IEICE TRANS. INF. & SYST., VOL.E93–D, NO.9 SEPTEMBER 2010

PAPER

A Comparative Study of Unsupervised Anomaly Detection Techniques Using Honeypot Data

Jungsuk SONG[†a)], ***Member***, **Hiroki TAKAKURA**[††*b)], ***Nonmember***, **Yasuo OKABE**[††c)], **Daisuke INOUE**[†d)], **Masashi ETO**[†e)], ***and*** **Koji NAKAO**[†f)], ***Members***

SUMMARY Intrusion Detection Systems (IDS) have been received considerable attention among the network security researchers as one of the most promising countermeasures to defend our crucial computer systems or networks against attackers on the Internet. Over the past few years, many machine learning techniques have been applied to IDSs so as to improve their performance and to construct them with low cost and effort. Especially, unsupervised anomaly detection techniques have a significant advantage in their capability to identify unforeseen attacks, *i.e.*, 0-day attacks, and to build intrusion detection models without any labeled (*i.e.*, pre-classified) training data in an automated manner. In this paper, we conduct a set of experiments to evaluate and analyze performance of the major *unsupervised* anomaly detection techniques using *real traffic data* which are obtained at our honeypots deployed inside and outside of the campus network of Kyoto University, and using *various evaluation criteria*, *i.e.*, performance evaluation by similarity measurements and the size of training data, overall performance, detection ability for unknown attacks, and time complexity. Our experimental results give some practical and useful guidelines to IDS researchers and operators, so that they can acquire insight to apply these techniques to the area of intrusion detection, and devise more effective intrusion detection models.

key words: *intrusion detection system, unsupervised machine learning techniques, real traffic data, various evaluation criteria*

1. Introduction

Due to the proliferation of high-speed Internet access and local networks, more and more computer systems, networks and organizations are becoming vulnerable to potential cyber attacks, such as network intrusions. Intrusion detection systems [1], [2] have played an important role as a tool for effectively depending our crucial computer systems or networks against attackers on the Internet.

During the last decade, many machine learning and data mining techniques have been applied to IDSs so as to improve their performance and to construct them with low cost and effort. Especially, *unsupervised* anomaly detection techniques [3]–[8] have a significant advantage in their capability to identify unforeseen attacks, *i.e.*, 0-day attacks, and to build intrusion detection models without any labeled training data in an automated manner. This kind of IDS builds normal patterns on the networks and then attempts to detect deviations from normal patterns in observed data. Since new intrusions will be different from normal patterns, it has capability to detect new types of intrusions.

In order to provide useful guidelines for security researchers and operators, several approaches have been performed to evaluate unsupervised machine learning techniques in intrusion detection [9]–[11]. However, they have two fatal weaknesses which are unacceptable when they are deployed on a real environment. One is that their experiments were performed using KDD Cup 99 dataset, which is a simulated benchmark data, and the types of simulated attacks are too old. The main reason why they used KDD Cup 99 dataset is that it is quite problematic to obtain high-quality evaluation data due to privacy and competitive issues, since many organizations are not willing to share their data with other institutions. Even if real data were available, labeling traffic data as normal or intrusion requires enormous amount of time for many human experts. The other problem is that their evaluation mainly focuses on measurement of only performance, *i.e.*, the detection accuracy and the false positive rate. Since IDSs have to deal with a tremendous traffic data containing many noise and their performance heavily depends on not only algorithms, but also many other factors such as similarity measurements, the size of training data, these criteria also should be considered as the significant evaluation indicator.

To the best of our knowledge, [11] is the most recent and extensive comparative research of *unsupervised* machine learning techniques for intrusion detection. In this paper, we expand its evaluation scheme, and the main contributions of our approach are as follows. First, we evaluate and analyze 3 outlier detection algorithms [13]–[15], 4 clustering algorithms [3], [8], [16], [17] and one-class SVM [6]. Unlike [11], we evaluate two additional unsupervised machine learning techniques, *i.e.*, LOF [15] and DBScan [17] which are approaches based on the *density* between data instances. Further, in case of the clustering techniques, we consider only K-means and Single Linkage which are basic clustering methods and our clustering method [8] (including DBScan), because [8] shows the best performance compared with the others. Note that we do not consider to

Manuscript received January 4, 2010.
Manuscript revised May 28, 2010.

[†]The authors are with National Institute of Information and Communications Technology, Koganei-shi, 184–8795 Japan.
[††]The authors are with Academic Center for Computing and Media Studies, Kyoto University, Kyoto-shi, 606–8501 Japan.
[*]Presently, with Information Technology Center, Nagoya University, Nagoya-shi, 464–8601 Japan.
a) E-mail: song@nict.go.jp
b) E-mail: takakura@itc.nagoya-u.ac.jp
c) E-mail: okabe@i.kyoto-u.ac.jp
d) E-mail: dai@nict.go.jp
e) E-mail: eto@nict.go.jp
f) E-mail: ko-nakao@nict.go.jp
DOI: 10.1587/transinf.E93.D.2544

evaluate other machine learning techniques such as artificial neural networks and bayesian network which have high time complexity in general, because our experimental results for time complexity in Sect. 3.5 definitely indicate that machine learning techniques with high time complexity are unsuitable for intrusion detection where a tremendously large traffic data should be inspected in real time. Second, in order to perform the evaluation, we use real traffic data which are obtained at our honeypots deployed inside and outside of the campus network of Kyoto University. Third, we investigate how similarity measurements (Appendix) used by each machine learning technique influence to its performance. Finally, we compare performance of each machine learning technique by the size of training data, overall performance, detection ability for unknown attacks with time complexity.

Our key findings are:

- Euclidean and Chebyshev distances are better choice for intrusion detection.
- In case of one-class SVM, it is obvious that RBF kernel is superior to the others.
- Clustering is more suitable approach than the outlier detection approaches for intrusion detection, while one-class SVM is intermediate.
- Most of the existing unsupervised anomaly detection techniques (excluding LOF) have weakness for noisy data.
- DBScan and LOF have long training time, while the others show relatively short training time.
- The testing time of 3 outlier detection approaches is inferior to the other 5 approaches.

The rest of the paper is organized as follows. In Sect. 2, we describe unsupervised anomaly detection techniques briefly. In Sect. 3, we show experimental results and their analysis. Finally, Sect. 4 gives some conclusion and future work.

