DP Matching Approach for Streaming Contents Detection Using Traffic Pattern

Kazumasa Matsuda[1], Hidehisa Nakayama[2], and Nei Kato[1]

[1] Graduate School of Information Science, Tohoku University
6-3-09 Aza-aoba, Aramaki, Aoba, Sendai, Miyagi, 980-8579, Japan
matsu@it.ecei.tohoku.ac.jp
[2] Department of Electronics and Intelligent Systems, Tohoku Institute of Technology
35-1 Yagiyama-Kasumicho, Taihaku, Sendai, Miyagi, 982-8577, Japan

Abstract. Since the video streaming technology has become widespread, the security of video contents has gained more importance. Our research group previously envisioned a video leakage detection method for streaming contents that makes use of traffic patterns extracted from only traffic volume information obtained from routers. However, as the background traffic increases, packet delay and jitter increase. As a consequence, the detection accuracy decreases. In this paper, we propose a new robust method, which is more resilient to increasing packet delay and jitter by improving the generation process of traffic patterns. In addition, we solve another problem induced by packet loss by using Dynamic Programming (DP) matching. Finally, we evaluate the influence of the background traffic on streaming video detection and discuss the results.

Keywords: content delivery, streaming, traffic pattern, video detection.

1 Introduction

With the rapid advance in the network speed and capacity, streaming technology has become popular recently. Video Delivery Services (e.g., Youtube [1]) help users watch the real-time video streaming more easily. Also, because web conference systems are available arranging, it becomes possible for the company to make their own choices without face-to-face meetings [2], [3]. Therefore, it is employed by many enterprises. On the other hand, we have to take measures against leaking the video streaming traffic because it contains classified information of the company. Generally, these technologies such as protection against confidentiality and author's copyright are called DRM (Digital Rights Management) technology, which can control usage of contents or illegal copying.

Encryption of contents with keys is a basic technique of DRM and it prevents abusive copies and usage of non-licensed users by means of distribution of the decryption key to only licensed users [4]. However, it is possible that the decrypted content is re-distributed to the third person. In recent years, streaming distribution via P2P (Peer to Peer) networks has, indeed, become widespread [5]. Hence, streaming traffic of enterprises' web conferences may be leaked to P2P networks.

The generally way of preventing leakage of streaming traffic to exterior networks is packet filtering on the firewall or the router. In packet filtering, we assign destination IP addresses, port numbers or protocol types such as TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) to block the traffic according to the assigned information [6]. However, in case of blocking the streaming traffic by means of conventional packet filtering, it is hard to filter the traffic because the destination address and port number are dynamic. Also, if there are public and secret contents in the streaming server, packet filtering cannot distinguish these contents.

The application filter, which analyzes the packet payloads, is required in order to block any streaming content. However, as traffic volume increases, the load of analysis also increases. For this reason, our research group previously proposed a video content identification method [7]. This method uses the captured traffic volume at the router nearby the distribution server and at the edge router connecting LAN (Local Area Network) and exterior network. Then the matching between these two traffic patterns lets us determine which video is served by the traffic currently on the edge router. This method can filter with relatively low load because of only traffic volume analysis (i.e., without analyzing packet payloads).

The matching requires traffic patterns based on these traffic volumes. However, the generation methods of traffic patterns in conventional schemes employ arrival time of packets. Hence, if packet jitter increases, video detection accuracy decreases due to the mismatching between these two traffic patterns. In this paper, we propose a new generation method of traffic patterns, which is robust against packet delay jitter by means of using packet size information instead of its arriving time. On the other hand, the expansion and contraction of traffic patterns frequently occurs in high packet loss environments. To solve this problem, we employ Dynamic Programming (DP) matching and enable video detection with high accuracy even in a high packet loss environment.

The rest of this paper is organized as follows. In Section 2, we introduce the overview of the video detection system using traffic volume and its usage. Section 3 presents the conventional method of traffic pattern generation, and discusses its inherent problem. In Section 4, we introduce details on the proposed method, which changes the generation method of traffic patterns and uses the DP matching. The performance evaluation of the envisioned method is presented in Section 5. Finally, Section 6 concludes the paper.

