A Method of Detecting Network Anomalies in Cyclic Traffic

Shigeaki Harada, Ryoichi Kawahara, Tatsuya Mori, Noriaki Kamiyama, Haruhisa Hasegawa and Hideaki Yoshino
NTT Service Integration Laboratories, NTT Corporation, 3-9-11 Midori-cho, Musashino-shi, Tokyo 180-8585, Japan

Abstract—We present a method of detecting network anomalies, such as DDoS (distributed denial of service) attacks and flash crowds, automatically in real time. We evaluated this method using measured traffic data and found that it successfully differentiated suspicious traffic. In this paper, we focus on cyclic traffic, which has a daily and/or weekly cycle, and show that the differentiation accuracy is improved by utilizing such a cyclic tendency in anomaly detection. Our method differentiates suspicious traffic that has different statistical characteristics from normal traffic. At the same time, it learns about cyclic large-volume traffic, such as traffic for network operations, and finally considers it to be legitimate.

I. INTRODUCTION

Anomaly detection in real time is a trigger for diagnosing and controlling traffic, so it is an important technique that influences the reliability of network operations. However, because the number of points/items to be monitored becomes enormous as the number of routers increases, operations in a large network become harder. In addition, to improve the detectability of anomalies, we should spatially partition the monitored traffic into some groups, for example by each source or destination IP address (src IP or dst IP), and multilaterally monitor each group [6], [10], [14]. However, this technique has the same problem as mentioned above. The number of items to be monitored becomes enormous. Therefore, it is necessary to automate anomaly detection in order to handle the enormous number of items to be monitored.

The determination of the end of anomalies is a trigger for aborting the diagnosis or deactivating the traffic control. When we face an enormous number of monitoring items, two or more anomalous traffic flows may be detected in a short period of time. Among such detected anomalies, we might want to focus on the diagnosis of continuing anomalies rather than anomalies that have already ended because continuing anomalies might be causing severe damage to users. Therefore, we need to determine whether a detected anomaly is continuing or not. Furthermore, we must determine the end of an anomaly in order to deactivate the traffic control because the traffic control, such as filtering, may influence normal traffic as well as anomalous traffic. As a result, innocent users might suffer unreasonable loss as a result of traffic control filtering anomalous traffic. Therefore, if we can determine the end of the anomalous traffic, we will be able to deactivate the network control as soon as possible after the anomaly has ended.

For these reasons, we need a technique that can both detect anomalies and determine the end of the anomaly with high accuracy. Moreover, considering the enormous number of items, both these processes must be automatic.

In this paper, we propose an automatic anomaly detection and anomaly termination detection algorithm. The key idea of our algorithm is to leverage cyclic characteristics such as the diurnal pattern of observed traffic in the prediction. It is well known that backbone traffic exhibits visible strong cyclic characteristics [17]. We focus on the high correlation between daily traffic patterns and parameterize statistics obtained by observation so as to model the cyclic traffic. Then we predict the long-term trend of observed traffic using a Kalman filter and the Hoeffding-Azuma inequality [3] based on the theory of stochastic processes, which rely on the cyclic traffic model.

This paper is organized as follows: Section II reviews previous studies and compares them with ours. Section III describes the traffic model used in this work. Section IV proposes our algorithm. Section V presents evaluation results using traffic data measured on an actual Internet backbone line. Section VI summarizes our work and mentions future work.

II. RELATED WORK

Anomaly detection has recently been studied intensely from various perspectives. Various statistical techniques such as signal processing [2], [22], time-series analysis [4], [9], [20], [21], multivariate analysis [1], [11], [12], and machine learning [1], [7], [15] have been applied to the detection of network anomalies. Zhang et al. [23] proposed a generic framework for detecting anomalous traffic in both the spatial and temporal domains. It has also been revealed that using some form of digest of network information such as entropy [5], [12], [13] and sketch [9] plays a vital role in establishing accurate and scalable anomaly detection schemes. The advantage of using such a digest lies in the great saving in the memory space needed to track the massive amount of measured information without losing important information such as spatial diversity. Furthermore, recent work has focused on how the packet sampling process, which is a current off-the-shelf scalable measurement technique widely used in commercial ISPs, affects network anomaly detection schemes [16]. To establish a scalable network anomaly detection system, Huang et al. [8] proposed a promising data filtering scheme that drastically reduces the overhead of communications among network measurement equipment while achieving accurate anomaly detection.