2. Unsupervised Anomaly Detection Techniques

2.1 Outlier Detection

Outlier detection techniques assign a score to each data instance which reflects how anomalous it is. In intrusion detection, high-scoring data instances are regarded as intrusion. Three main approaches for outlier detection have been proposed in the literature.

2.1.1 k-Nearest Neighbor (or *Kappa*)

This approach [13] computes $D_k(O)$ which represents the distance of a data instance O to the k-th nearest neighbor. The distance $D_k(O)$ is considered as a measure of the outlierness of data instance O. In other words, data instances with larger values $D_k(O)$ represent higher possibility of intrusion than data instances with smaller values $D_k(O)$.

2.1.2 Gamma

In case of *Kappa*, $D_k(O)$ considers the distance to only the k-th nearest neighbor, but it ignores the distances to the closer neighbors. *Gamma* [14] takes the distances to such all neighbors into account: $D_\gamma(O)$ is data instance O's average distance to its k nearest neighbors. Similar to *Kappa*, the distance $D_\gamma(O)$ of data instance O indicates its outlierness.

2.1.3 LOF

The key idea of this method [15] is to assign local outlier factor (LOF) to each data instance which represents its outlierness. Unlike the above two outlier detection algorithms, this algorithm considers the density between data instances. In this algorithm, LOF of a point O is weighted according to the density among other points around the point O. In other words, if the density among other points around the point O is higher, it lowers LOF of the point O.

2.2 Clustering

Clustering algorithms generate several clusters from a set of data, and after the clustering process they are labeled as either normal or attack according to a labeling heuristic. The most generic labeling heuristic labels each cluster according to its size, *i.e.*, the number of its members. If the size of a group is relatively larger, it is regarded as normal, otherwise regarded as attack. This labeling method is based on the fact that the ratio of normal data to attack data is extremely large in general. 3 clustering algorithms (K-means [16], Single Linkage [3], DBScan [17]) adopt this labeling method.

2.2.1 K-Means

K-means [16] is a typical clustering algorithm, and it partitions a set of data into k clusters through 3 steps. 1) It randomly chooses k data instances from dataset and makes them initial cluster centers. 2) It assigns each data instance to the closest center, and replaces every cluster's center with the mean of its members. 3) It repeats the processes between 1) and 2) until there is no change for each cluster, or other convergence criterion is met.

2.2.2 Song

Song [8] is a clustering method based on K-means. Its clustering process is basically the same as that of K-means, but it improves its performance by overcoming shortcomings of K-means. Furthermore, this approach adopts different labeling heuristic from the other 3 clustering algorithms. In this labeling process, it considers the maximum distance between a final cluster center and the initial cluster centers whose majority of the members consist of normal data instances. If the maximum distance between the final cluster

centers and the initial cluster centers is larger than its average distance for all final cluster centers, the corresponding cluster is labeled as attack. Otherwise it is labeled as normal.

2.2.3 Single Linkage

This algorithm [3] starts with an empty set of clusters. For each new data instance, it computes the distance between it and current all cluster centers, and the cluster with the shortest distance is selected. If that distance is less than a threshold W, then the data instance is assigned to that cluster. Otherwise, the data instance is farther away than W from all clusters, and then a new cluster is created with the data instance as its center.

2.2.4 DBScan

Similar to LOF, DBScan [17] relies on a density-based notion of clusters which is designed to discover clusters of arbitrary shape in spatial databases with noise.

2.3 One-Class SVM

Given a training data, one-class SVM [6] finds out a hypersphere where most of the data instances (*i.e.*, majority) are inside the hypersphere, and regards the inside of the hypersphere as a normal area. In the testing phase, it labels a data instance as intrusion if the data instance is outside the hypersphere, otherwise, labels the data instance as normal.

3. Experimental Results and Analysis

3.1 Data Description

In [18], we have deployed several types of honeypots on the 5 different networks which are inside and outside of Kyoto University: 1 class A and 4 class B networks. For example, there are some Windows base honeypots with Windows XP with SP2, full patched Window XP, Windows XP without any patch, Windows Vista, and symantec honeypot with Solaris, network printer, home appliance, *e.g.*, TV, Video Recorder and so on. In addition to traditional honeypots which only receive attacks, we deployed proactive systems which access to malicious web servers and join botnets to receive various types of commands. We collected all traffic data to/from the honeypots, and observed that most of them are attack data. For the collected traffic data, we investigated each connection whether there was buffer overflow attack or not. If the attack existed, a shellcode and an exploit code were identified by using the dedicated software [19]. Since Aug. 4th, 2005, we have assigned ID to each shellcode/exploit code. We also used IDS alerts obtained by SNS7160 IDS system [20] and malware information obtained by ClamAV [21] as extra information for the inspection. By using these diverse information, we thoroughly inspected the collected traffic data, and identified what happened on the networks.

In spite of this effort for identifying attacks over the networks, there is a possibility that unidentified attacks are being contained in the honeypot traffic data. However, in our investigation, we observed that the almost all honeypot traffic data used for our experiments were classified into attack data and there were few unidentified traffic data. Therefore, we regarded all the original honeypot traffic data as attack data and used all of them for our evaluation data, because performance of each machine learning algorithm is almost unaffected by a small amount of unidentified traffic data or they can be treated as just noisy data. Furthermore, it does not matter to keeping unidentified traffic data within the evaluation dataset from viewpoint of that all machine learning algorithms are fairly evaluated under the same condition (*i.e.*, with the same unidentified traffic data) even if unidentified traffic data are scattered in the evaluation dataset.

On the other hand, since the most of the honeypot traffic data are attack data, we should prepare a lot of normal data in order to evaluate performance of each algorithm fairly and effectively. To this end, we deployed a mail server for generating the normal traffic data, and regarded the traffic data recorded by it as normal data. The mail server was also operated with several communication protocols, *e.g.*, ssh, http and https, for its management and also received various attacks. Although all of these activities were included with the traffic data, they do not affect to performance of machine learning techniques considered in our experiments due to their small amount.

3.2 Evaluation Process of Intrusion Detection Models

Figure 1 shows the evaluation process of each machine learning algorithm for intrusion detection. The evaluation process is composed of two phases: training phase and testing phase. The training phase is summarized as follows.

