

Network-Based Traitor-Tracing Technique Using Traffic Pattern

Hidehisa Nakayama, *Member, IEEE*, Abbas Jamalipour, *Fellow, IEEE*, and Nei Kato, *Senior Member, IEEE*

Abstract—Today, with the rapid advance in broadband technology, streaming technology is applied to many applications, such as content delivery systems and web conference systems. On the other hand, we must implement digital rights management (DRM) to control content spreading and to avoid unintended content use. Traitor tracing is one of the key technologies that constructs DRM systems, and enables content distributors to observe and control content reception. General methods make use of watermarking to provide users' individual information unique to each user. However, these methods need to produce many individual contents. Especially, this is not realistic for real-time streaming systems. Furthermore, watermarking, which is a key technology adopted by contemporary methods, has known limitations and attacks against it. This is why the authors have proposed a method to monitor the content stream using traffic patterns constructed from only traffic volume information obtained from routers. The proposed method can determine who is watching the streaming content and whether or not a secondary content delivery exists. This information can be also used with general methods to construct a more practical traitor-tracing system. A method to cope with random errors and burst errors has also been investigated. Finally, the results of simulation and practical experiment are provided demonstrating the effectiveness of the proposed approach.

Index Terms—Digital rights management (DRM), pattern recognition, streaming contents, traffic pattern, traitor tracing.

I. INTRODUCTION

IN recent years, with the rapid advance in broadband technology, streaming technology has been applied for many applications, such as content delivery systems, video conference systems, e-learning systems, remote diagnosis systems, and so on [1], [2]. Video conference systems, which use local area networks (LANs) and the Internet, are a new technology and replace previous systems which use leased lines. For example, e-learning is already used for educational purposes in companies. The most important feature of this kind of system is that a person can obtain many kinds of lectures at any time without inviting instructors. This increases the efficiency of self-learning and reduces the cost of the education. However, these useful streaming systems can be threatened by many security

problems. An example of these problems is the use of content without knowledge of the content holders and content providers. Such uncomprehended uses involve retransmission of content as well as unauthorized uses. Hence, there is a high expectation on the digital rights management (DRM) technology which manages content usage.

Generally, an idealized DRM technology can continually manage content delivery and user's operations throughout the content's commercial life cycle [3]–[5]. In the protection of content with DRM technology, content is encrypted, and decryption keys are securely transmitted to users. Users can extract the content from capsules using received keys, which avoid data interception [6]–[8]. However, since this kind of protection is not able to restrain distributions of decrypted contents or use of plagiarized keys, content providers also implement the active management technology to monitor the content usage [9]–[11]. This technology is called traitor-tracing technology, which enables content providers to track the content usage.

General traitor-tracing methods use digital watermarks and encryption keys to observe propagation of the content [11]–[14]. The concept of general traitor tracing is to assign content with an individual identity. Ideally, watermarks embedded in the content are different from one other. However, this method requires a large amount of computations to encode many contents and to embed many watermarks. Since these costs are critical for real-time streaming, the reduction of these computations has been attempted in [12] and [13]. In addition to the computation, the shortcomings of watermarking, such as resistance to bit errors and the conversion of contents, are also serious issues [15], [16]. Furthermore, there are also known attacks against watermarks and cracking methods for decoders [6], [17]–[20]. An example of these attacks is the removal attack, which ruins watermarks by adding noise into the contents.

Therefore, to improve the success rate of tracking traitors, we need to discuss a technique to narrow the list of users who may be traitors by monitoring content distribution. As well as using alone, this technique aims to assist the general traitor-tracing method. To prevent a user's operation to interrupt monitoring as the case of attacks against watermarking, the monitoring process should be implemented based on information obtained at routers in the middle of the streaming path. This is why we proposed a concept to track a user's content receptions using information about traffic amounts [21]. With this concept, the content provider can detect who is watching the streaming content and whether the content is secondarily distributed or not. This is useful to narrow the list of traitors and is valuable for traitor-tracing methods which assume that they can make use of a technique to find in real time a possibility of the secondary content distribution before the accurate tracking process. Moreover, since the proposed concept does not make use of informa-

Manuscript received May 14, 2009; revised February 14, 2010; accepted March 01, 2010. Date of publication March 29, 2010; date of current version May 14, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Wade Trappe.

