to Anomaly Network Intrusion Detection

Ill-Young Weon*, Doo-Heon Song**, Sung-Bum Ko***, and Chang-Hoon Lee*

Abstract: Even though mainly statistical methods have been used in anomaly network intrusion detection, to detect various attack types, machine learning based anomaly detection was introduced. Machine learning based anomaly detection started from research applying traditional learning algorithms of artificial intelligence to intrusion detection. However, detection rates of these methods are not satisfactory. Especially, high false positive and repeated alarms about the same attack are problems. The main reason for this is that one packet is used as a basic learning unit. Most attacks consist of more than one packet. In addition, an attack does not lead to a consecutive packet stream. Therefore, with grouping of related packets, a new approach of group-based learning and detection is needed. This type of approach is similar to that of multiple-instance problems in the artificial intelligence community, which cannot clearly classify one instance, but classification of a group is possible. We suggest group generation algorithm grouping related packets, and a learning algorithm based on a unit of such group. To verify the usefulness of the suggested algorithm, 1998 DARPA data was used and the results show that our approach is quite useful.

Keywords: Multiple Instance Learning Problem, Network Intrusion Detection, Anomaly Detection.

1. Introduction

The traditional detection method used in the Network Intrusion Detection System (NIDS) is misuse detection. It extracts the signature from previous attack data, generates a pattern and detects new attacks by comparing them to the pattern. This manner of detection means a pattern has to be made for every attack. Also it is hard to detect when changed or unregistered attacks take place. To overcome this problem, anomaly detection was introduced. [1,2,3].

Anomaly detection has developed into a statistics-based and learning-based method. The first one utilizes various statistic techniques, models a normal status and sets a statistical threshold. By using the threshold, the network status is determined during detection of whether it is normal or abnormal. Learning based anomaly detection uses machine learning in artificial intelligence, makes a pattern during learning, and determines normal or abnormal status in detection by using this pattern. Even though statistic anomaly detection is strong in certain types of attack such as a DOS type attack, it cannot detect all kinds of attacks. On the other hand, machine learning based anomaly detection can complement weaknesses of the statistical method. Also, as it does not need the hands of human experts in creating learning patterns, it has lower costs in pattern creation and maintenance. The learning based anomaly method started from research applying traditional machine learning algorithms in artificial intelligence to detect intrusion. The problems are that it has a high false positive and low detection rate. Nevertheless, supervised learning shows better performance than unsupervised learning relatively in NIDS [4,5].

The smallest information unit, which can be acquired in a network, is a packet in learning based anomaly detection. How the packets are processed can show quite a different result even though the same learning algorithm is used. From the perspective of learning time, if it takes a long time to make a pattern in training, it may be a trivial problem. However, if it takes long time to process prior to detection in testing, it may be a big problem in terms of usefulness [6,7].

The current approach method which extracts each attribute from one packet and makes a basic unit data of learning, raises alarms per packet which consist of one attack. As a result, there are a lot of repetitious alarms. As a certain type of attack which consists of many packets cannot be detected with only one packet, the current way of detection, which is based on one packet, finds it hard to detect various attacks [20]. To solve this problem, simultaneous consideration of several packets related to one attack is required. That is, a new approach of group-based learning and detection is needed, with grouping of

Manuscript received September 30, 2005; accepted November 21, 2005. This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment).

* Dept. of Computer Engineeringeng neering, Konkuk University, Seoul, Korea ({clcc, chlee}@konkuk.ac.kr)

** Dept. of Computer Games & Information, Yong-in SongDam College, Young-in, Korea (dsong@ysc.ac.kr)

*** Dept. of Computer Science, Kongju National University, Kongju, Korea (sbko@cntc.ac.kr)

related packets. This type of approach is similar to that of the multiple instances learning problem in artificial intelligence, which cannot clearly classify one instance, but classification of groups is possible if several packets are grouped [8,9].

This paper proposes a model that applies a multiple-instance learning approach to detecting anomaly network intrusion and shows its usefulness. We suggest a group generation algorithm in which packets are profiled according to their destination IP and are divided into related groups. Also we propose a new group-based learning algorithm. We call this group of packets BAG. With a method of giving the class value to the BAG, a new BAG learning algorithm is proposed. The proposed BAG learning algorithm is based on memory-based learning [15]. To find out the usefulness of the proposed model, DARPA data from 1998 was used [10,11].