1. **Collecting**: collect raw traffic data from the Internet. In our experiments, we used two sets of the traffic data,

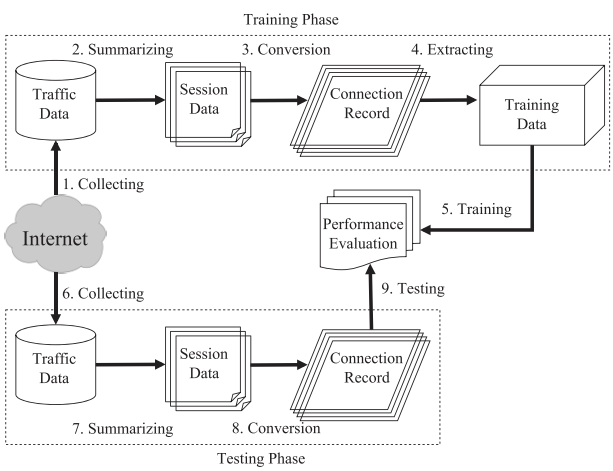

Fig. 1 Evaluation process of machine learning algorithms based on unsupervised anomaly detection.

a day (Nov. 2, 2007) and three days (Nov. 1–3, 2007) as the training data.

2. **Summarizing**: summarize collected raw traffic data in session data in that each session consists of its start time, duration, source address and port, destination address and port, and so on. We used Bro [22] for this summarizing process.
3. **Conversion**: convert summarized session data into connection records, which consist of 14 statistical features. These 14 statistical features were selected by the evaluation results of Mukkamala, et al. where they verified that many of the original 41 features of KDD Cup 99 dataset are duplicate and redundant [23], and they observed that only 14 features are most significant and essential. In addition to 14 fundamental features, we also appended 3 additional features to our benchmark data for further analysis by anyone; shellcode ID, malware identification and IDS alerts. Our benchmark data are opened to the public at [24], and visit our web site for more detail.
4. **Extracting**: extract the training data from converted connection records. The ratio of attack to normal in the training data should be small, because most of the traffic data observed on the networks are normal data generally. However, we collected traffic data through honeypot networks, so that the attack ratio was unnaturally high, *i.e.*, 80% on average. Similar to KDD Cup 99 dataset, therefore, we adjusted the attack ratio to 1%. As a result, 58,294 and 209,467 data instances were randomly and fairly selected from each set of the training data.
5. **Training**: train each unsupervised machine learning algorithm using the extracted training data, and thus the corresponding IDS model is constructed.

As the testing data, we prepared the traffic data of 10 days: December 1st, 8th, 15th and 22nd 2007, January 10th, 17th and 23rd 2008, February 9th, 16th and 23rd 2008. Also, in order to evaluate the detection capability for 0-day attacks, we prepared the traffic data of 5 days in which unknown attacks were observed: December 21st, 2007, January 30th 2008, February 7th, 10th and 11th 2008. In the testing phase, we applied the two processes, *i.e.*, 7 and 8 in Fig. 1, which are the same to those of the training phase to the traffic data of 15 days, and consequently obtained connection records with 14 statistical features. After that, we fed connection records of the testing data into IDS models which are built in the training phase, and then we evaluated each IDS model according to their detection mechanism.

In order to provide information regarding complexity of generating the evaluation dataset, we examined the processing time of the summarizing and conversion processes with respect to the original training data and testing data, and the results are shown in Table 1. This measurement was performed on the machine with Intel Xeon Quad-core 3.8 GHz CPU and 2 GB RAM. From Table 1, we can see easily that the size of raw traffic data used for generating our evaluation dataset is between 431 MB and 887 MB, and the generating processes of session data and connection records could be done within several minutes and tens of seconds, respectively.

Table 1 Time complexity of generating the evaluation dataset and the size of raw traffic data.

Date	Summarizing time (Sec.)	Conversion time (Sec.)	Size (MB)
Nov. 1, 2007	372	42	576
Nov. 2, 2007	375	29	493
Nov. 3, 2007	381	63	571
Dec. 1, 2007	375	33	654
Dec. 8, 2007	84	18	555
Dec. 15, 2007	20	19	442
Dec. 22, 2007	30	27	431
Jan. 10, 2008	373	43	447
Jan. 17, 2008	52	34	524
Jan. 23, 2008	49	46	762
Feb. 9, 2008	376	32	708
Feb. 16, 2008	371	34	671
Feb. 23, 2008	28	34	434
Dec. 21, 2007	25	26	479
Jan. 30, 2008	26	35	475
Feb. 7, 2008	380	34	887
Feb. 10, 2008	377	30	669
Feb. 11, 2008	432	31	741

3.3 Evaluation of Similarity Measurement

In case of the existing unsupervised machine learning approaches for intrusion detection, they mainly apply Euclidean distance to measure similarity between the data instances to be learned and tested. Euclidean distance can be regarded as a general selection for that, but it does not become the best choice in any situation. In our evaluation, we compare Euclidean distance with other 3 similarity measurements, *i.e.*, Manhattan, Canberra and Chebyshev distances, with respect to 4 clustering algorithms and 3 outlier detection algorithms. Since one-class SVM uses a kernel function for measuring similarity, we compare Polynomial and Sigmoid kernel functions with RBF kernel function (Radical Basis Function or Gaussian Function) which is adopted in the previous researches. See Appendixes A.1 and A.2 for more detail.

For this comparison, we used the training data of 1 day and the testing data of 10 days, and the experimental results below are averaged on the results for 10 days. Figure 2 shows the comparison of ROC curves of 8 algorithms by similarity measurement. We only show the portion of the ROC curve between the false positive rates of 0–10%, because higher rates are unacceptable for intrusion detection. From Fig. 2, we can observe that each algorithm has its best similarity measurement. For example, in case of Song, Chebyshev distance is better than the others at the false positive rate below 7%, whereas Euclidean distance is superior at the higher false positive rate. It seems that Euclidean and Chebyshev distances are better choice for intrusion detection from view point of overall performance. Furthermore,