H. Nakayama is with the Department of Electronics and Intelligent Systems, Tohoku Institute of Technology, Sendai-shi, 982-8577, Japan (e-mail: hidehisa@m.ieice.org).

A. Jamalipour is with the School of Electrical and Information Engineering, University of Sydney, Sydney NSW 2006, Australia.

N. Kato is with the Graduate School of Information Sciences, Tohoku University, Sendai-shi, 980-8579, Japan.

Digital Object Identifier 10.1109/TIFS.2010.2046961

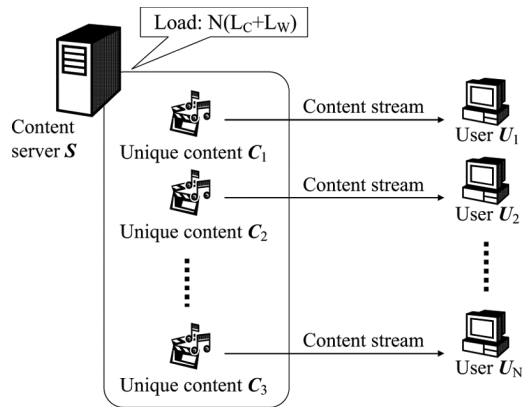


Fig. 1. Example of simplest traitor tracing.

tion within packets, it does not cause privacy issues [22]. A brief summary of the proposed concept is provided below.

The bit rate of variable bit rate (VBR) streaming content changes corresponding to changes in the content. When the content is delivered to the user as a streaming content, the traffic amount observed at routers near the server and the user represents a pattern unique for each content just as human fingerprints. By comparing user-side patterns to server-side ones, we are able to determine whether or not a user is watching the targeted content. The following is a brief procedure to determine whether or not a user is watching the content.

First, a server-side pattern of the streaming content is generated based on the observation of the traffic amount at the router near the content server. Next, a user-side pattern is generated in the same manner. Finally, the management server compares the user-side pattern with the server-side one to make judgments on the content reception of the user. This process requires no operation on the user's computer. In principle, this makes it impossible for the user to tamper with the tracing process. Additionally, since the proposed method makes use of only the information about traffic amount, it can be implemented with a lower computational cost than traditional methods which use watermark technology.

We refer to traitor-tracing technologies which are implemented in the midway between the user and server as network-based traitor-tracing technologies. We have introduced the concept of this new traitor-tracing method in [21]. In this paper, the detail of the proposed method, simulation results to evaluate the effectiveness in various environments, and experimental results in actual environments are shown.

The rest of this paper is organized as follows. In Section II, we introduce an overview of the DRM technology, the traitor-tracing technology, and the research works on these technologies. Section III presents details on technologies, which construct the proposed traitor-tracing system. The experiments with the real network environment are presented in Section IV. In Section V, we introduce the results of simulations with Network Simulator 2 (NS-2). Finally, Section VI concludes the paper.

II. RELATED WORKS

A brief yet comprehensive summary of DRM technology is provided by Eskicoglu and Lin *et al.* [5], [6]. DRM technology consists of several technologies, such as encryption [23]–[26]

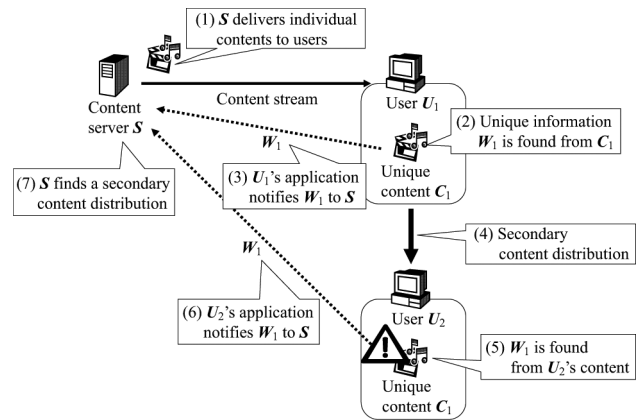


Fig. 2. Technique to verify the existence of secondary content distributions and to find the runoff source of it.

and access conditioning[27]. The traitor-tracing technology is also one of the key technologies that constructs DRM systems, and is used to monitor the content usage and to confirm that a user appropriately uses contents.