2 Video Detection System Using Traffic Volume

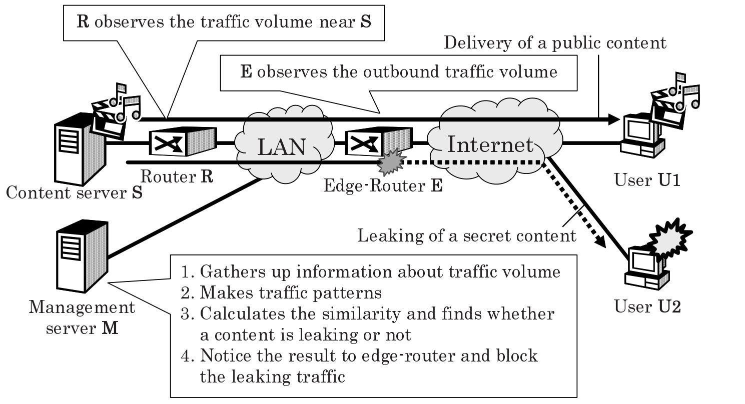
2.1 Overview

The bit rates of the Variable Bit Rate (VBR) streaming contents change according to the changes in the contents. When the content is delivered to the user through streaming, the traffic volume observed at the router close to the distribution server and at the edge router represents a unique pattern for each content just as a human fingerprint. By comparing this pattern on the delivery server-side with that on the edge router-side, it is, indeed, possible to detect whether the targeted content in the distribution server is leaking at the edge router or not. Many video session passes through the edge router, hence the edge router side pattern is generated from observing its traffic volume for a short time in contract with the distribution server. In addition, this method can be implemented with a lower computational cost because the traffic patterns matching operations are performed on the management server.

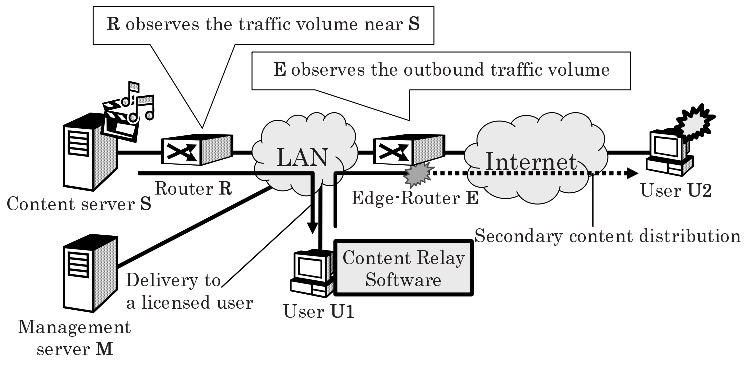
2.2 Overview of Video Detection System

In this section, we introduce two example of video leaking, and present an overview of the proposed system and how to apply the same.

The case of existing public and secret contents. In Fig. 1(a), distribution server S has both a content released to the exterior network and a confidential one, and user $U1$ is receiving a public streaming content. Then, we discuss the


R observes the traffic volume near S
Delivery of a public content
E observes the outbound traffic volume
LAN
Internet
Content server S
Router R
Edge-Router E
User U1
Leaking of a secret content
User U2
Management server M
1. Gathers up information about traffic volume
2. Makes traffic patterns
3. Calculates the similarity and finds whether a content is leaking or not
4. Notice the result to edge-router and block the leaking traffic

(a) The case of existence of public and secret contents.


R observes the traffic volume near S
E observes the outbound traffic volume
LAN
Internet
Content server S
Router R
Edge-Router E
User U2
Secondary content distribution
Content Relay Software
Delivery to a licensed user
User U1
Management server M

(b) The case of leakage of contents by a licensed user, $U1$.

Fig. 1. The video leakage detection mechanism

case where a confidential content is being leaked to user $U2$ due to misconfiguration of either the distribution server S or the packet filtering mechanism. In the proposed system, router R and edge router E observe distribution server side and outbound traffic volumes from the LAN, respectively. Each observed traffic information is sent to the management server, M. Next, M generates traffic patterns based on traffic volumes and calculates the correlation coefficient. Using this similarity, the system determine whether a confidential content is being leaked or not, and the edge router E prevent a session from getting leaked.

The case of re-distribution by licensed user. Fig. 1(b) shows that distribution server S deliver the contents to licensed user $U1$. Then, if the software which re-distributes the received content to exterior networks such as P2P networks, is installed in user $U1$, $U1$ re-distributes to unlicensed user $U2$. Re-distribution of destination and its port may be changed by the software. Therefore, it is hard to block by packet filtering. In this example, by means of observing router R and edge router E, and by patterns matching, it is possible to block these re-distribution sessions.

2.3 Matching Processes of Traffic Patterns

The outline of comparison of traffic patterns are shown in Fig. 2. The comparison of traffic patterns consists of three steps. First, in step 1, we place a window ($U[slot]$), which snips off a partial patterns X_U from the distribution server side traffic pattern X_S, where U is the length of the edge router side traffic pattern. Next, in step 2, similarity between the partial pattern X_U and the edge router side pattern T_U is calculated. In step 3, the window is moved from left to right by one slot. These three steps are repeated until the window reaches the rightmost part of the the distribution server side pattern, and finally obtains a similarity value of $(S - U + 1)$. When the similarity value is large, it means that the edge router side pattern is similar to the partial pattern found on the distribution server side. In this case, the management server decides that a confidential content is leaking. To decide whether a significantly large value exists or not, this method requires a threshold specific to the considered network setting and streaming dynamic.