The following is a composition of this paper. Chapter 2 explains the motivation of BAG and the BAG generation algorithm. Chapter 3 deals with the BAG learning algorithm. Chapter 4 analyzes the experiment of the proposed model and the result. Chapter 5 comprises a conclusion and future works.

2. Modeling of Packet Stream

2.1 Motivation

An analysis of an actual stream of packets in a network shows that normal and abnormal packets exist together. Fig 1 displays an example of the packet stream. If you see the Fig. 1, one attack related packets are mixed with normal packets. Therefore, if we collect all packets related to attacks, normal ones can be easily found. As an individual treatment of packets lowers efficiency, we need a group treatment. That is even though an individual view toward packets cannot detect attacks effectively, a comprehensive view can find attacks easily. This kind of approach has the same perspective of multiple instances learning in a machine-learning community. Therefore, we may effectively solve intrusion detection problems by using the above method. From now on, a group of related packets will be referred to as BAG in line with the practices of the *multiple-instance community*.

1 885592572 133860 20 64 0 tcp 192.168.1.30 192.168.0.20 6 2 1754 23 off off off off on outbound normal - -1 -1 -1
2 885592572 138635 20 254 0 tcp 192.168.0.20 192.168.1.30 6 14370 23 1754 off off off off off on outbound normal - -1 -1 -1
3 885592572 139101 20 64 0 tcp 192.168.1.30 192.168.0.20 5 124 1754 23 off off off off on off outbound normal - -1 -1 -1
4 885592572 158827 20 64 0 tcp 192.168.1.30 192.168.0.20 5 124 1754 23 off off off off on off outbound normal - -1 -1 -1
5 885592572 209927 20 254 0 tcp 192.168.0.20 192.168.1.30 5 14370 23 1754 off off off off on off outbound normal - -1 -1 -1
6 885592573 56771 20 254 0 tcp 192.168.0.20 192.168.1.30 5 14370 23 1754 off off off off on off outbound normal - -1 -1 -1
7 885592573 58434 20 64 0 tcp 192.168.1.30 192.168.0.20 5 124 1754 23 off off off off on off outbound normal - -1 -1 -1
8 885592573 62683 20 254 0 tcp 192.168.0.20 192.168.1.30 5 14370 23 1754 off off off off on off outbound normal - -1 -1 -1
9 885592573 82117 20 64 0 tcp 192.168.1.30 192.168.0.20 5 124 1754 23 off off off off on off outbound normal - -1 -1 -1
10 885592573 82916 20 254 0 tcp 192.168.0.20 192.168.1.30 5 14370 23 1754 off off off off on off outbound normal - -1 -1 -1
...
504 885592605 928858 20 64 0 tcp 192.168.1.30 192.168.0.40 6 2 1784 80 off off off off off on outbound abnormal phf 15 0 15
505 885592605 929443 20 254 0 tcp 192.168.0.40 192.168.1.30 6 14370 80 1784 off off off off off on outbound normal - -1 -1 -1
506 885592605 929888 20 64 0 tcp 192.168.1.30 192.168.0.40 5 124 1784 80 off off off off on off outbound abnormal phf 15 1 15
507 885592605 930298 20 64 0 tcp 192.168.1.30 192.168.0.40 5 124 1784 80 off off off off on off outbound abnormal phf 15 2 15
508 885592605 936177 20 254 0 tcp 192.168.0.40 192.168.1.30 5 14370 80 1784 off off off off on off outbound normal - -1 -1 -1
509 885592605 936331 20 254 0 tcp 192.168.0.40 192.168.1.30 5 14370 80 1784 off off off off on off outbound normal - -1 -1 -1
510 885592605 936700 20 64 0 tcp 192.168.1.30 192.168.0.40 5 124 1784 80 off off off off on off outbound abnormal phf 15 3 15
511 885592605 939439 20 64 0 tcp 192.168.1.30 192.168.0.40 5 124 1784 80 off off off off on off outbound abnormal phf 15 4 15

Fig. 1. Packet stream sample

Based on packet stream, attacks and victims are divided according to their IPs and ports. Table 1 contains detailed information about this.