The simplest traitor-tracing system is considered as shown in Fig. 1. The following is the procedure adopted by this system [11].

- 1) The content provider embeds unique information¹ into the content using digital watermarking, and produces copies of the content.
- 2) Each copy of the content is delivered to different users.
- 3) The user's application analyzes the content's data and re-assembles the embedded information.
- 4) The user's application notifies that he or she is watching the content, according to the extracted information.

Fig. 2 shows the mechanism to verify whether or not the secondary content distributions exist and to find the runoff source of the secondary content. First, unique content C_1 is delivered to user U_1 . Next, U_1 's application finds watermarked information W_1 and notifies it to server S . When an unauthorized secondary content distribution is conducted between U_1 and U_2 , W_1 is also found from U_2 's content. Next, U_2 's application notifies W_1 to S . Finally, S finds duplicated W_1 and concludes that the content is secondarily delivered from U_1 to U_2 indicated as a caution sign in Fig. 2.

However, this traitor-tracing method has two issues. First, this method requires a large amount of computation to encode content and to embed watermarks, since each user receives an individual content. For instance, if the load of content encoding and watermarking are L_C and L_W , respectively, the total load to deliver content which has traceability of traitors should be at least $N \times (L_C + L_W)$. Here, N is the number of content users. This is especially not realistic for real-time streaming systems. This is why a method to assist tracking traitors and to reduce the load is strongly needed [12], [13].

Second, the watermark, which is a key technology of traitor tracing and is used to embed information into the host signal [28]–[31], is not without its shortcomings [15], [16], [32]. Furthermore, there are known attacks against watermarks, such as the removal attack, collusion attack, copy attack, closest-point

¹In some methods, unique encryption keys represent unique information.

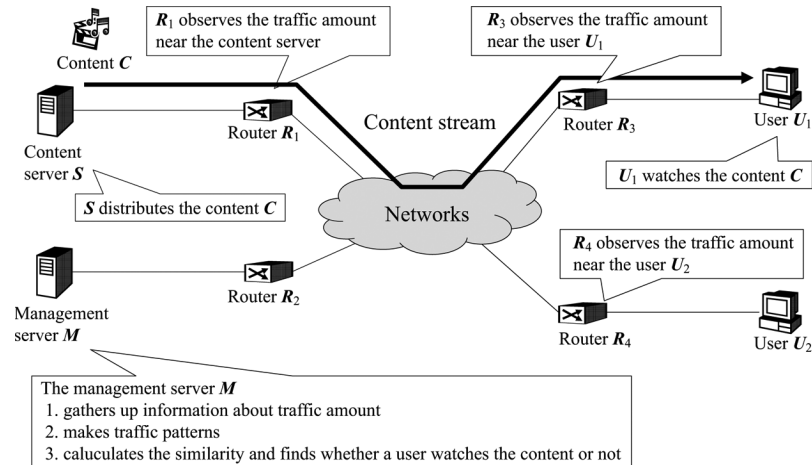


Fig. 3. Outline of the proposed scheme.

attack, and sensitivity attack [19], [20]. Voloshynovskiy *et al.* introduced a stochastic formulation of attacks [33]. According to their work, general watermarks perform poorly against the benchmark that they proposed. Consequently, when the watermark is used over networks, it causes some problems because of unclear network environments and users.

This is why traitor-tracing methods with watermarking require an additional technique to tighten the scope of application by narrowing down the list of users who may be traitors. Additionally, it is desirable to interoperate with a method to find suspicious content distributions without watermarking. In particular, we will consider detection of “illegal relay” by avoiding deciphering and decoding contents in watermarking. Towards this end, we envision a method to track the content stream using information about traffic amount observed at routers. Our method circumvents the need of decrypting and decoding the content, and this is the major contrast with respect to the watermarking technique. The proposed method can be not only used alone, but also used with general traitor-tracing methods in order to enhance the security.