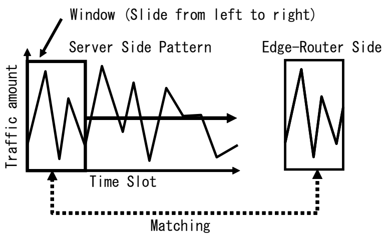


Fig. 2. An example of the comparison of traffic patterns

3 Definition of a Traffic Pattern in a Conventional Method

3.1 The Traffic Pattern Generation and Definition of Similarity

In method our previously proposed [7], the traffic pattern is defined as the volume of traffic for a certain time-slot, Δt[s], and expressed as a N-dimension vector.

$$X = (x_1, x_2, \cdots, x_N)^t, \quad T = N \cdot \Delta t \tag{1}$$

where T[s] is is the whole length of the traffic pattern and N[slot] is the number of time slots.

To detection a leaking, two patterns, X_S and Y_U are generated for the distribution side and the edge router side, respectively. Then X_S and Y_U are matched, i.e., compared. X_S and Y_U have S and Y-dimensions respectively, and $(S > U)$. The similarity between these two patterns are calculated as R_{XY} according to Eq. (2).

$$R_{XY} = \frac{X_U'^{\,t}\, Y_U'}{\sqrt{||X_U'||^2 \cdot ||Y_U'||^2}}, \quad -1 < R_{XY} < 1 \tag{2}$$

where X_U is the distribution server side pattern snipped off U-dimension, which is same as that of the edge router side patterns. In addition, X_U' and Y_U' are normalized patterns so that the mean is zero and the valiance is one, as shown in Eq. (3).

$$X_U' = \begin{pmatrix} (x_1 - \bar{x})/s_x \\ (x_2 - \bar{x})/s_x \\ \vdots \\ (x_U - \bar{x})/s_x \end{pmatrix}, Y_U' = \begin{pmatrix} (y_1 - \bar{y})/s_y \\ (y_2 - \bar{y})/s_y \\ \vdots \\ (y_U - \bar{y})/s_y \end{pmatrix} \tag{3}$$

where $\bar{x}$ and s_x denote the mean and standard deviation of the distribution server side pattern, respectively, while $\bar{y}$ and s_y indicate those of the edge router side pattern, respectively.

3.2 Problems Caused by Packet Delay Jitter

Generation method of traffic pattern of conventional method [7] is defined as the volume of traffic for a certain time-slot, as shown in Eq. (1). Then, we discuss the situation that some packets delays exceed Δt. These delayed packets are stored over the slot x_{i+1} instead of the primary slot x_i. Therefore, delay jitter of packet distorts the traffic pattern. As a consequence, the detection accuracy decreases.

4 A Novel Traffic Pattern Based on Packet Size

4.1 Enhancement of Traffic Pattern

Overview. To solve the aforementioned problem related to packet delay jitter, we propose, in this paper, a new generation method of traffic patterns, which is robust against packet delay jitter.

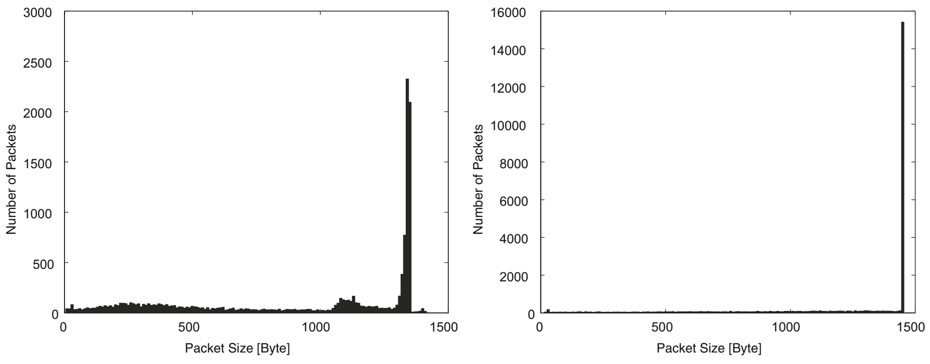
In the envisioned method, the splitting trigger of the traffic pattern receives packets with a certain packet size, instead of time-slot. As a result, the proposed method can produce a traffic pattern with robustness against packet delay jitter and achieves high detection accuracy regardless of a high packet delay jitter environment. This method can make the same traffic pattern between the distribution server and edge router sides unless packet re-ordering or packet loss occurs.