Table 1. Attack types from the viewpoint of packet stream

Attack Type	Attacker IP	Attacker Port	Victim IP	Victim Port
0	static	static	**static**	static
1	static	dynamic	**static**	static
2	dynamic	static	**static**	static
3	dynamic	dynamic	**static**	static
...	...	...	...	...
15	dynamic	dynamic	dynamic	dynamic

In the early years of networks, the majority of attack types had fixed factors. However current types show an increasing rate of floating factors. No. 0 is a basic type of attack. No.15 is known as a perfect attack type, which has not yet been found so far. Ways to classify attacks into related packets include a packet grouping by TCP session, which is suitable only for the No. 0 type of attack. Also, if attackers do not complete sessions intentionally, it is hard to end grouping. In this regard, methods are needed to deal with more various kinds of attacks as well as the No. 0 type.

2.2 Feature Selection

Information can be acquired in a packet from a header and data part. According to layers of detecting protocols, the data part can be a header part. While we collected packet data from a TCP level, necessary learning data was

acquired from a header part of TCP [13].

Fifteen signatures were found in the TCP header. To extract a signature related to class values, we applied a mutual information technique [12]. Mutual information can measure a relation degree between fields in one data, with a range of 0 to 1. The more it is close to 1, the more two fields are related, and vice versa. The following is a mutual information formula we used.

$$I(X,Y) = \sum_{x,y} p(x,y) \times \log_2 \frac{p(x,y)}{f(x)g(y)}$$

Table 2 shows mutual information values of each field and class on 8 selected fields among 15 signatures.

Table 2. Result of mutual information value for feature selection

Field	MI	Selection
Service type	0.0035	Accepted
Time to live	0.0838	Accepted
Source IP	0.0865	Accepted
Destination IP	0.1535	Accepted
Source port	0.3253	Accepted
Destination port	0.127(	Accepted
Ack	0.0007	Accepted
Urg	0.0007	Accepted
Class		

2.3 BAG Generation

This paper suggests a packet profiling by victim's destination IP rather than a TCP session unit for making BAGs. This means we think from the perspective of victims. After raw packet streams are reclassified by destination IP, they are classified into related packet groups once again. Finally, by choosing an appropriate number of packets, BAGs are made.

Several methods were adopted to divide packets, which were classified by destination IP into BAGs, related groups. The time of occurrence for packets, and individual checking of packets acted as standards. Regarding the time of occurrence for packets, if new packets occur in BAG and if the occurrence time of the last packet in BAG and that of a new packet is higher than the threshold (certain values), it is considered as the appearance of the new stream of packet. In this case, the existing BAG is closed, and then replaced by a new BAG. In addition, if the difference between time of occurrence for new packets and average time of occurrence for existing packets in BAG is larger than the threshold, the existing BAG is closed, and then replaced by a new BAG.

For class check of packets, the IBL existing machine learning algorithm is extended to find out whether each packet has a normal class value or not. When newly occurred packets are added as factors of BAG, the extent of risk is calculated. If the difference between the extent of risk with added packets and the extent of risk without any added packets in BAG is larger than the threshold, it is considered as an appearance of a new stream. The existing BAG is maintained. However, a new BAG is started to reflect a new stream of packets. The following shows how to define the extent of risk in BAG A.

$$RR(A) = \frac{Number\ of\ abnormal\ packets\ for\ BagA}{Total\ packet\ number\ for\ BagA}$$

The learning algorithm we used in the class check of packets is extended, focusing on three functions like below with original IBL [14,15].

First, in the case of distance calculation between instances, the IBL of symbol type data is expressed as 0 or 1. Therefore compared with standardized attributes, which tend to have values between 0 and 1, it has more weighted values. To reflect such tendency, XIBL applied the "Value Difference Metric" (VDM) which can express distance between discrete data in consecutive numbers between 0 and 1 [18].

A second, "Leave-one-out" noise filter based on statistics was adopted to solve noise-sensitiveness of IBL [14,18].