Among the large number of papers on anomaly detection, several have addressed the online detection of anomalous traffic, which is what we focus on in this paper. Brutlag et al. [4] used the Holt-Winters forecasting model as the basis of their method. The key idea of their approach is to find a violation of the prediction in the time window. Krishnamurthy et al. [9] used several time-series forecasting/smoothing techniques such as variants of the ARIMA (auto-regressive integrated moving average) model and the Holt-Winters model to identify...
anomalous hosts or flows that greatly contribute to a significant change in the overall traffic pattern. Soule et al. [19], [20] showed an approach to monitoring origin-destination flows in a large network based on the Kalman filter. They estimated and predicted a traffic matrix, which is a mapping function from the observed traffic to the origin-destination flows, and detected anomalies by comparing the predicted traffic matrix with the actual traffic matrix. Ahmed et al. [1] proposed an online anomaly detection method for multivariate data based on an extension of the online learning algorithm.

This paper goes beyond those approaches to develop a new framework for an online anomaly detection scheme that can differentiate the normal and anomalous phases and hence detect not only the beginning but also the end of an anomalous phase. In this paper, we evaluate our algorithm with the univariate model, but our algorithm is also applicable to the multivariate model for detecting network-wide anomalies.

III. FORMALIZATION OF THE PROBLEM

Here, we first define the observed time-series data used in this paper. We denote the value observed at discrete time period $t$ as $y_t$. This $y_t$ indicates a traffic volume, such as the number of packets or the number of flows, counted during the measurement period $[t - \Delta, t]$, where $\Delta$ is the length of a fixed measurement period. From now on, we simply call the time-series observations $y_1, y_2, \ldots$ the traffic. The traffic can be separated into two components: normal traffic, which is steady, and anomalous traffic, which represents fluctuations in the normal traffic.

In this paper, we focus on the traffic that has cyclic changes in volume, as shown in Fig. 1: the normalized number of bytes measured over a measurement period of 5 min at JPNAP [24], which is one of the largest commercial Internet exchanges in Japan. Such traffic can be observed at multiplexing points like Internet backbones. In Fig. 1, we show a certain day of the week during nine weeks of traffic. We can see the diurnal pattern and three outliers in the 2nd, 5th, and 7th weeks. In this paper, we consider such outliers as traffic that is suspected of being caused by an anomaly, and we detect them using a threshold calculated from observation statistics.

A. Modeling of normal traffic

In this work, we consider normal traffic as a mixture of the trend $x_1, x_2, \ldots$, which is the long-term variation, and fluctuations $v_1, v_2, \ldots$, which are short-term variations. That is, the observations $y_1, y_2, \ldots$ are given by the sum of the trend values $x_1, x_2, \ldots$ and the fluctuation values $v_1, v_2, \ldots$ at each time as $y_t = x_t + v_t$. We consider the trend value $x_t$ as the state and the fluctuation value $v_t$ as the observation error. Note that we cannot observe the state value $x_t$ and observation error $v_t$ separately, so we need to estimate the state $x_t$ from the observation $y_t$. Furthermore, we assume that $v_t$ obeys a normal distribution with zero mean and variance $\text{Var}[v_t] = \sigma^2$. Furthermore, the process noise is assumed to be independent of the observation error.

In this paper, the traffic model is described using the observation and the state as

$$x_{t+1} = c_i x_t + w_t,$$  
$$y_t = x_t + v_t.$$  

From now, we simply call $\theta_t = (c_t, r_t, q_t)$ of normal traffic the parameter at time $t$.

B. Modeling of cyclic patterns

When the observations $y_1, y_2, \ldots$ have a cycle of $T$, the traffic model can be specified by $T$ parameters $\theta_1, \ldots, \theta_T$. In order to model the observation precisely, we should estimate $T$ parameters. For example, when the measurement period is every 5 min and the cycle $T$ is 1 week, we need to maintain 2016 parameters. In this paper, however, we segmentize a cycle into $M(< T)$ timelaps, such as 1 or 2 hours, and we estimate each of the $M$ parameters, which we call slot parameters denoted by $\hat{\theta}^j_i$ from observations made in corresponding timeslots (Fig. 2). The reason we consider segmented timeslots is to estimate parameters correctly by using many samples. However, to avoid averaging the time-variation of the observations, we should set the timeslots to short intervals.