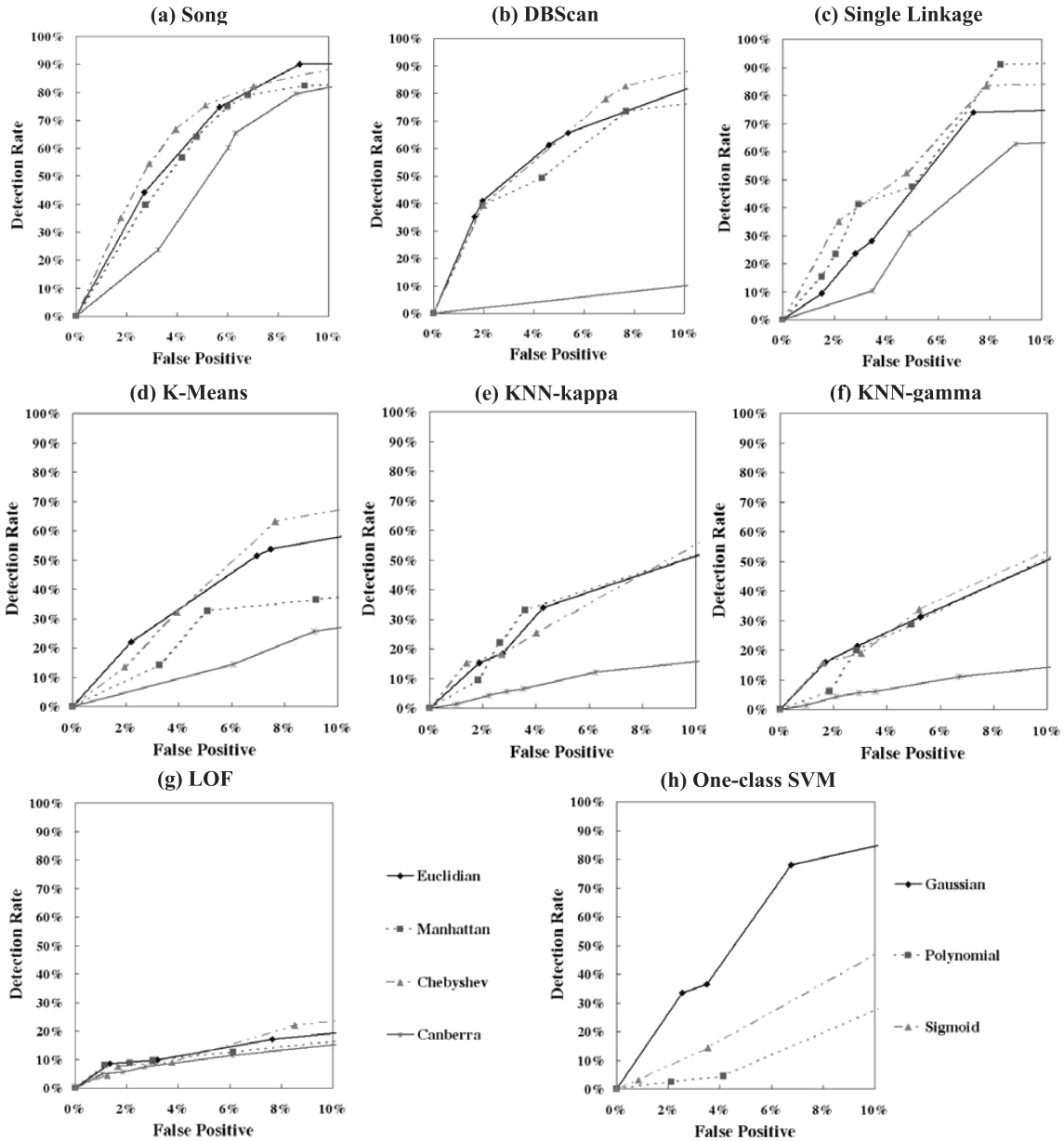

Fig. 2 ROC curves for each algorithm according to similarity measurement.

it is easy to see that Canberra distance is not suitable for similarity measurement. In case of one-class SVM, it is obvious that RBF kernel is superior to the others. Note that the following our experimental results focus on the evaluation for only two similarity measurements; Chebyshev distance and RBF kernel function. Due to space limitations, only the results for Chebyshev distance are shown here, but similar results were also observed for Euclidean distance.

These comparison results give us the following two points. One is when we devising a new distance measurement for intrusion detection, we should take account of the notable feature. If two data instances have a highly different value at a certain dimension, then their distance, *i.e.*, similarity should be dominated by that dimension. Euclidean and Chebyshev distances weight to such a dimension, that is, the squared value in Euclidean distance and the maximum value in Chebyshev distance are emphasized. The distance between two data instances, therefore, becomes quite long. While in case of the rest, they regard the values of all dimension as the same weight. The other is useful information provided by one-class SVM which transforms the traffic data with many noise data to follow the normal distribution. Therefore, if one devises an intrusion detection model based on machine learning techniques, particularly probability theory, he/she should consider this characteristic of the traffic data.

3.4 Performance Evaluation

3.4.1 Performance by Data Size

Since the attack techniques are changing every day, and the network situation is also always changing, these unstable conditions may heavily influence performance of machine learning techniques. In order to cope with these issues, it is required to construct an intrusion detection model trained by the training data as large as possible, and to update it periodically, so that it is able to cover a wide range of ongoing attacks and normal patterns. However, there is also a problem that the noisy data also increase with the expansion of the size of the training data. Therefore, it is worthwhile investigating whether each machine learning algorithm is robust or not to the noisy data.

In this experiment, we evaluated performance of 8 algorithms by the size of the training data, *i.e.*, 1 day and 3 days, and obtained their ROC curves as shown in Fig. 3. From Fig. 3, we can see that performance of most of the algorithms (excluding LOF) is decreased with the increase of the training data. This is because the noisy data also grow with the increase of the training data as mentioned above, even though the range of the normal patterns and the attack patterns which are covered by the training data is extended.