A third, "backward stepwise regression" class check determination method, which was proved statistically, was adopted rather than a "reward-penalty" in designating class weighted values among selected learning by IB4.

```
LA: Learning algorithm for packet classification
RR(A): Risk rate of BAG A
ATD(A): Average time difference of packet origination time in BAG A
LT(A): Origination time of latest packet in BAG A
T(P): Origination time of packet P

Grouping raw packets with the same destination IP.
For each same destination IP packet P
{
  Classify packet P's class value as normal or abnormal using LA.
  If(there is no BAG which has the same destination IP)
  {
    Begin a new BAG.
  }
Else
  {
    For each same destination IP BAG A
    {
      If( |LT(A) – T(P)| > threshold1)
      {
        End BAG A and begin a new BAG.
      }
      Else if( | ATD(A) with P - ATD(A) without P | > threshold2)
      {
        End BAG A and begin a new BAG.
      }
      Else if( |RR(A) with P – RR(A) without P| > threshold3)
      {
        Begin a new BAG.
      }
      Else
      {
        Add a packet P to BAG A.
      }
    }
  }
}
```

Fig. 2. BAG generation algorithm

A detailed explanation about XIBL and a functionality experiment is beyond the scope of this paper. However, several papers that have already been published contain information about it [16,17]. The reason why we used IBL in the class check is that it showed better functions than C4.5 or NN in previous network intrusion detection studies which targeted one packet as learning object [9].

Fig. 2 describes the creation process of BAG in algorithm.

3. BAG Learning

3.1 Similarity of Two BAGs

As the BAG learning algorithm we are using is based on memory based learning, it is important to measure the similarity of two BAGs. If we assume BAG A is composed of m packet instances, A is defined as {A1, A2, A3...Am}. If we assume BAG B is composed of n packet instances, B is defined as {B1,B2,B3,...,Bn}. If the first instance, A1 consists of k class values, A1 is defined as { a_{11}, a_{12}, $a_{13,...}$, a_{1k} }. The same applies to B1 defining B1 as { b_{11}, b_{12}, $b_{13,...}$, b_{1k} }. Also, if it is defined like a_{*1} ={a_{11},$a_{21}a_{31}$,a_{m1} and b_{*1}={b_{11},b_{21},b_{31},...,b_{n1}}, and if class values are consecutive of a certain field, the distance between a_{*i} and b_{*I} ($d(a_{*i}, b_{*i})$) is as follows:

$$d(a_{*_i}, b_{*_i}) = \frac{1}{m}\sum_{j}^{m} a_{ji} - \frac{1}{n}\sum_{j}^{n} b_{ji}$$

If the i^{th} field is discrete and the possible number of values is S, Pai(j) is defined as a probable value of number J in a_{*i}. Therefore, in the case of discrete value, $d(a_{*i}, b_{*i})$ is defined as follows:

$$d(a_{*_i}, b_{*_i}) = \frac{1}{S}\sum_{j=1}^{s} (p_{ai}(j) - p_{bi}(j))$$

D (A, B), distance between BAG A and BAG B is defined as follows:

$$D(A, B) = \sum_{i}^{k} d(a_{*_i}, b_{*_i})^2$$

Similarity between BAG A and B is defined as follows:

$$Similarity(A, B) = \frac{1}{\sqrt{D(A, B)}}$$

3.2 BAG Learning Algorithm

There is a difference in ways of dealing with BAGs between traditional multiple instance problems and intrusion detection. Traditional multiple instance problems determine the class of BAG as abnormal even if only one attack instance is included in one BAG. However, if the same approach applies to intrusion detection, a lot of alarms will go off with a high number of false positives. As a result, to decide the class of one BAG, we utilized Abnormal Rate of BAG: ARB, which is a proportional value of abnormal instance against all instances of BAG as a parameter. ARB was applied in different ways.

The learning algorithm we use is based on memory-based learning. This is first to store learning BAGs as knowledge. Find out a BAG most similar to the one to be detected. Determine the class of a BAG to be detected as the same as that of the most similar BAG. This type of learning is simple. Also, compared to inductive learning or neuropil, learning speed is fast. As expressed knowledge is BAG itself, people can understand the basis of the decision to some extent. As mentioned before, IBL, a single-instance version of IBLmemory based learning showed a positive result in intrusion detection [19].