From now on, let $A^j_i$ denote the $j$th cycle and let $S^j_i$ denote the $i$th timeslot in the $j$th cycle. Then, the cycles and slots are described by a set of timestamps:

$$S^j_i = \{t^j_1, t^j_2, \ldots, t^j_M + T - 1\},$$
$$A^j_i = \bigcup_{i=1}^M S^j_i = \{t^j_1, t^j_2, \ldots, t^j_M + T - 1\}.$$  

Here, $t^j_i$ denotes the starting time of the $i$th timeslot in the $j$th cycle. We let the size of each timeslot be $|S^j_i| = z_t, (i = 1, 2, \ldots)$.  

IV. PROPOSED ALGORITHM

Our algorithm has two functions: sequentially processed anomaly detection and periodically processed slot parameter estimation (Fig. 3).

In the anomaly detection process, our algorithm calculates threshold values so as to detect outliers. The threshold values

---

**Fig. 1.** Periodic traffic.
Initial learning: estimate each $\hat{\theta}_i^j$ using the slot parameters which were estimated in the previous inequality to estimate unobservable behaviors of normal traffic because it resulted from anomalous parameters. Therefore, observation for the Kalman filter or for threshold updating is determined to be anomalous, we do not use the current based on the prediction. On the other hand, if the observation observation, which will be used for the next determination, Kalman filter to make the next prediction from the current

A. Calculating the baseline

are based on both the expectation value of normal traffic (baseline) and the acceptable range. In the slot parameter estimation process, our algorithm uses the EM (expectation maximizing) algorithm to estimate each slot parameter [18].

1) Calculating baseline $\hat{y}_{t+1}$.

2) Estimation step: When a new observation $y_t$ arrives, the Kalman filter updates the state estimation $x_{t+1}$ and its variance $p_{t+1}$. The new state and variance estimations are calculated as follows:

$$x_{t+1} = x_{t+1} + k_t(y_t - x_{t+1}),$$

(5)

$$p_{t+1} = p_{t+1} - k_t p_{t+1}.$$  

(6)

Here, $k_{t+1}$, which is called the Kalman gain, is given by

$$k_t = \frac{p_{t+1}}{d_t^2 + p_{t+1}}.$$ 

(7)

b) Prediction step: Given the state estimation $x_{t+1}$ and the variance $p_{t+1}$ at time $t$, the state prediction $x_{t+1}$ and the variance prediction $p_{t+1}$ are calculated as follows:

$$x_{t+1} = c_t' x_{t+1},$$

(8)

$$p_{t+1} = p_{t+1} + q_t.$$ 

(9)

The obtained state prediction $x_{t+1}$ is the baseline based on the normal traffic model in timeslot $S_t$. That is,

$$\hat{y}_{t+1} = E[x_{t+1} + v_{t+1}] = c_t' x_{t+1}. $$

(10)

Note that $E[v_{t+1}] = 0.$

2) n-step prediction: After an observation has been determined to be anomalous by the Kalman-filter-based anomaly detection, our algorithm uses HA prediction, which is based on the Hoeffding-Azuma inequality, to calculate a baseline. Let $l$ denote the last time an observation was determined to be normal, and let $n$ denote the elapsed time since time $l$.

First, we briefly review the Hoeffding-Azuma inequality. Let $X_1, X_2, \ldots, X_n$ be a random sequence that satisfies $E[X_i | X_1, \ldots, X_{i-1}] = 0$ for $k = 1, 2, \ldots, n$. The Hoeffding-Azuma inequality gives the upper bound of the deviation between the sum of random variables $\sum_{k=1}^n X_k$ and the expectation $E[\sum_{k=1}^n X_k] = 0$.

**Theorem 1.** (Hoeffding-Azuma) Let $a_1, \ldots, a_n$ be constants and assume that $X_1, \ldots, X_n$ is a martingale sequence, which satisfies $|X_k| \leq a_k$. Then, for any $n, \epsilon > 0$,

$$\Pr[|\sum_{k=1}^n X_k| \geq \epsilon] \leq 2\exp\left(-\frac{\epsilon^2}{2\sum_{k=1}^n a_k^2}\right).$$ 

(11)

Here, a random sequence $X_1, \ldots, X_n$ that satisfies $E[X_k | X_1, \ldots, X_{k-1}] = 0$ for $k = 1, 2, \ldots, n$ is called a martingale sequence.