In Section III, we introduce the proposed tracing system for streaming multimedia contents.

III. NETWORK-BASED TRAITOR TRACING FOR STREAMING CONTENTS

At first, we introduce a brief summary of the proposed system. The outline of the proposed scheme is shown in Fig. 3, where the content server S distributes the content C . Both users U_1 and U_2 receive the content C . The routers R_1 , R_3 , and R_4 observe the amount of traffic. For example, router R_1 monitors the flow of the content server S and sums up the sizes of packets. Here, the proposed system makes use of packet filtering about IP addresses to specify the content server’s flow. The obtained information at routers is sent to the management server M with protocols, such as Simple Network Management Protocol (SNMP) and Internet Control Message Protocol (ICMP). Even when a user simultaneously watches two video streaming contents, it is possible to differentiate them based on the used port numbers since each streaming application employs a different port number for its transmission.

Next, the management server M constructs traffic patterns at both the server and user sides. Here, the router R_1 near the content server observes the traffic amount during the time the content C is delivered, and the server-side traffic pattern is constructed using this information. On the other hand, the user-side routers R_3 and R_4 observe the traffic amount for a few minutes and the management server constructs the sort traffic patterns at the user-side.

After that, with the results of comparison of these two traffic patterns, the management server M finds which user is watching the content C . The content provider can make use of this data to manage the content delivery, using other DRM technologies. In Fig. 3, the proposed system allows us to know that only U_1 receives C . Moreover, even if the user U_1 retransmits the stream of content C to the user U_2 , our system detects a stream of the content C traversing the router.

As an example, the proposed system can detect the flow of steaming content in a variety of networks. The proposed system is suitable for many types of applications, which use streaming technology, such as Helix and Quicktime servers provided by Real Networks and Apple, respectively. Note that this system is implemented for only streaming content, not downloaded content. Additionally, the proposed system is applied to a certain degree of limited networks where monitoring is allowed. For instance, video streaming systems using corporate LANs is a suitable case. With the proposed system, administrators of streaming systems can oversee the flow of contents and know who is watching them without monitoring the streaming data. This reduces not only the security risk on theft by a third party but also by the administrator.

In this section, we propose and explain a method to detect streaming of content by observing the amount of traffic at routers. In Section III-A, we first introduce the definition of a traffic pattern and similarity, which are core technologies of the proposed method, before introducing the whole streaming detection system. In Section III-B, the scheme to compare the server-side and the user-side traffic patterns is presented. Finally, we give an account for the determination threshold to determine whether or not the user is watching the content in Section III-C.

A. Definition of a Traffic Pattern and Similarity

In this paper, we focus on VBR streaming content and make use of a “traffic pattern” or the fluctuation of traffic amount, to find the streaming content flow.

The traffic pattern is defined as the amount of traffic for a certain time-slot Δt [sec] and expressed as an N -dimension vector in the following expression:

$$\mathbf{X} = (x_1, x_2, \dots, x_N)^t, \quad T = N\Delta t. \quad (1)$$

Here, T [sec] is the whole length of the traffic pattern and N [slot] is the number of “time slots.”

In the proposed method, the server-side router observes the traffic amount during the entire time that the content is being delivered and the server-side traffic pattern is expressed as $\mathbf{X}_S = (x_1, x_2, \dots, x_S)^t$ according to (1). The user-side traffic pattern is expressed as $\mathbf{Y}_U = (y_1, y_2, \dots, y_U)^t$. Here, S and U are the number of time-slots and the length of the user-side observation is shorter than that of the server-side, i.e., S [slot] $>$ U [slot].