A big problem of the memory-based algorithm is that it requires a reduction of knowledge. If learning materials are huge, it takes a large part of the memory. So, to efficiently detect network intrusion, knowledge should be reduced to a certain level. To solve such issues, geometric clustering was adopted. This is to find out the density of BAG groups with the same class and align BAGs in an order of closeness to the center of calculated density. Centering on the closest BAG, a semi diameter starts to extend to include the next BAG. Knowledge is expressed with a central BAG of a cluster and semi diameter as a sphere convex hull (SCH). If A is the center of the BAG and semi diameter is R, it is described as SCH (A, R). If an extended SCH includes BAGs with different classes, borders or overlaps with SCH with same class, the semi diameter of the extended SCH decreases so as not to border, overlap or include BAGs. This approach is based on an idea that if clusters with the same class collect in spaces of n dimensions, the center and semi diameter are enough to express that group clearly. Also even though we cannot divide spaces of n dimensions into groups of SCH precisely, SCH of high density can reduce storage spaces a lot. However, this method cannot include all patterns.

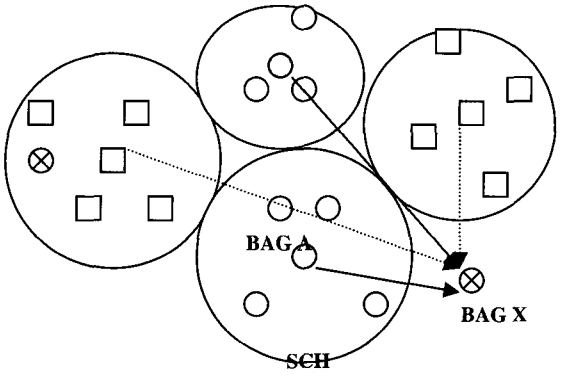

Fig. 3. The concept of the detection process for BAG X in SCH space

In the case of detection, BAG X should be viewed to ascertain whether its unknown class is included in stored patterns, SCH. If included, the class of BAG X is determined as that of a central BAG. If not included in any SCH, the distance between BAG and SCH is calculated to select number K SCH. By calculating the abnormal rate (ARSCH) of included SCH, threshold values are determined, leading to class values. Therefore, distance similarity between BAG X and SCH with A as a central BAG and semi diameter R, - D_{SCH} (X, SCH (A, R))- is defined as follows:

$$D_{SCH}(X, SCH(A,R)) = |D(X,A) - R|$$

$$Similarity_{SCH}(X, SCH(A,R)) = \frac{1}{\sqrt{D_{SCH}(X, SCH(A,R))}}$$

Fig.3 shows a detection process in a two-dimensional space. Fig. 4 and Fig. 5 describes Sphere Convex Hull Learning (SCH) in codes.

```
SCH: Sphere convex hull which contains center BAG and Radius
R: Radius of SCH
V: Class value
Save all learning BAG to memory.
For each BAG's class value V
{
  While(There are any BAG which is not included in SCH)
  {
    Get the density of every BAG which has the same class value V.
    Select SCH's center BAG which has a most similarity to BAG's
    density.
    Extend R to the BAG which has a most similarity to the center
    BAG.
    While(There is no collision with another SCH AND There is no
    BAG which has another class)
    {
      Extend R to the Next BAG and Remove Previous BAG from
      memory.
    }
  }
}
```

Fig. 4. BAG training algorithm

```
ARSCH: Abnormal rate for SCH
X: Test BAG
K: Number of SCH to conclude X's class
Similarity(X,SCH): Similarity between BAG X and SCH

For Test BAG X
{
  For each SCH
  {
    If( X is inner space of SCH)
    {
      Get X's class value as SCH's class and terminate process.
    }
  }

  For each SCH
  {
    Get Similarity(X,SCH)
  }
  Select K number of SCHs which have most similarity for BAG X.
  Get ARSCH for K number of SCH.
  If( ARSCH >= threshold )
  {
    Set BAG X's class value as "abnormal".
  }
  Else
  {
    Set BAG X's class value as "normal".
  }
}
```

Fig. 5. BAG testing algorithm

4. Experiment Results

4.1 Experimental Data and BAG Generation

For experiments, we used DARPA data from 1998. The DARPA data, which were all data from the previous 7 weeks, provide tcpdump of network packets and list files on attacks. As the size of data was too huge, we only introduced the data on Thursday of the 6th week. As various kinds of attacks were implemented on that day and the size of those were appropriate, the data was chosen for the experiment.