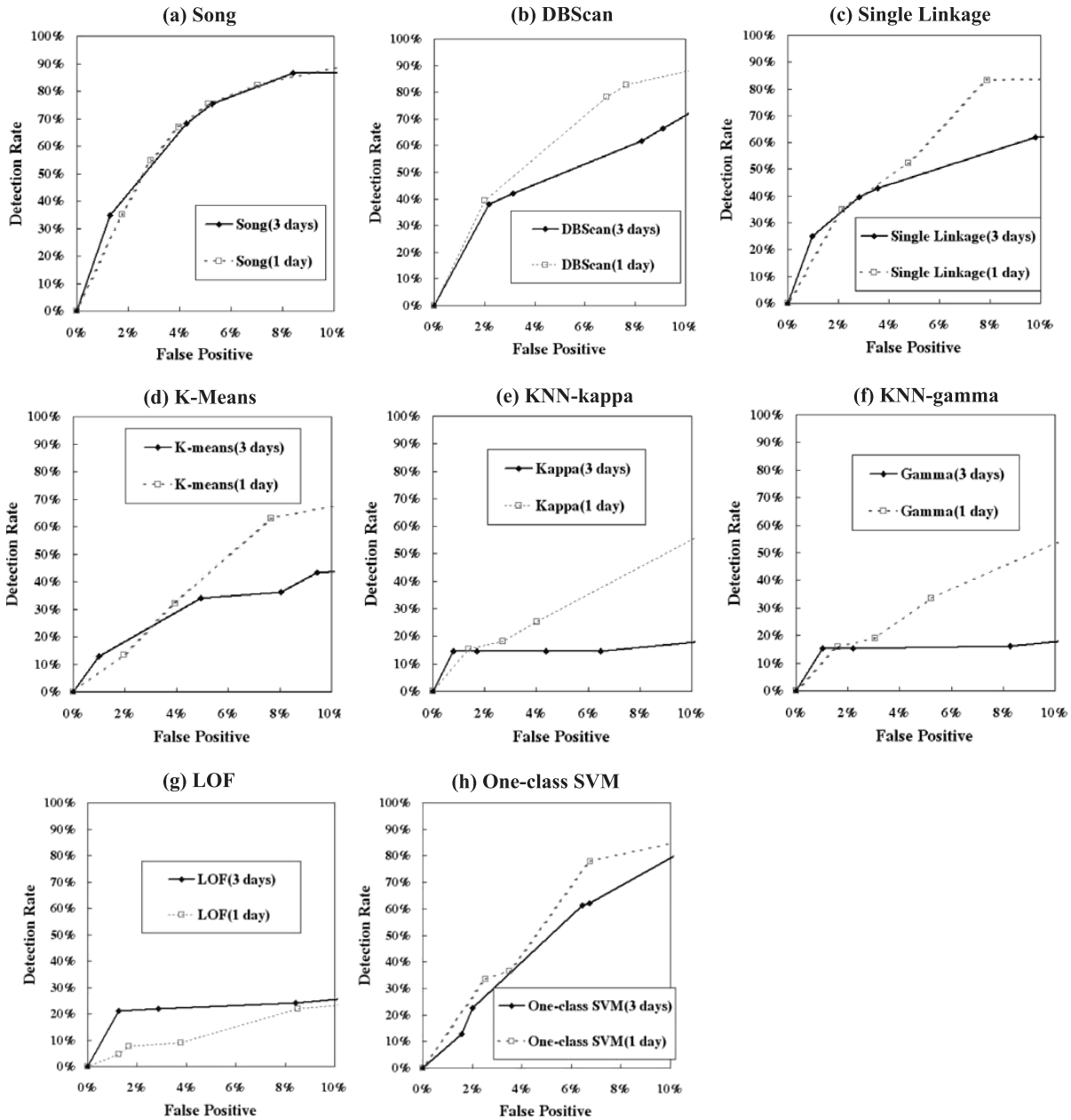

Fig. 3 ROC curves for 8 algorithms according to two training data (1 day and 3 days) under Chebyshev distance.

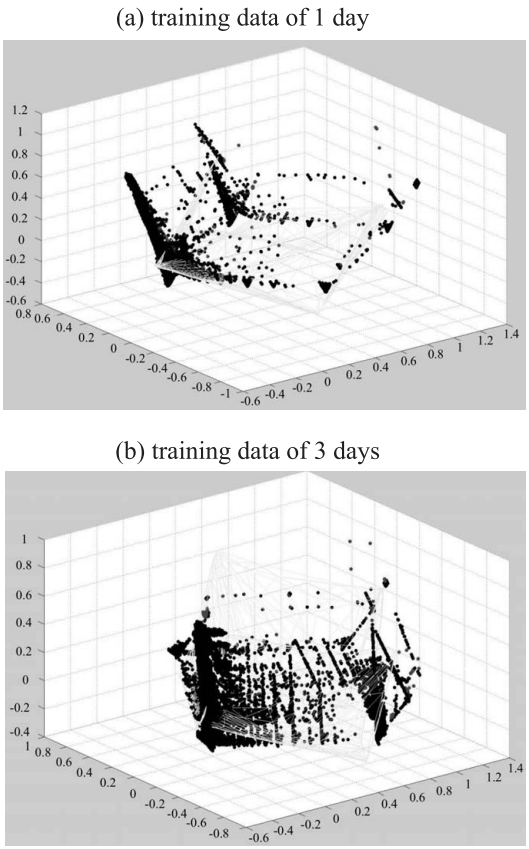

Fig. 4 3 dimensional projection by PCA. Each black point indicates a data instance.

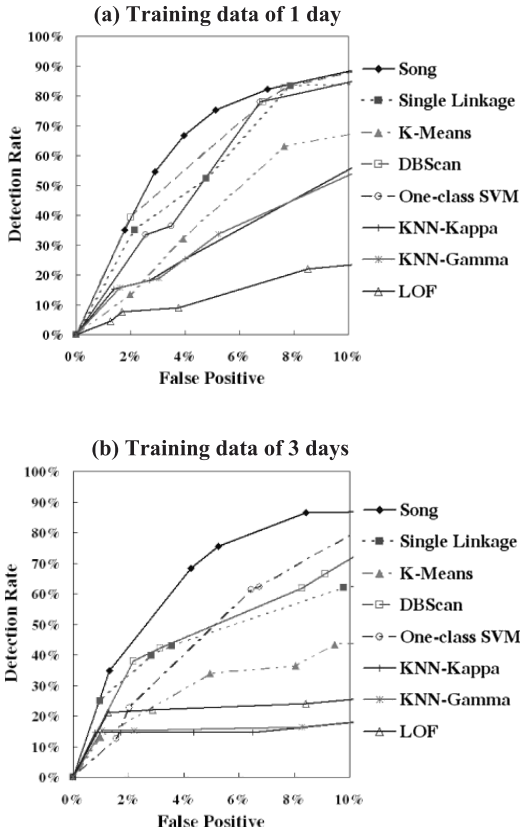

Fig. 5 Overall performance for each algorithm.

In fact, Fig. 4 shows the results that our training data of 1 day and 3 days are projected into 3 dimensional space by PCA (Principle Components Analysis). From Fig. 4, we can observe that the training data of 3 days are more complex (*i.e.*, more noisy) than that of 1 day. In other words, in case of the training data of 3 days, more data points do not exist on the line of coordinates than those of the training data of 1 day. Therefore, it needs to devise unsupervised anomaly detection techniques that are robust to the noisy data, and make the best use of the extended normal patterns. Because long-term training data enable one to construct more stabile intrusion detection model. Note that although performance of LOF becomes better with respect to the expansion of the training data, its performance is still low.