We pay attention to the similarity between the user-side pattern \mathbf{Y}_U (U [slot]) and the server-side pattern \mathbf{X}_S (S [slot]). To calculate the partial similarity, a partial pattern \mathbf{X}_U , whose length is U [slot], is snipped off from the server-side pattern. Before calculating the similarity of two patterns, \mathbf{X}_U and \mathbf{Y}_U are normalized as

$$\mathbf{X}'_U = \begin{pmatrix} \frac{(x_1 - \bar{x})}{s_x} \\ \frac{(x_2 - \bar{x})}{s_x} \\ \vdots \\ \frac{(x_U - \bar{x})}{s_x} \end{pmatrix}, \quad \mathbf{Y}'_U = \begin{pmatrix} \frac{(y_1 - \bar{y})}{s_y} \\ \frac{(y_2 - \bar{y})}{s_y} \\ \vdots \\ \frac{(y_U - \bar{y})}{s_y} \end{pmatrix}. \quad (2)$$

Here, \bar{x} and \bar{y} are the means of each vector. s_x and s_y are the standard deviations. After normalizing, the means of \mathbf{X}'_U and \mathbf{Y}'_U are zero and the variances are 1. Finally, with (2), the similarity of the user-side pattern and the part of the server-side pattern is calculated by the following equation:

$$R_{XY} = \frac{\mathbf{X}'_U{}^t \mathbf{Y}'_U}{\sqrt{\|\mathbf{X}'_U\|^2 \cdot \|\mathbf{Y}'_U\|^2}}. \quad (3)$$

This is equivalent to the cross-correlation coefficient. When two patterns are similar to each other, R_{XY} approximates one.

The above-mentioned process obtains similarity between patterns which are of the same length. However, in the proposed method, we need to compare \mathbf{X}_S and \mathbf{Y}_U whose lengths are different from each other (S [slot] $>$ U [slot]). This is why the proposed method makes use of a “window” to snip off a partial pattern \mathbf{X}_U (U [slot]), which is of the same length as the user-side traffic pattern \mathbf{Y}_U . With the partial pattern of the server-side traffic pattern, the proposed method calculates similarity R_{XY} by moving the window from left to right on the server-side traffic pattern \mathbf{X}_S . In Section III-B, we explain the details of comparison between \mathbf{X}_S and \mathbf{Y}_U .

B. Comparison of Traffic Patterns Using Similarity

The outline of comparison of traffic patterns is shown in Fig. 4. The comparison of traffic patterns consists of three steps.

First, in Step 1, we place a window (U [slot]), which snips off a partial pattern \mathbf{X}_U from the server-side traffic pattern \mathbf{X}_S .

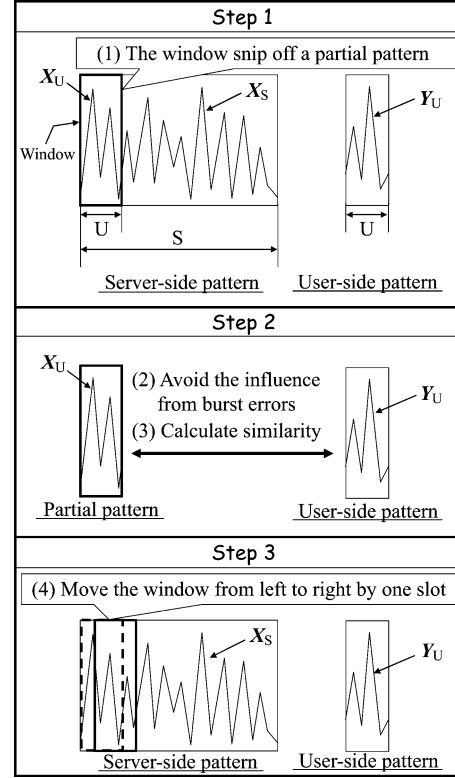


Fig. 4. Outline of the comparison of traffic patterns.

Next, in Step 2, similarity between the partial pattern \mathbf{X}_U and the user-side pattern \mathbf{Y}_U is calculated. In this step, in addition to the calculation of similarity, the proposed method also conducts a countermeasure against burst errors. We will introduce this later. In Step 3, the window is moved from left to right by one slot. The proposed method repeats these three steps until the window reaches the rightmost part of the server-side pattern, and finally obtains $(S - U + 1)$ values of similarity.