The scope of the network to be detected was limited to TCP. UDP, ICMP etc., were excluded. Considered types of attacks were also restricted to DOS, PROBE, and R2L. The U2R attack type, which has to analyze bsm of hosts, was excluded as it is beyond the purpose of this paper.

For the creation of BAG, we used 2 time-related and 1 risk-related parameters. According to the changes in parameters, diverse results have come out. To illustrate, we analyzed the case in which threshold A, B, and C were set as 0.5 like below. This case has shown a good result in threshold values change. About 30,000 BAGs were used in analysis and in three parts, a feasibility of created BAGs were analyzed. The first is to analyze the number of packets included in each created BAG. If the number of packets is too small or big, BAG grouping is meaningless. Fig. 6 shows a size distribution of packets consisting of BAGs. The number of packets in one BAG ranges from 50 to 20 with an average of 30 and the standard deviation of 20. Second, regarding class distribution of packets, we

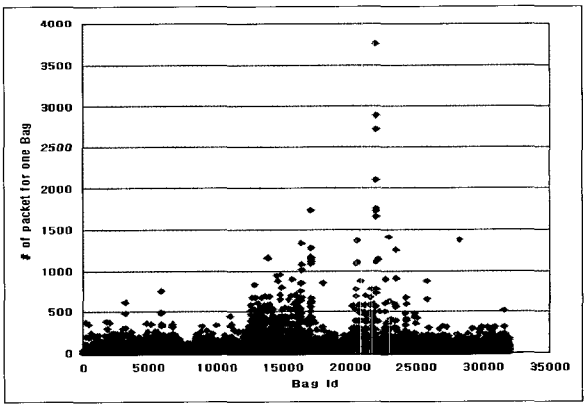

Fig. 6. A distribution of packet number for one BAG

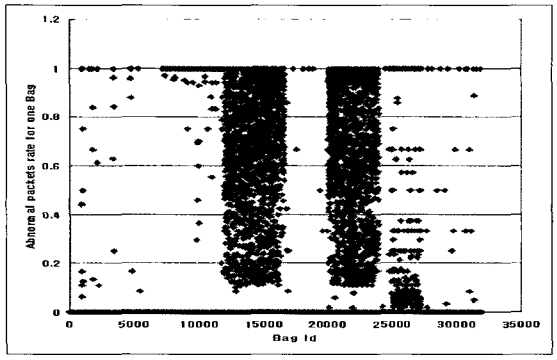

Fig. 7. A distribution of abnormal packet rate for one BAG

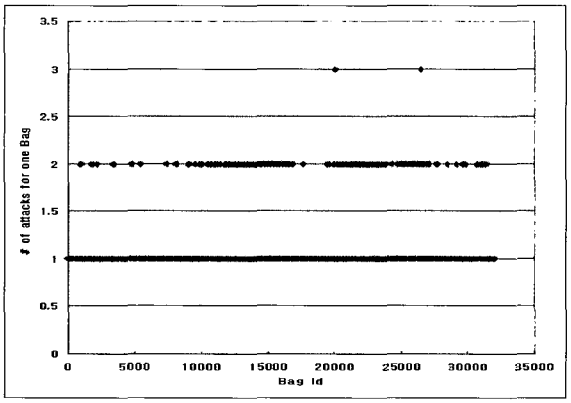

Fig. 8. A distribution of number of attacks for one BAG

measured abnormal rate in packets. As this value determines a representative class of BAGs later, it is very important. If the value is close to 0 or 1, it means the BAG is created with packets of similar class. The Fig. 7 shows a distribution rate of abnormal packets in one BAG. While the value is close to 0 or 1 in most cases, some BAGs were clustered around 0.5. Third, we analyzed whether packet streams related to same attacks or normal sessions composed of one BAG or not. Fig. 8 displays the number of attacks forming one BAG. In this part, we regarded a nor mal packet stream as one type. That is if the number of attack types in BAGs is 2, it is composed of normal stream and 1 attack type.