3.4.2 Overall Performance

Figure 5 shows the overall performance of each machine learning algorithm by utilizing 10-day testing data. Comparing (a) and (b) of Fig. 5, we can see that the clustering base approaches outperform the outlier detection base approaches, whereas one-class SVM is intermediate. This can be analyzed as follows. In case of the outlier detection base approaches, their performance heavily depends on false positives and false negatives which are misclassified during the training phase. If a data instance is classified as false positive during the training phase, all data instances of the testing data which are closest to the data instance are also misclassified as false positive. On the other hand, the clustering base approaches classify each data instance of the testing data by comparing with the clusters' centers which are labeled as attack or normal during the training phase. Therefore, they become robust to a small amount of false positives and false negatives, because the clusters' centers are averaged by their members.

Now, we have to give an attention to the existing researches which show quite different evaluation results from our ones. Eskin et al. [4] present three algorithms, a fix-width clustering technique, K-nearest neighbor and one-class SVM, and evaluate them by performing experiments on KDD Cup 99 dataset [12]. This evaluation shows that all three algorithms perform relatively close to each other. Lazarevic et al. [9] compare several distance-based and density-based outlier detection schemes as well as one-class SVM. Their evaluation results demonstrate that the most promising technique for detecting intrusions is the Local Outlier Factor (LOF) approach. In [11], they provide the comparative results of several clustering methods, K-nearest neighbor [13] and one-class SVM [6]. This comparative study shows that one-class SVM [6] and Y-means [5] clustering method have better performance than the others on average. As a summary, it can be considered that performance of the outlier detection approaches is comparable to

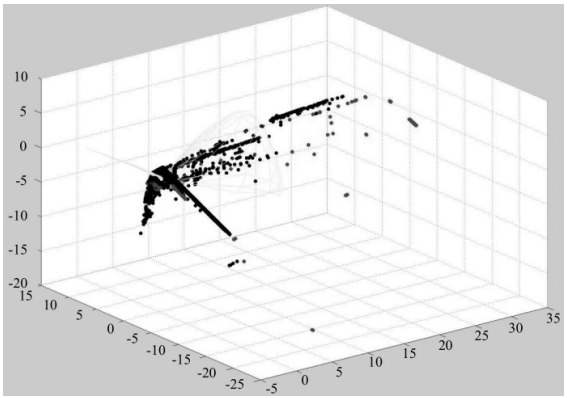

Fig. 6 3 dimensional PCA projection for KDD Cup 99 dataset. Each black point indicates a data instance.

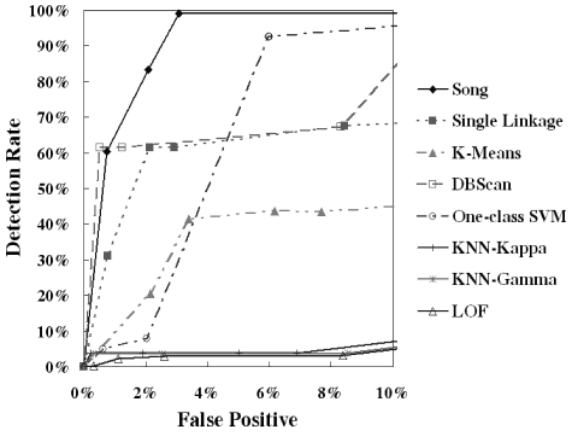

Fig. 7 ROC curves for unknown attacks.

others.

In order to clarify the reason of these different evaluation results, we obtained the PCA results from KDD Cup 99 dataset which was used in the existing approaches as shown in Fig. 6. KDD Cup 99 dataset has the almost same size of our training data as 1-day dataset. Therefore, comparing (a) of Figs. 4 and 6, it is easy to see that KDD Cup 99 dataset can be projected by only 3 principle components, and the noisy data within KDD Cup 99 dataset are extremely small compared with the real data. Therefore, it can be considered that the outlier detection algorithms show better performance when they were applied to KDD Cup 99 dataset than the real data, because KDD Cup 99 dataset consists of almost pure data instances, and consequently there was little influence by false positives to performance of the outlier detection algorithms.

3.4.3 Performance for Unknown Attacks

Figure 7 shows ROC curves of 8 algorithms for unknown attacks. For this evaluation, we used the training data of 3 days and the testing data of 5 days where unknown attacks were observed by our analysis. From Fig. 7, we can observe that similar results with Fig. 5 are obtained.

3.5 Evaluation of Time Complexity

Time complexity of each technique is one of the most significant indicators to determine whether IDS can adopt the technique in a real environment. The following measurement was performed on the machine with Intel Xeon 3.8 GHz CPU and 3.5 GB RAM. Figure 8 (a) shows the training time of each approach where 1 day (58,294 instances) and 3 days (209,467 instances) training data are used, and we can easily observe that DBScan and LOF consume long training time, *i.e.*, around 3 hours and 18 hours for 1 day and 3 days training data, respectively. Meanwhile the others show relatively short training time. Figure 8 (b) shows the testing time of each algorithm, and we can see that 3 outlier detection algorithms have long testing time. This is because they compare the distance between all data

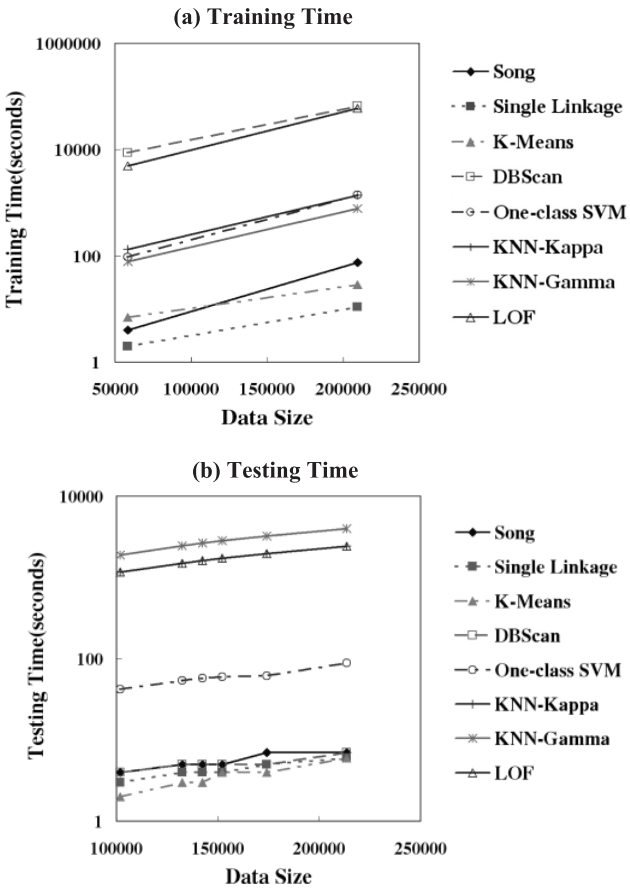

Fig. 8 Training time and testing time.

instances within the training data and the testing data one by one. While the others have relatively short detection time. In case of one-class SVM, its short detection time is obtained from the fact that it only uses several support vectors during the testing phase.