Next, we explain a way to avoid the influence from burst errors and also to evaluate the result of comparison.

1) *Countermeasure Against Burst Errors*: Burst errors deteriorate the user-side traffic pattern, which makes it difficult to normally calculate similarity. To avoid the influence from burst errors, we transform the partial pattern \mathbf{X}_U and the user-side traffic pattern \mathbf{Y}_U . The outline of the transformation is shown in Fig. 5.

As shown in the “original patterns” section in Fig. 5, the user-side traffic pattern can partially sag because of the influence of burst errors. In Step 1, we remove elements within the user-side traffic pattern whose values are equal to or less than a certain threshold T_P . For example, in Fig. 5, three elements y_i , y_{i+1} , and y_{i+2} , which are lower than the threshold, are removed. In Step 2, we remove elements within the partial pattern in the window, according to the user-side removal process. More specifically, we remove the elements in the same positions as removed elements on the user-side. In Fig. 5, three elements x_i , x_{i+1} , and x_{i+2} are removed. In Step 3, the remaining elements are connected. Consequently, we obtain new patterns, $\mathbf{X}_{U'}$ (U' [slot]) and $\mathbf{Y}_{U'}$ (U' [slot]).

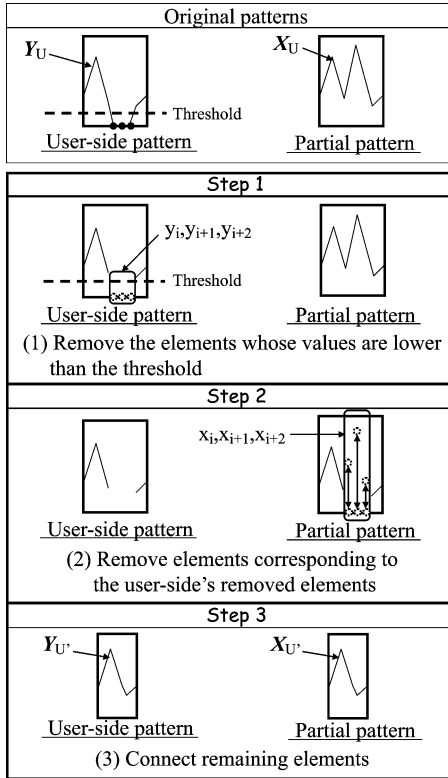


Fig. 5. Outline of the avoidance of the influence from burst errors.

These three steps are conducted before calculating similarity. When there is no burst error, these processes do not cause any change on traffic patterns.

2) *Evaluation of Results of the Comparison:* Fig. 6 shows an example of results of the comparison in Fig. 4. Figs. 6(a) and (b) demonstrate the matching results in case of the same contents and different contents, respectively. The horizontal and vertical axes refer to the time slot and similarity, respectively. The profile is constructed from the histogram of the obtained similarity values.

When a large value exists in these values [as shown in Fig. 6(a)], it means that the user-side traffic pattern is similar to the partial pattern of the server-side traffic. In this case, the management server decides that the user is watching the content. We refer to such a user-side pattern as a “matched pattern.” To decide whether a significantly large value exists or not, the proposed method uses a threshold T_R . Generally speaking, the performance of the proposed method relies on appropriateness of the threshold T_R . We use a dynamic method to decide the value of T_R . In the following subsection, we explain the way to calculate the threshold dynamically.

C. Dynamic Detection Threshold

Packet loss and jitter influence the proposed method. For instance, if packets drop in the middle of content streaming, the user-side traffic pattern will deteriorate totally or partly, so that the maximum value of similarity becomes smaller. Likewise, jitter deteriorates the user-side traffic pattern. Thus, in this case, it is difficult to find a peak in the similarity graph with the static