4.2 Results for Three Attack Categories

We designated ARB values as 70% to determine the representative class of BAGs. The percentage, 70% is a logical choice. In case of a DOS attack type, we set 2,000 normal BAGs and 1,500 abnormal BAGs. In PROB, normal BAGs and abnormal BAGs were 1,000 and 1,200 respectively. For R2L type, with 1,000 normal BAGs and 1,200 abnormal BAGs, we divided it into 30% for testing and 70% for learning. When tested BAGs were not included in any SCH, the standard K values determining the closest SCH were segmented into 1, 3 and 5.

The Fig. 9 describes an analysis of results by using each attack type and attack detection rates according to K values. In general, while the result of PROB was showing a good performance, a detection rate was low in the case of R2L. The most probable reason is that related patterns were located in data parts of a packet. The Fig. 10 indicated the number of false alarms in normal BAGs per attack type as a percentage against K values. As DOS type attacks take place more often than other attack types, it had a higher false alarm rate.

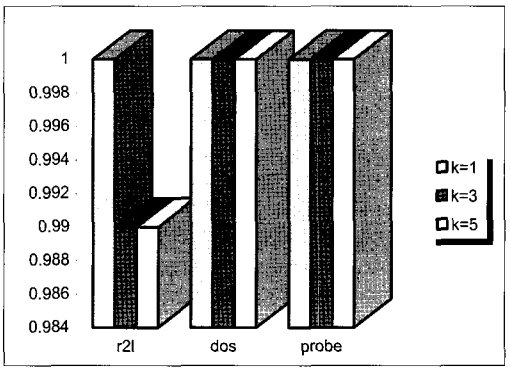

Fig. 9. The percentage attacks detected by K

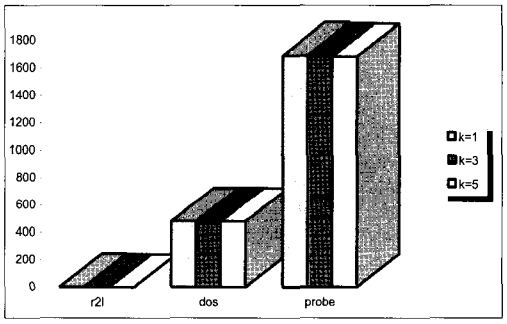

Fig. 10. False alarms for one day by K

5. Conclusion

To detect various kinds of attacks and reduce repetitious alarms on the same attack, it is effective to deal with packets as continuous streams. While existing anomaly detection based on machine learning deals with a packet as a basic unit, this paper suggests a new approach to dealing with groups of related packets. We propose a BAG creation algorithm which groups packets and a learning algorithm based on a unit of BAG.

To find out the functionality of proposed models, a part of the DARPA data is extracted. We analyze the size of the BAG, an abnormal rate in BAG, and the types of attacks consisting of BAGs to find out the usability of the creation algorithm. The proposed BAG creation algorithm shows a good size and abnormal rate. In addition, we are able to see that attack types of BAG are created through a continuous stream of packets. An attack detection rate and false alarm rate are analyzed by attack type. While PROB type is easily detected, the detection rate in the R2L type is relatively low. However, if we compare the result of this

paper to that of existing methods based on one packet, the quantity and quality of alarms are better.

It is necessary to extend the proposed model. First of all, as the proposed learning algorithm is based on batch processing, we have to change it into an incremental learning method in order to decrease the learning space. Also, an evaluation method of the system is changed from the simple learning method to the special method which considers attack domain knowledge.

References

[1] C. Kruegel and G. Vigna. Anomaly detection of web-based attacks. In Proceedings of the 10th ACM Conference on Computer and Communication Security (CCS '03), pages 251--261, Washington DC, USA, October 2003. ACM Press.

[2] P. Barford, J. Kline, D. Plonka, and A. Ron. A Signal Analysis of Network Traffic Anomalies. In Proceedings of ACM SIGCOMM Internet Measurement Workshop, November 2002.

[3] F Gonzalez and D Dasgupta. Anomaly detection using real-valued negative selection. Journal of Genetic Programming and Evolvable Machines, 4:383--403, 2003.