Replacing the right-hand side of (11) by $\delta$, with probability of at least $1 - \delta$, we get

$$\sum_{k=1}^n X_k \leq \sqrt{2(\sum_{k=1}^n a_k^2) \ln(2/\delta)}. $$

(12)

We call this the “Hoeffding-Azuma inequality” in this paper.

Recall that the trend coefficient, the variances of the observation error and process noise are constant in the same timeslot; that is, $c_t = c_t', r_t = r_t', q_t = q_t'$, $\forall t \in S_t'$. The process
noise \( w_t \) obeys the normal distribution with zero mean and variance \( q_t^2 \). Thus, the conditional expectation of \( x_t \) given a sequence \( x_1, \ldots, x_{t-1} \) is

\[
E[x_{t+1}|x_1, \ldots, x_t] = c_t^2 x_t.
\]

Therefore, the sequence of the process noise \( w_1, \ldots, w_t \) is a martingale sequence because \( w_t = x_{t+1} - c_t x_t \).

On the other hand, when the observation at time \( t \) had been determined to be normal, the n-step prediction from time \( t \) with respect to the state is given by \( x_{t+n} = (c_t^2)^n x_t + \sum_{k=1}^{t-n} (c_t^2)^{k-1} w_{t+k-1} \). Hence, if we can set \( \delta \) and the coefficient sequence \( a_1, \ldots, a_n \) such that \( |(c_t^2)^{k-1} w_{t+k-1}| \leq a_t \), we can estimate the upper bound of \( \sum_{k=1}^{t-n} (c_t^2)^{k-1} w_{t+k-1} \) by using the Hoeffding-Azuma inequality.

As a result, the n-step prediction from the last time at which the observation was determined to be normal is given by

\[
x_{t+n} = (c_t^2)^n x_t/2 \pm \sqrt{2 \sum_{k=1}^{t-n} a_k^2} \ln(2/\delta).
\]

Here, we can determine the coefficients \( a_1, \ldots, a_n \), but the details are omitted.

Finally, letting \( \delta \in (0, 1) \) and \( \gamma > 1 \) be a fixed constant, the n-step prediction during the anomaly is given by

\[
\hat{y}_{t+n} = E[x_{t+n} + v_{t+n}] = (c_t^2)^n x_t/2 + \gamma \sigma_t^2 \sqrt{n \ln(2/\delta)}.
\]

Note that \( E[v_{t+n}] = 0 \).

B. Calculating the acceptable range

Our algorithm defines the acceptable range based on a standard deviation of observations. The standard deviation can be derived from each pair of variances \( (\bar{r}, \tilde{q}) \), which were estimated for the slot parameter \( \hat{\theta} \). In this paper, the algorithm calculates the weighted average of the two estimated parameters so as to consider a transition in a timeslot, as shown in Fig. 4.

The acceptable range is calculated as follows. Recall that \( t^j_i \) denotes the start time of timeslot \( S^j_i \), and \( z_l \) denotes the length of the timeslot. Let \( \bar{t}^j_i = t^j_i + (z_l - 1)/2 \) denote the central time of timeslot \( S^j_i \). Then, when \( t_i^j \leq t < \bar{t}^j_i \), the estimated parameter at time \( t \) is calculated by

\[
\hat{\theta}_t = \frac{t - \bar{t}^j_i}{t^j_i - \bar{t}^j_i} \hat{\theta}^j_i + \frac{\bar{t}^j_i - t}{t^j_i - \bar{t}^j_i} \hat{\theta}_{t^j_i}.
\]

Also, when \( \bar{t}^j_i \leq t < t^j_{i+1} \), the estimated parameter at time \( t \) is calculated in the same manner.

Next, we calculate the standard deviation from the slot parameters obtained above. Recall that, while the HA prediction considers process noise, the Kalman filter does not. The standard deviations of the Kalman filter prediction and the HA prediction are given by \( \bar{\sigma}_{t+1} = \sqrt{t_{l+1} + q_{t+1}} \) and \( \sigma_{t+1} = \sqrt{t_{r+1}} \), respectively.