To decide the appropriate packet size, we need to take into account the distribution of packet size. We introduce overview of this survey and results, and decide a certain packet size to split traffic patterns in this section.

The distribution of packet sizes. The bit rates of Variable Bit Rate (VBR) streaming content change according to changes in the contents. Hence each packet size also changes momentarily. Then, if there is variability of packet size, we can split a traffic pattern while receiving a packet having a certain packet size. We connected the distribution server and the user with 1Gbps LAN and delivered six streaming contents, and captured each traffic volume at the router in order to search the distribution of packet sizes. Helix server [8] and Darwin Streaming Server (DSS) [9] were used as the distribution servers in this survey. The result of this survey is shown in Table. 1 and the example of the distribution of packet size is shown in Fig. 3. According to Table. 1, all video contents have variability and similar distributions. Therefore, the proposed method does not depend on content type. Also, according to Fig. 3, there are many packets of MSS (Maximum Segment Size) in both Helix Server and Darwin Streaming Server. If the packet size range of splitting traffic pattern is set around the MSS, the number of packets into each slot becomes extremely low. On the other hand, there are some packets of small size moderately. Packet size range is set to small

Table 1. Average of streaming packet size and variance

Video name	Length	Helix Server		Darwin Streaming Server	
		Ave. of packet size	Variance	Ave. of packet size	Variance
Anime	1m40s	953.5 Bytes	206915	1212.9 Bytes	154573
Documentary 1	1m38s	928.6 Bytes	246853	1253.9 Bytes	136184
Documentary 2	1m43s	940.8 Bytes	217488	1224.8 Bytes	142604
Variety show	2m0s	694.8 Bytes	222917	1268.1 Bytes	115492
Baseball	2m0s	719.4 Bytes	209743	1241.5 Bytes	136110
News	2m0s	710.9 Bytes	218307	1238.8 Bytes	118364



(a) Helix Server (b) Darwin Streaming Server

Fig. 3. Example of packet size distribution (Anime)

size in the proposed method so that the number of packets into each slot is also moderate.

4.2 Traffic Pattern Matching Using DP Matching

Discussion about matching method. The generation method of traffic patterns which is based on packet size is robust against packet delay jitter. However, if a packet which is separable a traffic pattern is dropped, the traffic pattern is not split by this packet. As a result, traffic patterns between the distribution server and the edge router sides become different.

Fig. 4 shows an example of each traffic pattern based on the same video content, with 1% loss between the distribution server and the edge router. According to Fig. 4, the gap between two traffic patterns was caused by packet loss.

When such a gap in traffic patterns occurs due to packet loss, we can alleviate it by means of expansion and contraction of the pattern on one side so that the matching will be successful. Such a matching technique is called DP (Dynamic Programming) Matching [10], and we adopted this method in our proposed scheme. We introduce how to apply this matching method to the proposed approach in the remainder of this section.

How to apply DP matching. To apply DP matching to the proposed approach, we need to define the distance $d(i,j)$ between a_i and b_j in the distribution server side pattern T and edge router side pattern R, as shown below.

$$\begin{cases} T = a_1 a_2 \cdots a_i \cdots a_I \\ R = b_1 b_2 \cdots b_j \cdots b_J \end{cases} \tag{4}$$

where the server side pattern is snipped off the window which is the same length as the edge router side pattern and the window is moved from left to right by one slot. Hence $(I = J = U)$, where U is the length of edge router side pattern.

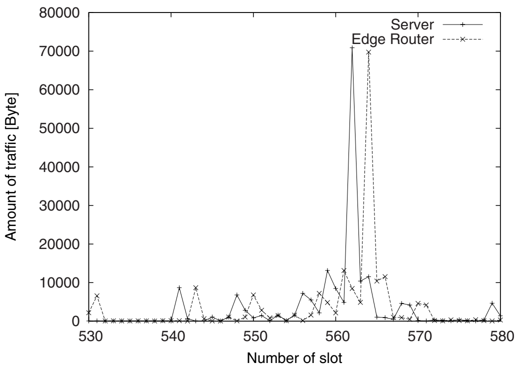


Fig. 4. Server side and edge-router side traffic pattern (loss: 1%)

We describe the cost of displacement to convert from T to R. Each distance of vector between a_i and b_j is defined as $d(i,j) = |a_i - b_j|$. and cost r per gap is added. Finally, T and R are normalized by the number of slots.