[4] Javitz, H. and Alfonso Valdes, S. The NIDES Statistical Component Description and Justification, Annual Report, SRI International, 333 Ravenwood Avenue, Menlo Park, CA 94025, March 1994.

[5] M. Markou and S. Singh. Novelty detection: a review-part 1: statistical approaches. Signal Processing, v.83 n.12, p.2481-2497, December 2003

[6] W. LEE. "A Data Mining Framework for constructing Features and Models for Intrusion Detection Systems", Ph.D. Dissertation, Columbia University, 1999.

[7] A.K. Ghosh, A. Schwatzbard and M. Shatz, Learning Program Behavior Profiles for Intrusion Detection, in Proceedings 1st USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, California, April 1999.

[8] Wang, J. and Sucker, J.-D. Solving the Multiple-Instance Learning Problem: A Lazy Learning Approach, Proceedings 17th International Conference on Machine Learning (pp. 1119-1125). San Francisco: Morgan Kaufmann, 2000.

[9] Dietterich, T. G., Lathrop, F. H. and Lozano-Perez, T. Solving the multiple-instance problem with axis-parallel rectangles. Artificial Intelligence, 89, 31-71.1997.

[10] Lippman. R. et. Al. Evaluation intrusion detection systems: The 1998 DARPA Off-line intrusion detection evaluation, Proc. Of DARPA Information Survivability Conference and Exposition, pp 12-26, 2000.

[11] DARPA data set: www.ll.mit.edu/IST/ideval

[12] Mutual Information: http://en.wikipedia.org/wiki/Mutual_information

[13] Behrouz A. Forouzan. TCP/IP Protocol Suite. MaGRAW-HILL, 2000.

[14] Aha, D. & Kibler, D., Noise-tolerant instance-based learning algorithms. Proceedings of the Eleventh International Joint Conference on Artificial Intelligence pp.794-799, 1989.

[15] Stanfill C., & Waltz, D., Toward memory-based reasoning. Communications of the ACM, 1986.

[16] Won, I., Song, D., Lee, C. Heo., Y. & Jang, J., A Machine Learning approach toward an environment-free network anomaly IDS – A primer report, In Proc of 5th International Conference on Advanced Communication Technology, 2003.

[17] Song,D.,Won, I.,Cang, Lee, The Utility of Packet level decision in Misused Intrusion Detection System: An analysis of DARPA dataset toward a hybrid behavior based IDS. The 3rd Asia Pacific International Symposium on Information Technology, Jan. 13-14 2004, Istanbul, Turkey.

[18] S. Cost, and S. Salzberg, A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features, Machine Learning 10, 00.57-78,1993.

[19] Joo, D., The Design and Analysis of Intrusion Detection Systems using Data Mining, KAIST PH.D , 2003.

20. Sadat Malik. Network Security Principles and Practices, Cisco Press, pp. 420. 2003.

Ill-Young Weon
1997- BS Computer Science at KyungWon University
2000- M.S Computer Engineering at Konkuk University.
2000 ~ - Ph.D. course at Konkuk University
Interested: Artificial Intelligence, Network Security, Complexity Theory

Doo-Heon Song
1981- BS Calculus Statistics at Seoul University
1983- MS Computer Science at Korea Institute Science Technology University
1994- Certificate Ph.D. Computer Science at University of California
1983~1986- Researcher at Korea Institute Science Technology University
1997~ - Professor at Yongin Songdam College
Interested: Data Mining, Machine Learning, Database, Security

Sung-Bum Ko
1980- BS Electrical Engineering at Soongsil University
1983- MS Electrical Engineering at Seoul University
2003- Ph.D. Computer Science at Konkuk University
1983 ~ - Professor at Kongju University
Interested: Artificial Intelligence, Complexity Theory

Chang Hoon Lee
1980- BS Mathematics at Yonsei University
1987- MS Computer Science at Korea Institute Science Technology University
1993- PhD. Computer Science at Korea Institute Science Technology University
1996 ~ 2000- Head of Information Technology Center at Konkuk University
2001 ~ 2002- Rector of Information Technology Center at Konkuk University
1980 ~ - Professor at Konkuk University
Interested: Intelligent Systems, Operating Systems, Security, E-Commerce