An Automatic and Dynamic Parameter Tuning of a Statistic-based Anomaly Detection Algorithm

Yosuke Himura
The University of Tokyo
him@hongo.wide.ad.jp

Kensuke Fukuda
National Institute of Informatics /
PRESTO, JST
kensuke@nii.ac.jp

Kenjiro Cho
Internet Initiative Japan
kjc@iijlab.net

Hiroshi Esaki
The University of Tokyo
hiroshi@wide.ad.jp

***Abstract*—The detection of anomalies in network traffic is a crucial issue affecting the security of Internet users. A statistical network anomaly detection algorithm is a promising way of detecting such anomalies, however, it has to be given appropriate parameters for accurate detection and identification. In general, it is very difficult to obtain appropriate parameter settings a priori, because network traffic is not stable in time or space. Thus, although many anomaly detection methods have been proposed, there has been little discussion about their parameter tunings.**

In this paper, we investigate an automatic and dynamic parameter tuning of a statistical network traffic anomaly detection method. In particular, we clarify whether one can consistently use the best parameter fixed for a certain instance; this choice clearly depends on the macroscopic and dynamic behavior of Internet traffic anomalies. We ascertain the appropriate learning period for setting a parameter of an anomaly detection algorithm based on a sketch and multi-scale gamma-function model by using real network traces measured in a trans-Pacific link over a period of six months.

The main results of our study are as follows: (1) Without learning, the best parameter varies day by day. (2) With a longer learning period, the best parameter setting is affected by significant data during the learning period. (3) The appropriate period of the learning is about 3 days. (4) The performance degradation from introducing dynamic parameter tuning is 17% in the best case.

***Index Terms*—statistical network anomaly detection, parameter tuning**

I. Introduction

Nowadays, Internet users are exposed to many menaces like viruses, worms, and DDoS. To make networks secure for users, these menaces must be accurately detected in real-time. Many network traffic anomaly detection methods have been proposed, and they are broadly categorized into two approaches. One is the signature-based approach [1], which inspects fingerprints of packet payloads and compares them with ones in an anomaly database. This approach has high identification accuracy, but it has to have a set of corresponding signatures a priori. The other approach is based on statistical models [2]–[10]. It defines a network anomaly as a deviation from a referential statistical behavior of network traffic. The parameter setting of the statistical model directly affects to the accuracy of the detection and identification. Generally, however, it is very difficult to choose an appropriate parameter setting in advance, because the dynamics of macroscopic Internet traffic anomalies are highly variable depending on the time and location to be measured. For example, an outbreak of a new virus (zero-day attack) is a common situation in the current Internet. The parameter tuning is a difficult task, and little attention has been paid to it [11]. We need to obtain an appropriate parameter setting dynamically and automatically to be able to deploy algorithms that will be capable of accurate and real-time detection. Furthermore, automatic and dynamic parameter tuning is essential not only for detecting anomalies in the real world but also for comparing anomaly detection methods; for a fair comparison, we need both a parameter optimization and a performance measurement of each method using the same dataset.

In this paper, we discuss an automatic and dynamic parameter tuning of a statistical network traffic anomaly detection algorithm. In particular, we focus on the appropriate learning period for the best parameter setting. Intuitively, the learning period has a tradeoff: (a) A shorter learning period leads to low accuracy because there are fewer data points to get proper statistics. (b) A longer learning period will not capture abrupt changes in the macroscopic dynamics of network traffic anomalies. We investigate this tradeoff by using real traffic traces measured at a trans-Pacific link over the course of six months. Our contribution is to show the importance of automatic parameter tuning for statistical network traffic anomaly detection methods in the real world.

II. Dataset and anomaly detection algorithm

A. Anomaly detection algorithm based on sketch and multi-scale gamma-function model

We used an anomaly detection algorithm based on sketch and the multi-scale gamma-function model [2]. In this paper, we define “an event” as a set of packets which have the same source IP address, that is, a set of packets from the same host. We empirically focus on the events which have more than 1000 packets to obtain proper statistics for event classification (section III-A). The anomaly detection procedure has three steps.