When the distance between two vector, $(a_1 a_2 \cdots a_i)$ and $(b_1 b_2 \cdots b_j)$ defined as $g(i,j)$, $D(T,R)$ is given by following recurrence equation because of principle of optimality of dynamic programming.

① Initial condition $g(0,0) = d(0,0)$
$g(i,0) = g(i-1,0) + r + d(i,0)$
for $i = 1, 2, \cdots, I$
$g(0,j) = g(0,j-1) + r + d(0,j)$
for $j = 1, 2, \cdots, J$
② Execute ④ about $i = 1, 2, \cdots, I$
③ Execute ④ about $j = 1, 2, \cdots, J$
④ $g(i,j) = \min \begin{cases} g(i-1,j) + r + d(i,j) \\ g(i-1,j-1) + d(i,j) \\ g(i,j-1) + r + d(i,j) \end{cases}$
⑤ $D(T,R) = g(I,J)/(I+J) = g(U,U)/2U$

The similarity between the server and the edge router sides is calculated as $D(T,R)$. Because the window is moved by one slot while calculating each similarity, conclusive similarity between two patterns is the minimum of each similarity value.

An example of similarity graph using DP matching. Fig. 5 shows the similarity graph using DP matching between two traffic patterns which are based on the same content with packet loss rate 1%. If two traffic patterns are based on the same content, we can find the minimum peak in the similarity graph. Because when the window of the server side pattern corresponds with that of the edge router, the costs of expansion, contraction, and displacement are low.

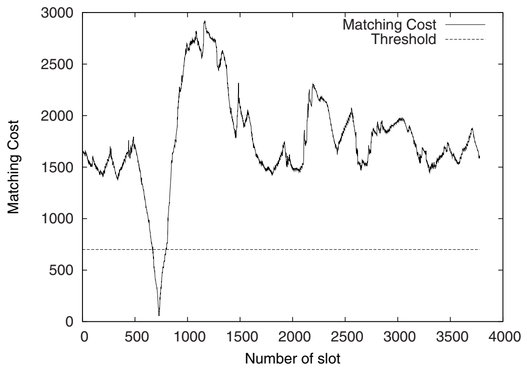


Fig. 5. Example of the DP matching similarity graph

The proposed method sets the threshold to detect this minimum value in order to detect whether the video content is being leaked at the edge router or not.

5 Performance Evaluation

5.1 Overview of the Experiment

To evaluate the feasibility of the proposed method, we conduct experiments about the influence of delay jitter, packet loss, and background traffic on the conventional [7] and the proposed methods. In the proposed method, we consider both the conventional matching (dubbed as P-TRAT) and the DP matching (DP-TRAT) techniques.

Six different contents, namely news, comedy show, anime, live baseball match, and so forth, are distributed from the server to the user, and the corresponding traffic volume of each contents is captured. Then, the traffic patterns are generated from the captured traffic volumes at the content server and also at the user. These traffic patterns are used to compute the similarity. Finally, we confirm the detection accuracy of the proposal approach. Table. 2 shows the experimental environment, and Fig. 6 depicts the topology of the considered network. In addition, NetEm [11] is used in order to produce packet jitter and packet loss. NetEm is a network simulator, which runs on Linux, and can occur jitter, packet loss, or packet re-ordering arbitrarily at a network interface. The background traffic is generated by nuttcp [12], which contains UDP flows from the server to the user.

The user traffic pattern is generated by capturing the whole content volume, and it is snipped off with 2500 packets. According to Fig. 2, observation of the user side traffic is shorter than that of the server side, the user side volume is observed at the whole time and snipped off it changing the snipping window position by 20 points. Finally, we calculate each similarity using the 20 user side

Table 2. Experimental environment

OS	Debian Linux 4.0iKernel v2.6.18j
Sever PC: CPU	Intel Pentium D 3.2 GHz
Client PC: CPU	Intel Core 2 Duo E6600 2.4 GHz
Bridge PC: CPU	Intel Xeon X5450 3.0 GHz
Network card	Intel PRO/1000 MT Server Adapter
Streaming server	Helix Server v11.1.6
Media player	RealPlayer v10.0.9
Video bit-rate	1 Mbps
Traffic observation library	libpcap v0.7.2
Number of packets in userside traffic pattern	2500
Conventional: time slot size	120 ms
P-TRAT, DP-TRAT: threshold of packet size	200 Bytes
Conventional: detection threshold	Chebyshev's inequality
P-TRAT: detection threshold	0.7
DP-TRAT: detection threshold	700

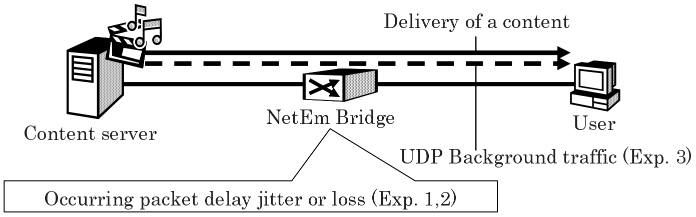

Delivery of a content
Content server
NetEm Bridge
User
UDP Background traffic (Exp. 3)
Occurring packet delay jitter or loss (Exp. 1,2)

Fig. 6. Consider network topology

traffic generated in the experiment. The average of these similarities is used as result.