C. Examining an observation using the threshold

Our algorithm detects an outlier as a suspicious observation by using a threshold \( \hat{y}_{t+1} \), which is calculated from the baseline \( \hat{x}_{t+1} \) and the acceptable range \( \bar{\sigma}_{t+1} \). That is, our algorithm sets thresholds \( \hat{y}_{t+1} = \hat{x}_{t+1} + \alpha \sigma_{t+1} \) and \( \hat{y}_{t+1} = \hat{x}_{t+1} - \alpha \sigma_{t+1} \) and generates an alarm if \( y_{t+1} > \hat{y}_{t+1} \) or \( y_{t+1} < \hat{y}_{t+1} \). Here, \( \alpha \) is a parameter that controls the sensitivity of detection.

For both the Kalman filter and HA prediction, the amount of calculation required to obtain a threshold is constant. Furthermore, because the threshold value at time \( t \) is calculated at time \( t-1 \), our algorithm can check observations in real time.

D. Updating slot parameters

At the end of every timeslot, our algorithm estimates the parameters of the corresponding observations using the EM algorithm [18]. After the last observation has been received in the timeslot, the slot parameter will be updated in the following manner. First, using the observations \( y_k \), the EM algorithm estimates the temporal parameter \( \hat{\theta}(S^j_i) \). Next, our algorithm updates the slot parameter to the weighted average of the temporal parameter \( \hat{\theta}(S^j_i) \) and the old slot parameter \( \hat{\theta}^j_i \). That is, if we let \( \eta \in (0, 1) \) be a certain coefficient, it sets

\[
\hat{\theta}^{j+1}_i = (1 - \eta) \hat{\theta}^j_i + \eta \hat{\theta}(S^j_i).
\]

There are two reasons for choosing to use the weighted average in parameter updating. One is to curb the influence of one-cycle-only transitions. The other is to reflect the following transition in the parameter estimation.

In addition, while our algorithm usually uses the actual observations made in the corresponding timeslots when it estimates the temporal parameters, in order to reduce the effect of anomalous observations on the estimation, it replaces a suspicious observation by the threshold value used to distinguish the outlier. Our algorithm calculates baselines and acceptable ranges based on the updated slot parameter \( \hat{\theta}^{j+1}_i \) in timeslot \( S^j_{i+1} \) of the next cycle.

V. Evaluation

The traffic data sequences \( \{y_1, y_2, \ldots\} \) that we used in this evaluation were the same data as in section III. Recall that the suspicious traffic was three spikes in the 2nd, 5th, and 7th weeks.

The results of running our algorithm are shown in Fig. 5, where the cycle lengths were 1 week and the timeslot lengths were all 2 hours. Note that the y-axes of these graphs have been normalized by the same coefficient. In this case, we had 84 slot parameters and parameter estimation was done every 2 hours so as to update one of them. In addition, the algorithm needs some learning time in the first cycle in order to estimate every slot parameter. Here, the parameters to be input initially were set as \( \alpha = 6 \), \( \delta = 0.01 \), \( \gamma = 1.5 \), and \( \eta = 0.5 \).

In this figure, we separated alerts into two levels: the high-level alert indicates that the observation exceeded the upper
The half-finished spike to be normal. We think that this can be avoided by considering two or more observations before the threshold and the low-level alert indicates the opposite. The results show that our algorithm could detect the three instances of suspicious traffic; however, in two of them, it could not determine the end of the spike and misjudged them to be lower outliers. The reason for this is that our algorithm misjudged the half-finished spike to be normal. We think that this can be avoided by considering two or more observations before the algorithm makes a decision.

In addition, we found that our algorithm also detected two conspicuous spikes, whose volumes were not so large, in the 3rd and 8th weeks. Because it can judge that durations like these are short, it can avoid excess alerts in cases like these.

VI. CONCLUSION

In this paper we have proposed a algorithm to distinguish anomalous (suspicous) changes from normal (legitimate) traffic in real time. We evaluated it using real traffic data and found that it could detect anomalous changes, while ignoring normal changes. Moreover, it enables us to find the end of anomalies accurately. If a network operator do not know the end of anomalies, she would be forced to keep track of all the observed anomalies manually; which is obviously inefficient way of managing her networks.

In our future work, we will seek to automate the way of adopting the appropriate lengths of the cycle and the time slot. While our detection scheme works quite well, we envision it could have a room to be further improved by adjusting the way we learn data that contains anomalous traffic.

ACKNOWLEDGMENTS

We are grateful to Yukiyasu Tarui of JPNAP for providing us with the traffic data.

REFERENCES