(1) Sketch: traffic is divided into events with a hash function of a quasi-huge hash table created from several hash functions of small hash tables [8].

(2) Multi-scale gamma-function model: each event’s histogram of the number of packets arriving in a certain

TABLE I
CATEGORIES AND EXAMPLES OF HEURISTICS.

category	explanation	example of heuristics
Attack	The host sends malicious packets	If the ratio of SYN flagged packets is more than 20%, then the host is regarded as "an attacker of SYN flooding attack" and the event is classified into Attack category
Victim	The host receives malicious packets	If the ratio of SYN/ACK flagged packets is more than 20%, then the host is regarded be "a victim of SYN flooding attack" and the event is classified into Victim category
Warning	The host is legitimate, but it can be malicious in some cases	If the ratio of HTTP request packets is more than 50% then the host is regarded as "a sender of many HTTP requests" and the event is classified into Warning category
OK	The host is legitimate	If the ratio of packets with source port 80 is more than 50%, then the host is regarded as "a web server" and the event is classified into OK category
Special	The host is a server or client of a specific application such as DNS and FTP	If the ratio of packets with source port 53 is more than 50%, then the host is regarded as "a DNS server" and the event is classified into Special category
Unknown	The host is not classified into any above categories	If the host cannot be classified into any above categories, then the event is classified into Unknown category

timescale is approximated as a gamma distribution with several timescale. The gamma distribution has two parameters: α determines the shape of histogram, and β the scale. α is helpful for detecting low-intensity hidden anomalies.

(3) Anomaly identification: The αs of every event are compared each other, and events which have outlier parameters are anomalies; the same goes for the βs. For example, focusing an event's α, if $|\alpha - E[\alpha]| > \theta_\alpha$, the event is judged to be an anomaly. Here, $E[\alpha]$ stands for the average α among the events in the data and θ_α is the threshold for α.

Although the algorithm requires several parameters, we will concentrate on a main parameter θ_α because the change in the shape parameter clarifies low-intensity anomalies.

B. MAWI dataset

We performed our evaluation by using real traffic data taken from MAWI traffic repository [12]. The traffic traces were measured at a trans-Pacific link between Japan and the U.S. and consisted of 15 min. pcap traces (at 2 p.m. JST) from 2001. The payloads of the packets were removed and both IP addresses were anonymized; the prefix structure was preserved.

We chose traces from Oct. 2005 to Apr. 2006. The number and volume of traces were 207 and 113GB, respectively. Table II briefly describes the dataset. Note that the link was often congested during the observation period. These traces are not consecutive (15 min. for a day), but the measurement point and the start time (2 p.m.) of all traces were the same. Hence, these traces are good enough for investigating the algorithm's ability to follow the macroscopic changes in traffic anomalies.

TABLE II
BRIEF DESCRIPTION OF MAWI DATASET.

date	direction	link	packet rate	bit rate
From Oct. 2005 to Apr. 2006	Japan to US	18Mbps	4.27 Kpps	12.1 Mbps
	US to Japan	18Mbps	4.50 Kpps	15.0 Mbps

III. METHODOLOGY

A. Event classification

We define the optimal parameter (optimal threshold alpha) θ_α^{opt} as the parameter that leads to the best detection performance. The performance $a(\theta_\alpha)$ is expressed as $a(\theta_\alpha) = A/T$ where A and T are the number of anomalies and total number of events detected with the threshold θ_α. In other words, $1 - a(\theta_\alpha)$ is the false-negative rate and θ_α^{opt} minimizes it. Although this metric is different from the common performance metric (the false-positive rate), it is proper for deciding θ_α^{opt}. For the decision on θ_α^{opt}, we need to classify detected events; for computing $a(\theta_\alpha)$, we must identify which events are anomalous or not. Since no classification method is perfect, we must consider the worst case of classification (all unclassified events are anomalies) and the false-negative rate is a more robust metric for unclassified events.