The proposed method uses the precision and the recall as indices to verify the performance of the proposed method which are generally used in the validation of performance of the Information Retrieval systems. These indices represent the correctness and the completeness of the detection result, respectively. However, because these indices generally have a trade-off relationship, we also use F-measure, which is a comprehensive measurement to evaluate the performance of our approach. A large value of the F-measure indicates high performance, when it has a large value. These indices are expressed as follows.

$$\text{Precision} : Pr = \frac{C}{A} \times 100\,[\%] \tag{5}$$

$$\text{Recall} : Re = \frac{C}{W} \times 100\,[\%] \tag{6}$$

$$\text{F-measure} : F = \frac{2 \times Pr \times Re}{Pr + Re} \, [\%] \tag{7}$$

Here, C is the number of detections number of correct detection decisions (i.e., in terms of discovering an ongoing content-leakage). A is the number of detections, which is declared to be leaking the content, including the false positives. W is the number of actual content-leaks. These indices are calculated by combining all server side and user side patterns. We consider six server side traffic patterns, six user side patterns, and 20 window positions at the user side, thus a total combination of 720 different patterns.

5.2 Evaluation #1: Verification of the Influence of Packet Delay Jitter

Experimental setup. Six different contents are distributed from the server to the user by adding the packet delay jitter via NetEm bridge. The added delay jitter comprises of the following: delay value 200ms and jitter value ($\pm 0 \sim \pm 200$) ms. The jitter is gradually added every 20 ms.

Experimental results. Fig. 7 shows the results of the conducted experiment about precision, recall and F-measure, respectively. As shown in Fig. 7(a) and 7(b), the conventional method is affected by packet jitter and its recall is lower than the proposed method. On the other hand, the proposed method retains 100% F-measures regardless of the packet jitter.

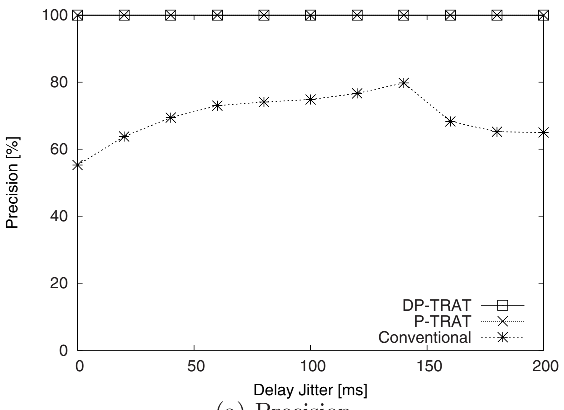
5.3 Evaluation #2: Verification of the Influence of Packet Loss

Experimental setup. In the same way as in the first experiment, six different contents are distributed and the packet loss of $0.1 \sim 10\%$ via NetEm bridge is injected.

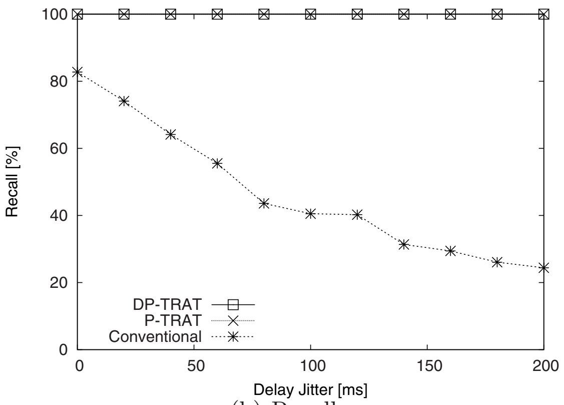
Experimental results. Fig. 8 shows the results of these indices. As shown in Fig. 8(a), the precision is retained by each considered method. However, as the packet loss rate increases, the recall of P-TRAT is low with a loss rate exceeding 0.5% (Fig. 8(b)). On the other hand, DP-TRAT achieves high indices regardless of the increasing packet loss rate.