Our testing data were extracted from the original traffic data without any adjustment. Each day, we observed from 14.1 to 60.1 pps (packet per second) on average. Maximum and minimum ppses were 1,078 (2007-12-01 03:01:08) and

0 (many times), respectively. On the other hand, our training data were modified for adjusting the ratio of attack and normal data. Although we could not calculate accurate pps of them, they shown similar trend of our testing data.

These pps values of our training and testing data demonstrate that machine learning techniques with high time complexity are unpractical for intrusion detection. In general, traffic data to be inspected in the network of organizations such as enterprises and universities are extremely larger than our experimental data. In this paper, therefore, we focus only on light weight algorithms to evaluate their performance.

4. Conclusion and Future Work

In this study, we have conducted a set of blind experiments of unsupervised anomaly detection techniques using honeypot data. Our main goal is to provide practical and useful guidelines to security researchers and operators. Our experimental results show that

- Each algorithm has its best similarity measurement. Users have to choice an optimized method for similarity measurement with respect to the corresponding algorithm. It seems that Euclidean and Chebyshev distances are better choice for intrusion detection from view point of overall performance. Canberra distance is not suitable for comparing similarity between the data instances. In case of one-class SVM, it is obvious that RBF kernel is superior to the others.
- The clustering base approaches are more suitable than the outlier detection base approaches for intrusion detection from viewpoint of overall performance, detection capability for unknown attacks and time complexity, whereas one-class SVM is intermediate.
- Most of the existing unsupervised anomaly detection techniques (excluding LOF) have weakness for noisy data. Although performance of LOF becomes better with respect to the expansion of the training data, its performance is still low.
- Outlier detection base approaches are not suitable for intrusion detection. Because their performance heavily depends on the structural complexity of benchmark data.
- DBScan and LOF have long training time, while the others show relatively short training time.
- The testing time of 3 outlier detection approaches is inferior to the other 5 approaches.

For future work, first of all, we need to verify performance of each machine learning technique over more various and larger real data, so that we can find out the suitable updating interval and the optimized size of the training data. However, such experiments are time-consuming, and this should be treated as an individual research topic. Secondly, in case of unsupervised techniques, they require several parameters during the training and the testing phases. For example, K-means has to set the number of clusters, k, to be created finally before it starts the clustering process. In many cases, their performance heavily depends on the values of parameters used. Therefore, it needs to investigate whether they are sensitive or not to parameters. Finally, we would like to expand our evaluation with respect to more various attack ratio of the training data, *e.g.*, the attack ratio of 0.5%, 2%, 5%. even though the situation where the ratio of attack data to normal data is actually extremely small makes our evaluation to reasonable. Because in a real world, there may exist some organizations where the ratio of the attack data is unpredictable, *i.e.*, higher than 1% or lower than 1%.

References

[1] D.E. Denning, "An intrusion detection model," IEEE Trans. Softw. Eng., vol.SE-13, no.2, pp.222–232, 1987.
[2] R. Bace and P. Mell, "Intrusion detection systems," NIST Special Publications SP 800-31, Nov. 2001.
[3] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion detection with unlabeled data using clustering," Proc. ACM CSS Workshop on Data Mining Applied to Security, 2001.
[4] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A geometric framework for unsupervised anomaly detection: Intrusion detection in unlabeled data," Applications of Data Mining in Computer Security, 2002.
[5] Y. Guan, A. Ghorbani, and N. Belacel, "Y-means: A clustering method for intrusion detection," Proc. IEEE Canadian Conference on Electrical and Computer Engineering, 2003.
[6] K.L. Li, H.K. Huang, S.F. Tian, and W. Xu, "Improving one-class SVM for anomaly detection," International Conference on Machine Learning and Cybernetics, vol.5, pp.3077–3081, 2003.
[7] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," ACSC2005, 2005.
[8] J. Song, K. Ohira, H. Takakura, Y. Okabe, and Y. Kwon, "A clustering method for improving performance of anomaly-based intrusion detection system," IEICE Trans. Inf. & Syst., vol.E91-D, no.5, pp.1282–1291, May 2008.
[9] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," Proc. Third SIAM International Conference on Data Mining, 2003.
[10] P. Laskov, P. Düssel, C. Schäfer, and K. Rieck, "Learning intrusion detection: Supervised or unsupervised?," ICIAP 2005, LNCS 3617, pp.50–57, 2005.
[11] R. Sadoddin and A.A. Ghorbani, "A comparative study of unsupervised machine learning and data mining techniques for intrusion detection," MLDM 2007, LNAI 4571, pp.404–418, 2007.
[12] The third international knowledge discovery and data mining tools competition dataset KDD99-Cup, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999.
[13] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," Proc. ACM SIGMOD Conference, 2000.
[14] S. Harmeling, G. Dornhege, D. Tax, F. Meinecke, and K. Muller, "From outliers to prototypes: Ordering data," Neurocomputing, vol.69, no.13-15, pp.1608–1618, 2006.
[15] M.M. Breunig, H.P. Kriegel, R.T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," SIGMOD Rec., vol.29, no.2, pp.93–104, 2000.
[16] J. MCQUEEN, "Some methods for classification and analysis of multivariate observations," Proc. Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp.281–297, 1967.
[17] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise,"

Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, AAAI Press, Portland, Oregon, 1996.
[18] J. Song, H. Takakura, and Y. Okabe, "Cooperation of intelligent honeypots to detect unknown malicious codes," WOMBAT Workshop on Information Security Threat Data Collection ans Sharing (WISTDCS 2008), pp.31–39, The IEEE CS Press, April 2008.
[19] http://www.secure-ware.com/contents/product/ashula.html
[20] Symantec Network Security 7100 Series.
[21] http://www.clamav.net/
[22] http://www.bro-ids.org/
[23] S. Mukkamala and A.H. Sung, "Identifying significant features for network forensic analysis using artificial intelligent techniques," International Journal of Digital Evidence, vol.1, no.4, 2003.
[24] http://www.takakura.com/Kyoto_data/

Appendix: Similarity Measurement

A.1 Distance Measurement

Given a pair of objects, $\mathbf{a} = \{a_1, a_2, \ldots, a_n\}$, $\mathbf{b} = \{b_1, b_2, \ldots, b_n\}$, the distance between $\mathbf{a}$ and $\mathbf{b}$, $d(\mathbf{a}, \mathbf{b})$, is as follows.