We classified events into six categories (Attack, Victim, Warning, OK, Special, and Unknown) with our heuristics based on port number, TCP flag, and communication network structure. Table I shows an explanation of the categories and examples of heuristics. Here, we define the Anomaly category as including both Attack and Victim categories. Fig.1 shows the classification results on our dataset: (a) a breakdown by category and (b) a detailed breakdown of the Attack category. In this figure, there are aggregated statistics from Oct. 2005 to Apr. 2006. For example, Attack events account for 8.9%, and the most dominant attack is "scanning with SYN packets" (SYN_scan in Fig.1(b)) which accounts for 31.2% of Attack events. Note that the breakdown varies on a daily level. With these data, we will identify whether a detected event is a true anomaly or not.

B. Learning period

Learning is a method to obtain a parameter for appropriate detection on a certain day by considering past data. The number of events per day might be too small to give proper statistics to obtain θ_α^{opt} and high anomaly detection performance. Hence, we need to process data periods lasting

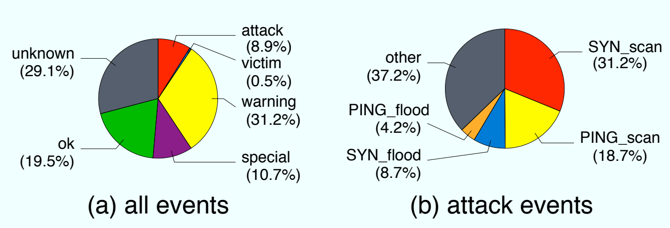

Fig. 1. Event breakdown of MAWI dataset (from Oct. 2005 to Apr. 2006): (a) all events, (b) attack events.

several days. The number of days of data for deciding θ_α^{opt} is called the “learning period τ”. Obviously, τ has a tradeoff:

- when τ is shorter, one cannot decide θ_α^{opt} because there are few data points to obtain proper statistics.
- when τ is longer, θ_α^{opt} is not suitable because a longer τ leads to an unexpected delay for following macroscopic changes of anomalies.

Thus, we need to find an appropriate value for τ.

C. Optimal parameter setting

The procedure to determine θ_α^{opt} is as follows.

1) The default optimal parameter is θ_α^{optdef} = 1.7.
2) Run the detection procedure over past τ days data by changing θ_α (increase by 0.1 from 0.0) in order to find a (local) maximum. Count A and T and compute $a(\theta_\alpha)$. θ_α^{opt} is uniquely determined to be $\theta_\alpha^{opt} = \arg\max a(\theta_\alpha)$ that satisfies $a(\theta_\alpha^{opt} - 0.1) < a(\theta_\alpha^{opt})$ and $a(\theta_\alpha^{opt} + 0.1) < a(\theta_\alpha^{opt})$.
3) If there is no θ_α satisfying the above condition, θ_α^{opt} of the previous day is used.

Fig.2 shows an example of the output by the above procedure. The x-axis is θ_α, and the y-axis is the ratio of detected events. Fig.2(a) and (b) are the results on Dec. 6th 2005 for 1 day and 28 days of learning, respectively. The performance of $a(\theta_\alpha)$ is the ratio of Attack (red) and Victim (blue) events over the total events.

- θ_α^{opt} in Fig.2(a) is around 4.4, because $a(\theta_\alpha^{opt}) = 100\%$. However, these parameters are not reasonable; the number of detected events is too small (only one event is detected and there are many missed events). Hence, choosing a local maximum is better for keeping a reasonable number of detected events; Here, we have one at $\theta_\alpha = 1.8$, and four anomaly events are detected. Conversely, only one anomaly event is detected when $\theta_\alpha > 2.0$.
- On the other hand, $\theta_\alpha = 2.4$ in Fig.2(b) is optimal for our heuristics. Thus, a large amount of learning data makes it easier to decide on a good θ_α^{opt}.