5.4 Evaluation #3: Verification of the Influence of Background Traffic

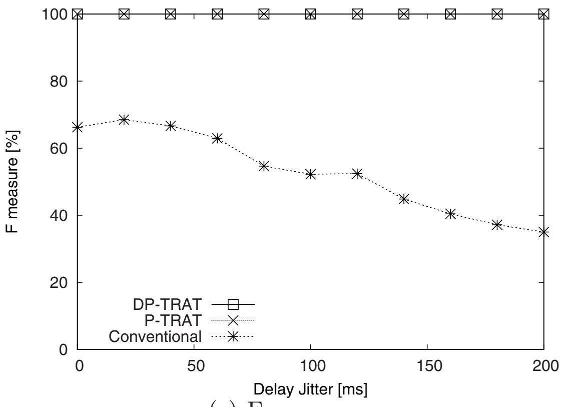
Experimental setup. In the same way as in the first and second experimental set-ups, six different contents are distributed. At the same time, UDP traffic flows ranging from $0 \sim 500$ Mbps are generated from the server to the user by nuttcp.



(a) Precision

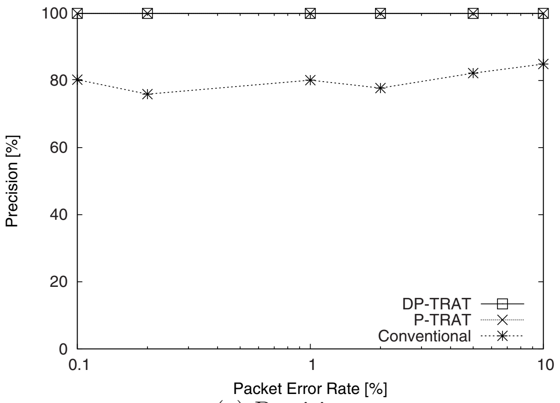


(b) Recall

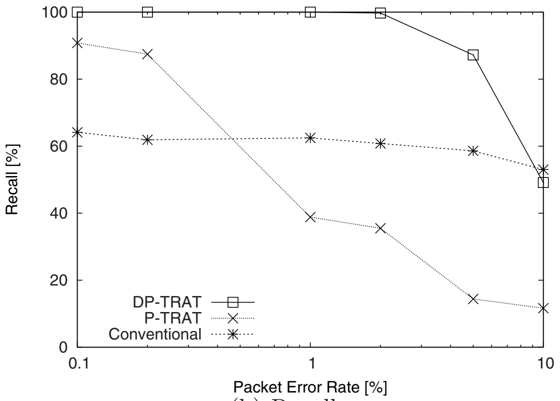


(c) F-measure

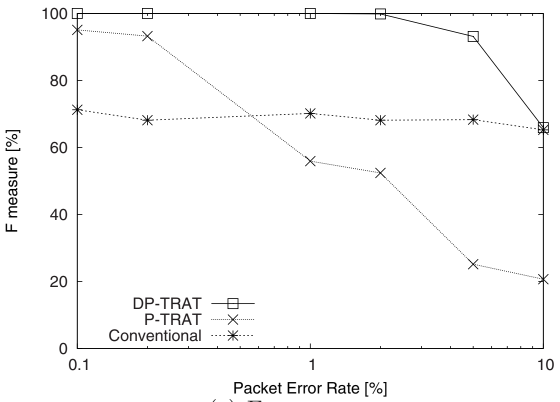
Fig. 7. Result of the evaluation #1 about jitter