- Euclidean distance:

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \quad (\text{A}\cdot 1)$$

- Manhattan distance:

$$d(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} |a_i - b_i| \quad (\text{A}\cdot 2)$$

- Canberra distance:

$$d(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} \frac{|a_i - b_i|}{|a_i| + |b_i|} \quad (\text{A}\cdot 3)$$

- Chebyshev distance:

$$d(\mathbf{a}, \mathbf{b}) = \max_i |a_i - b_i|. \quad (\text{A}\cdot 4)$$

A.2 Kernel Function

Given a pair of objects, $\mathbf{a} = \{a_1, a_2, \ldots, a_n\}$, $\mathbf{b} = \{b_1, b_2, \ldots, b_n\}$, $K(\mathbf{a}, \mathbf{b})$ between $\mathbf{a}$ and $\mathbf{b}$ is as follows.

- Polynomial kernel:

$$K(\mathbf{a}, \mathbf{b}) = (a^T b + 1)^d \quad (\text{A}\cdot 5)$$

- RBF kernel:

$$K(\mathbf{a}, \mathbf{b}) = \exp\left(\frac{-\| a - b \|^2}{2\sigma^2}\right) \quad (\text{A}\cdot 6)$$

- Sigmoid kernel:

$$K(\mathbf{a}, \mathbf{b}) = \tanh(k a^T b + \theta). \quad (\text{A}\cdot 7)$$

Jungsuk Song received his B.S. and M.S. degrees in Information and Telecommunication Eng. from Korea Aerospace University, Korea in 2003 and 2005, respectively. He received his Ph.D. degree in the Graduate School of Informatics, Kyoto University, Japan in 2009. He is currently an expert researcher at National Institute of Information and Communications Technology (NICT), Japan. His research interests include network security, data mining, machine learning, security issues on IPv6, spam analysis, and cryptography theory. He is a member of IEEE.

Hiroki Takakura received his B.S. and M.S. degrees from Kyushu University in 1990, and 1992, and D.Eng. degree from Kyoto University in 1995. He was a research fellow of Japan Society for Promotion of Science since 1994 to 1995, research associate at Nara Institute of Science and Technology since 1995 to 1997, lecturer at Kyoto University since 1997 to 2000, associate professor at Kyoto University since 2000 to 2009. He is currently a professor at Nagoya University since 2010. He was also a visiting scholar at University Illinois at Urbana Champaign. His research interests include network security, databases, and geographic information system. He is a member of Information Processing Society, Japan; Geographic Information Systems in Japan; The Institute of Systems, Control and Information Engineers; and ACM.

Yasuo Okabe received the M.E. from Department of Information Science, Kyoto University in 1988. From 1988 he was an Instructor of Faculty of Engineering, from 1994 he was an Associate Professor of Data Processing Center, and from 1998 he was an Associate Professor of Graduate School of Informatics, Kyoto University. He is now a Professor of Academic Center for Computing and Media Studies, Kyoto University. Ph.D. in Engineering. His current research interest includes Internet architecture, ubiquitous networking and distributed algorithms. He is a member of IPSJ, ISCIE, JSSST, IEEE, ACM and EATCS.

Daisuke Inoue received his B.E. and M.E. degrees in electrical and computer engineering and Ph.D. degree in engineering from Yokohama National University in 1998, 2000 and 2003, respectively. He joined the Communications Research Laboratory (CRL), Japan, in 2003. The CRL was relaunched as the National Institute of Information and Communications Technology (NICT) in 2004, where he is a senior researcher of the Information Security Research Center. His research interests include security and privacy technologies in wired and wireless networks, incident analysis and response technologies based on network monitoring and malware analysis. He received the best paper award at the 2002 Symposium on Cryptography and Information Security (SCIS 2002), the best paper award at the 2nd and 3rd Joint Workshop on Information Security (JWIS 2007 and 2008), and the commendation for science and technology by the minister of MEXT, Japan, in 2009.

Masashi Eto received LL.B. degree from Keio University in 1999, received the M.E. and Ph.D. degrees from Nara Institute of Science and Technology (NAIST) in 2003, 2005, respectively. From 1999 to 2003, he was a system engineer at Nihon Unisys, Ltd., Japan. He is currently a researcher at National Institute of Information and Communications Technology (NICT), Japan. His research interests include network monitoring, intrusion detection, malware analysis and auto-configuration of the Internetworking. He received the best paper award at the 2007 Symposium on Cryptography and Information Security (SCIS 2007), the best paper award at the 2nd and 3rd Joint Workshop on Information Security (JWIS 2007 and 2008), and the commendation for science and technology by the minister of MEXT, Japan, in 2009.

Koji Nakao is the Information Security Fellow in KDDI, Japan. Since joining KDDI in 1979, he has been engaged in the research on multimedia communications, communication protocol, secure communicating system and information security technology for the telecommunications network. He is also an active member of Japan ISMS user group, which was established in the 1st Quarter of 2004. He is the board member of Japan Information Security Audit Association (JASA) and that of Telecom-ISAC Japan, and concurrently, a Technical Group Chairs (ICSS: information communication system security) of the Institute of Electronics, Information and Communication Engineers. He received the B.E. degree of Mathematics from Waseda University, in Japan, in 1979. He received the IPSJ Research Award in 1992, METI Ministry Award and KPMG Security Award in 2006, Contribution Award (Japan ITU), NICT Research Award, Best Paper Award (JWIS) and MIC Bureau Award in 2007. He is a member of IPJS. He has also been a part-time instructor in Waseda University since 2002.