In addition, we found that the typical time-series pattern of events detected by using a tuned α is continuous; Fig.3(a) shows an example. On the other hand, there are only a few spiky traffic patterns (e.g., in Fig.3(b)); a tuned β would detect more.

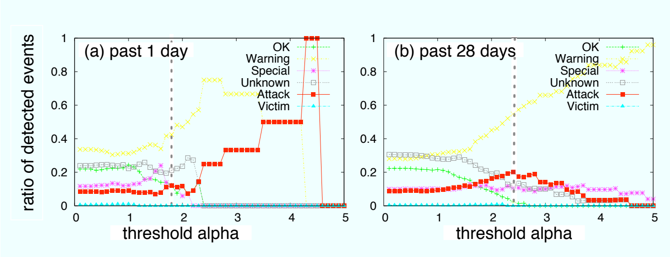

Fig. 2. Example of determining the optimal parameter θ_α^{opt} : (a) τ = 1-day learning period, (b) τ = 28-day learning period.

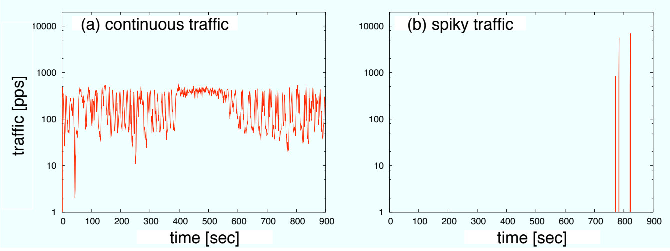

Fig. 3. Example of anomaly traffic detected by tuned alpha: (a) typical traffic, (b) untypical traffic. Both traffic are judged as “SYN flooding attack”.

IV. Results

A. Changes in optimal parameter

Fig.4 shows the dependency of θ_α^{opt} on date for fixed learning periods. The x-axes in Fig.4(a), (b) and (c) are date (from Nov. 2005 to Apr. 2006) and the y-axes are θ_α^{opt} for each date. The x-axes in Fig.4(d), (e) and (f) are date, and the y-axes are the number of detected events, with θ_α^{opt} determined from the above figures (red: anomaly events, green: normal events). Fig.4(a) and (d) show the results for no parameter learning; each θ_α^{opt} on a certain day yields to the best performance of the day. Fig.4(b) and (e) show the results for τ = 3-day of learning, and Fig.4(c) and (f) show the results for τ = 28-day of learning.

Obviously, θ_α^{opt} depends on τ; θ_α^{opt} with a shorter τ is scattered, whereas θ_α^{opt} with a longer τ is high and stable. One plausible reason for this result is that data for a specific day (or specific days) have a strong influence on θ_α^{opt}. Thus, since the learning with a longer τ is affected by specific data for a long period, it produces a stable θ_α^{opt} as long as learning of past τ days includes the data. In addition, θ_α^{opt} with a longer τ is inaccurate, because it degrades the performance $a(\theta_\alpha^{opt})$ (Fig.4(f)). The reason is that one cannot follow the changes in anomaly traffic with a longer τ. Thus, learning with a longer τ is inappropriate for obtaining a suitable θ_α^{opt} for practical anomaly detection.