(a) Precision

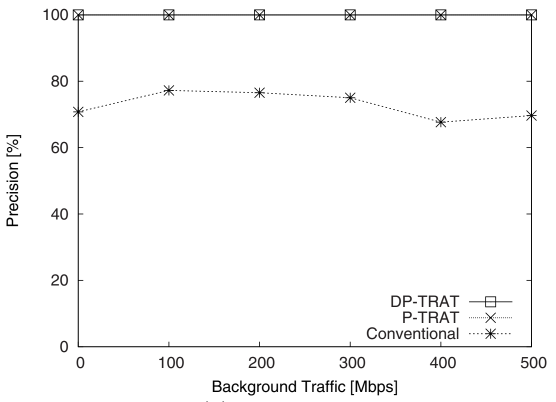


(b) Recall

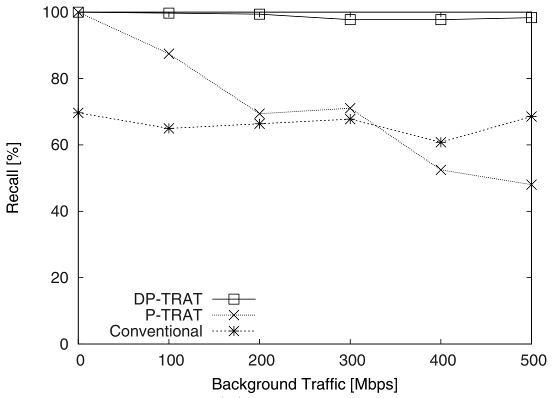


(c) F-measure

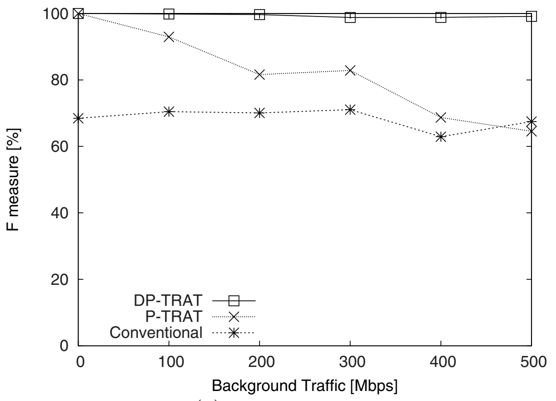
Fig. 8. Result of the evaluation #2 about packet loss



(a) Precision



(b) Recall



(c) F-measure

Fig. 9. Result of the evaluation #3 about background traffic

Experimental results. Fig. 9 shows the results of these indices. As shown in Fig. 9(a), each method is precise enough. However, as the background traffic increases, the recall and F-measures of P-TRAT drop. On the other hand, DP-TRAT retains these indices regardless of the increasing background traffic. In addition, the indices obtained through the proposed DP-TRAT are higher than those in the conventional method.

6 Conclusion

In this paper, we proposed a mechanism for effectively detecting the leaks in the streaming contents by observing and correlating the traffic at the distribution server and also at the edge router. This method can be used regardless of the destination IP addresses, port numbers and protocol types since it uses only packet size information. Therefore, this method can deny the session which is not blocked by packet filtering. As a result, this method can be employed as a firewall-aided technique.

We previously envisioned a video leakage detection method for streaming contents that makes use of traffic patterns extracted from only traffic volume information obtained from routers. However, as the packet delay and jitter increase, its detection accuracy significantly decreases. To overcome this issue, we applied the Dynamic Programming (DP) matching scheme in our previously proposed technique by expanding and contracting traffic patterns appropriately. Finally, we evaluated the performance of our enhanced approach under the influence of packet delay, packet loss, and background traffic environment.

References

1. Youtube - Broadcast Yourself, `http://www.youtube.com/`
2. Yang, Z., Ma, H., Zhang, J.: A dynamic scalable service model for SIP-based video conference. In: Proc. 9th International Conference on Computer Supported Cooperative Work in Design, Coventry, UK, May 2005, vol. 1, pp. 24–26 (2005)
3. Shimakawa, M., Holed, D.P., Tobagi, F.A.: Video-conferencing and Data Traffic over an IEEE 802.11g WLAN using DCF and EDCA. In: Proc. International Conference on Communications (ICC), Seoul, Korea, May 2005, vol. 2, pp. 16–20 (2005)
4. Lin, E.I., Eskicioglu, A.M., Lagendijk, R.L., Delp, E.J.: Advances in Digital Video Content Protection. Proc. IEEE 93(1), 171–183 (2005)
5. Peltotalo, J., Harju, J., Jantunen, A., Saukko, M., Vaatamoinen, L.: Peer-to-Peer Streaming Technology Survey. In: Proc. 7th International Conference on Networking, Cancun, Mexico (April 2008)
6. Zwicky, E.D., Cooper, S., Chapman, D.B.: Building Internet Firewalls, 2nd edn. O'Reilly, Sebastopol (2002)
7. Dobashi, M., Nakayama, H., Kato, N., Nemoto, Y., Jamalipour, A.: Traitor Tracing Technology of Streaming Contents Delivery using Traffic Pattern in Wired/Wireless Environments. In: Proc. IEEE Global Telecommunications Conference, San Francisco, California, USA, November/December (2006)

8. RealNetworks, Product and Services > Media Servers, http://www.realnetworks.com/products/media_delivery.html
9. Apple, Open Source - Server - Streaming Server, http://developer.apple.com/opensource/server/streaming/
10. Geiger, D., Gupta, A., Costa, L.A., Vlontzos, J.: Dynamic Programming for Detecting, Tracking, and Matching Deformable Contours. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(3), 294–302 (1995)
11. Hemminger, S.: Network Emulation with NetEm. In: Proc. Linux Conference Australia, Canberra, Australia (April 2005)
12. The nuttcp development team, http://www.nuttcp.net/