B. Variable learning period

We investigated the strong influence from specific data by computing the best performance with a variable τ. Fig.5 shows the result; the x-axis in Fig.5(a) is date, and the y-axis is τ', which leads to the best performance for the past τ days of detection, i.e., $\tau' = \arg\max_\tau a(\theta_\alpha^{opt})$. The x-axis in Fig.5(b)

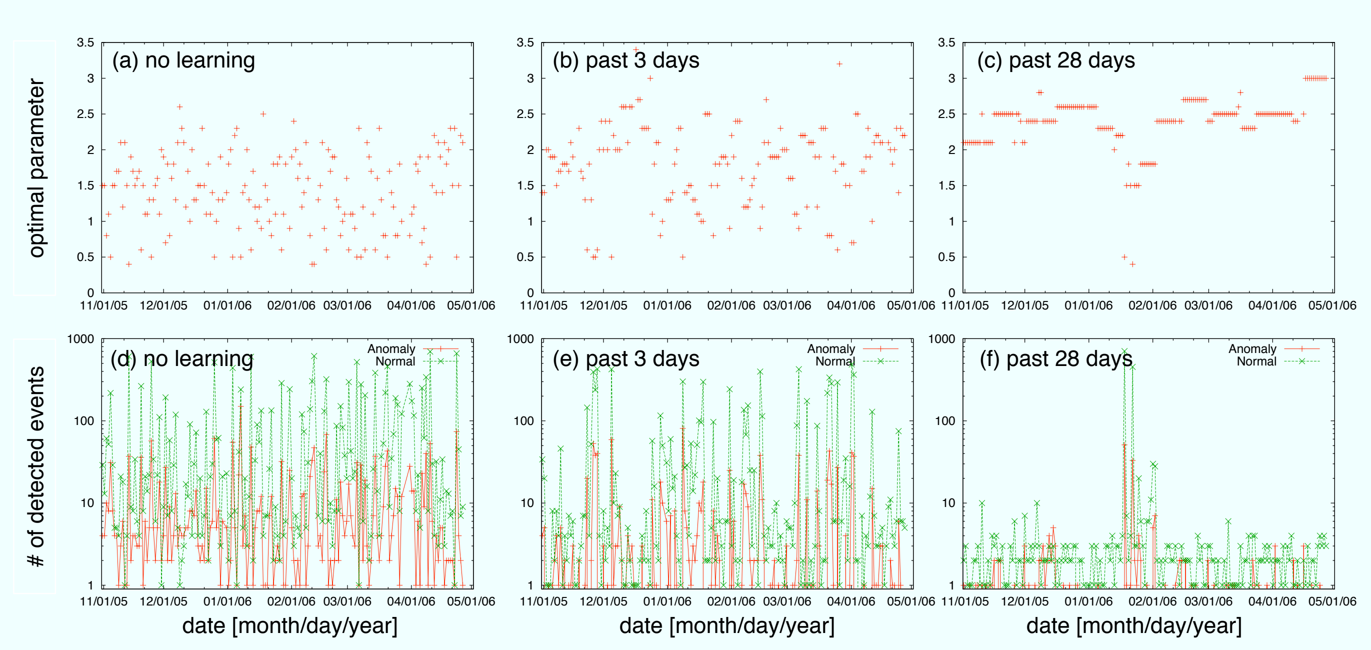

Fig. 4. Changes in θ_α^{opt}: (a) and (d) no learning, (b) and (e) past τ = 3-day learning, (c) and (f) past τ = 28-day learning.

is date and the y-axis is the θ_α^{opt} for the best τ'. Fig.5(a) has several lines, which means that some data have strong influences on θ_α^{opt}, and the learnings of τ' days consider those data. To investigate the reason for this effect, let us consider points A and B which are two of the roots of these lines.

A: The data for 2005-12-09 (point A) give $\theta_\alpha^{opt} = 2.6$. With this threshold, the detected events on the data are 5 Attack[1] and 1 P2P events.

B: The combination of the data from the Feb. 18th and 19th 2006 sets $\theta_\alpha^{opt} = 2.5$. This threshold produces 3 Attack[2], 4 Warning, and 1 Unknown events.

Since such data have a strong effect on $a(\theta_\alpha^{opt})$, a variable τ is inappropriate for following the macroscopic changes in Internet traffic. In addition, the changes in θ_α^{opt} for variable τ (Fig.5(b)) are almost same as those for τ = 28-day (Fig.4(f)), and the performance of a variable τ' is lower than that of τ = 3. Thus, τ must be appropriately fixed. Another way to avoid the influence of specific data is putting moving weights on the past data. However, this method needs a number of parameters (e.g., we need to know how to determine the weights).

C. Optimal learning period

We applied our automatic and dynamic parameter tuning method to the a real dataset of traces from Nov. 2005 to Apr. 2006. Fig.6 shows the dependency of $a(\theta_\alpha^{opt})$ on τ. The x-axis is τ (from 1 day to 28 days), the left-hand y-axis is the average number of events detected with θ_α^{opt}, and the right-hand y-axis is the average detection rate. The red and green lines show respectively the average number of anomaly events and normal events detected in a day, and the yellow line shows the average performance. Each line has its standard error.

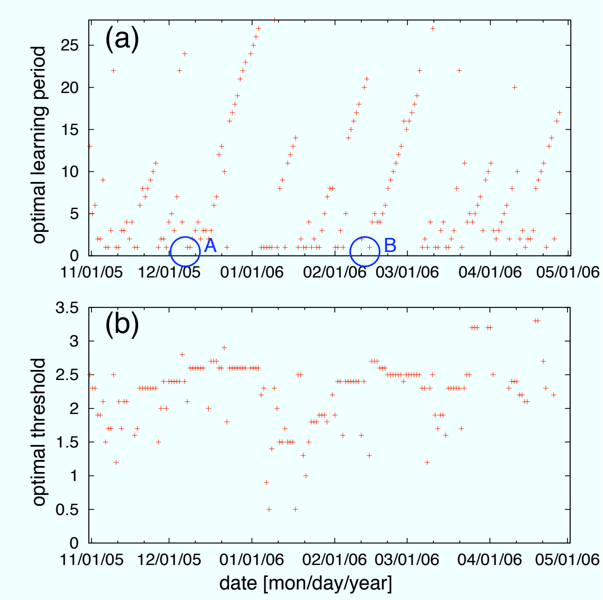

Fig. 5. Changes in θ_α^{opt} with variable learning period

- The red and green lines show that the number of detected events has a dependency on τ; a longer τ reduces the number of events detected in a day. This is because a longer τ results in high θ_α^{opt} (Fig.4).
- The yellow line shows that the average $a(\theta_\alpha^{opt})$ depends on τ; a shorter τ (1 or 2 days) leads to worse performance. The reason is that a shorter τ cannot yield enough

[1] port scans of Dasher.B virus, from source port 6000 to destination port 1025

[2] 2 scans from source port 6000 to destination port 8080 on 18th, and 1 scan from 1663 to 1434 on 19th

amounts of datasets for accurately determining θ_α^{opt}.

The figure clearly shows that a τ of 3 days is the most appropriate, because $a(\theta_\alpha^{opt})$ is larger and the number of detected anomaly events is larger. In addition, we have to compare the performance $a(\theta_\alpha^{opt})$ with the percentage of Attack events in the dataset: 8.9% (Fig.1). Any increase in $a(\theta_\alpha^{opt})$ with respect to the percentage will be a sign of the algorithm's efficacy.

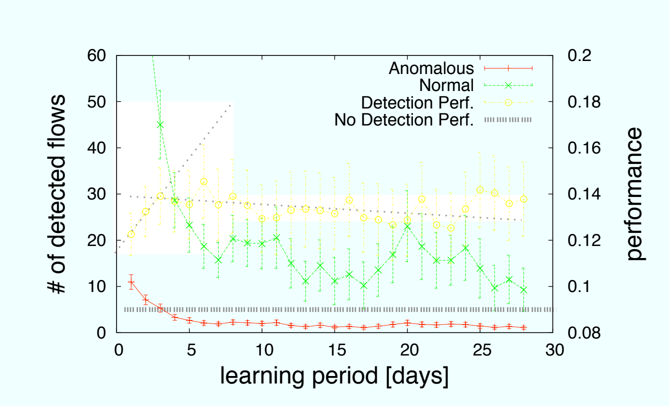

Fig. 6. Relation between length of learning period τ and the average performance $E[a(\theta_\alpha^{opt})]$(from Oct. 2005 to Apr. 2006).

D. *Performance evaluation*

Fig.7 illustrates the effect from introducing the parameter learning on performance. In Fig.7(a), the x-axis is date and the y-axis is the number of detected anomaly events. The y-axis in Fig.7(b) is the ratio of the number of detected anomaly events to that of all detected events in each day. In both figures, the gray lines are the results of the best (no learning) parameter, as mentioned in IV-A. The red lines are the results of θ_α^{opt} determined with a learning period of $\tau = 3days$.

Fig.7(b) shows that the learning generally degrades the performance, but the degradation is not critical to a practical detection. The parameter learning degrades the average performance by only 16.6%, and its standard error is 9.2%. Also, the learning reduces the number of detected anomaly events (Fig.7(a)). With the best parameter, the average number of detected anomaly events is 10.8 ± 1.3. On the other hand, with learning, it is 5.4 ± 0.9; thus, learning reduces the number of detected events by about 50%.

V. Concluding Remarks

A. *Discussion*

High variability of the optimal parameter: Our results illustrate the importance of a dynamic parameter tuning of statistical anomaly detection algorithms for deploying them in the real Internet. An inappropriate parameter significantly degrades performance, because the ratio of the number of anomaly events to that of total events is not constant relative to the value of the parameter. In addition, even if we set an optimal parameter, we cannot use the same value consistently, because the optimal parameter is not constant (Fig.4). This feature is not specific to our method; since Internet traffic has a tradeoff, other methods also likely require parameter tuning at the daily level in order to follow the macroscopic behavior of network anomalies.

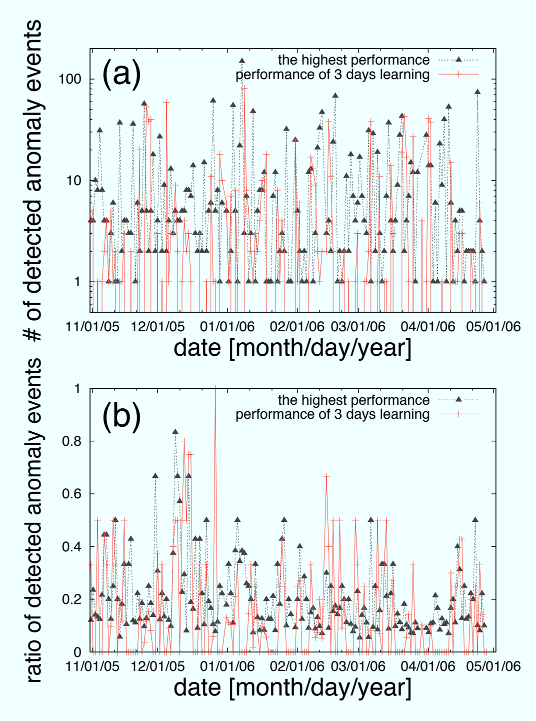

Fig. 7. Performance degradation caused by introducing parameter learning: (a) the number of detected anomaly events, (b) the ratio of detected anomaly events.

Advantage of using the multi-scale gamma-function model: Fig.4(a) shows that θ_α^{opt} is quite scattered; here, we point out only that the macroscopic behavior of the anomaly traffic exhibits no typical pattern. On the other hand, since α determines the shape of the histogram of the number of packets at a certain timescale, the multi-scale gamma-function model can follow the fluctuating traffic patterns. Consequently, the anomaly detection algorithm based on the multi-scale gamma-function model is a promising approach.

Validity of our classification method: Of course, our classification method has a room for improvement, because it classifies not a small number of events into the Unknown category (Fig.1). To make further improvements, we will use the idea of BLINC [13]. In particular, we will introduce BLINC's traffic classification method that builds a structure of connection patterns and applies them to heuristics. BLINC's idea and our classification method have a similarity in that both can classify at the host level (what we called "events" in this paper) rather than at the flow or packet level; hence the two methods should work well together.

B. *Summary and conclusion*

We discussed automatic and dynamic parameter tuning of a statistical Internet traffic anomaly detection algorithm, by using real traffic traces gathered over the period of six months.

The main results are as follows. (1) Without learning, the best parameter varies day by day. (2) With a longer learning period, the best parameter setting is affected by significant events during the learning period. (3) The appropriate period of learning is about 3 days. (4) The performance degradation caused by introducing dynamic parameter tuning is 17% in the best case. We showed the importance of dynamic parameter tuning in statistical network traffic anomaly detection methods for the real-world deployment. To confirm the efficacy of our method, we will conduct further research with other datasets (e.g., CAIDA's DITL dataset [14]) for analyses at the daily level and investigation on other links.

Acknowledgements

We appreciate to Patrice Abry, Pierre Borgnat, Guillaume Dewaele and Kaoru Yoshida for their advice. This work was partially supported by the Strategic International Cooperative Program of CNRS and JST, and by the Grant-in-Aid for Scientific Research on priority areas (info-plosion) from MEXT Japan.

References

[1] M. Roesch, "Snort - Lightweight Intrusion Detection for Networks," *LISA'99: Proceedings of the 13th USENIX conference on System administration*, pp. 229–238, November 1999.

[2] G. Dewaele, K. Fukuda, P. Borgnat, P. Abry, and K. Cho, "Extracting Hidden Anomalies using Sketch and Non Gaussian Multiresolution Statistical Detection Procedure," *ACM SIGCOMM LSAD'07*, pp. 145–152, August 2007.

[3] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," *ACM IMW'02*, pp. 71–82, November 2002.

[4] J. Brutlag, "Aberrant Behavior Detection in Time Series for Network Monitoring," *USENIX LISA 2000*, pp. 139–146, December 2000.

[5] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing Network-Wide Traffic Anomalies," *ACM SIGCOMM'04*, pp. 219–230, August 2004.

[6] S. Jin and D. S. Yeung, "A Covariance Analysis Model for DDoS Attack Detection," *IEEE Communications Society Volume 4*, pp. 1882–1886, August 2003.

[7] Y. Kim, W. C. Lau, M. C. Chuah, and H. J. Chao, "PacketScore: A Statistics-Based Packet Filtering Scheme against Distributed Denial-of-Service," *IEEE Transaction on Dependable and Secure Computing Volume 3 No. 2*, pp. 141–155, April-June 2006.

[8] B. Krishnamurty, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based Change Detection: Methods Evaluation and Applications," *ACM IMC'03*, pp. 234–247, October 2003.

[9] M. P. Stoeklin, J. Y. L. Boudec, and A. Kind, "A Two-Layered Anomaly Detection Technique Based on Multi-modal Flow Behavior Models," *PAM2008*, pp. 212–221, April 2008.

[10] Y. Gu, A. McCallum, and D. Towsley, "Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation," *IMC2005*, pp. 345–350, October 2005.

[11] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of PCA for Traffic Anomaly Detection," *ACM SIGMETRICS 2007*, pp. 109–120, June 2007.

[12] K. Cho, K. Mitsuya, and A. Kato, "Traffic Data Reposity at the WIDE Project," *USENIX 2000 FREENIX Track*, June 2000.

[13] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel Traffic Classification in the Dark," *ACM SIGCOMM'05*, pp. 229–240, August 2005.

[14] kc claffy, "A Day in the Life of the Internet: Proposed community-wide experiment," *ACM SIGCOMM CCR Volume 36, No. 2*, pp. 39–40, October 2006.