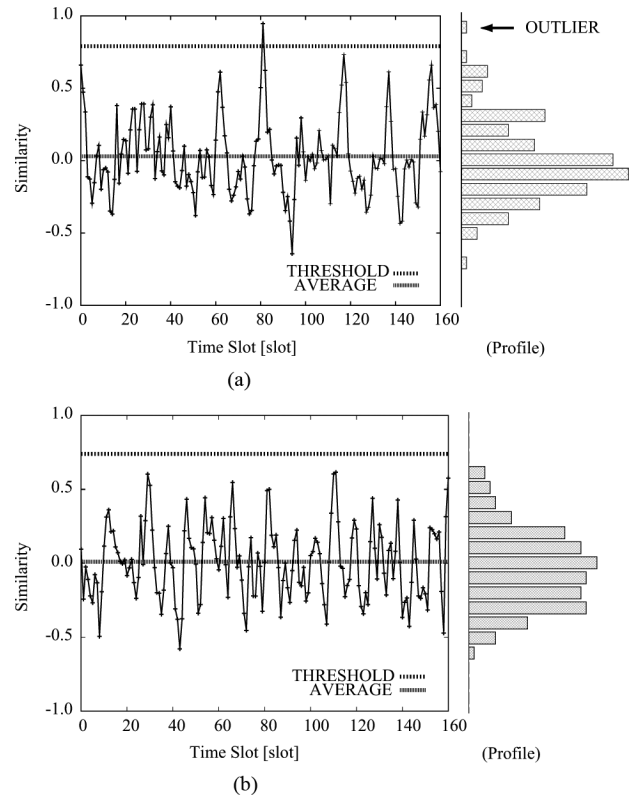


Fig. 6. Examples of the similarity graph. (a) Matching result of same contents. (b) Matching result of different contents.

determination threshold. That is why, to accommodate the proposed method for a changing network environment, a dynamic threshold is required.

The majority of similarity data constructed by the proposed method is small and its distribution is considered to be normally distributed around zero, since the distribution of cross-correlation coefficient values of two different random waveforms is approximated to a normal distribution [34], as shown in Fig. 6. On the other hand, the similarity of two patterns similar to each other is large, compared to other cases. Such a large value is called an “outlier.” The dynamic determination threshold to detect an outlier among some values is defined as the following equation:

$$T_R = \min(\mu_R + 4\sigma_R, 1.0). \quad (4)$$

Here, μ_R and σ_R are the mean and the variance of the values of similarity, respectively. They are computed based on the profile. Each user traffic pattern has its own μ_R and σ_R values. Generally speaking, it is possible to balance the trade-off between the detection ratio and false-positives by adjusting the coefficient of σ_R . A relatively large value of this coefficient guarantees a low false-positive ratio at the cost of a low detection ratio. On the other hand, smaller values of this coefficient result in the reverse effect. According to Chebyshev’s inequality, the probability that the data are equal or greater than the threshold T_R is about 6%. Consequently, a similarity greater than T_R is considered to be an outlier which is numerically distant from the rest of the values on the profile [as shown in Fig. 6(a)].

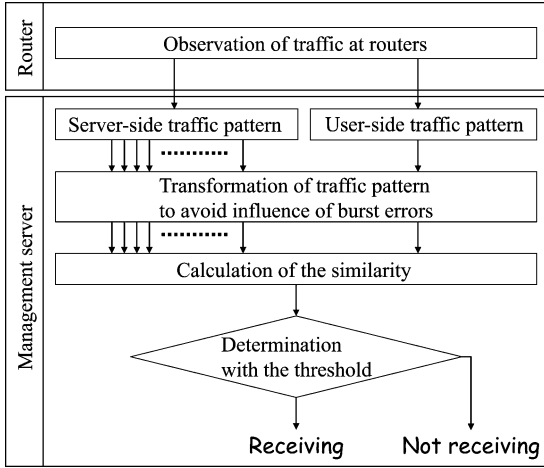


Fig. 7. Algorithm of the proposed method.

Fig. 7 shows the proposed determination algorithm which can be applied to various environments including wired/wireless environments. The content provider can implement the proposed system without taking into account where users are receiving the content.

The proposed method does not require any additional computation cost at the content server. Instead, we place a management server to calculate the similarity of traffic patterns. When we assume that the number of multiplications conducted in the calculation of similarity is M , the total number of multiplications conducted in the proposed method is about $(S - U + 1) \times M$. Here, S and U are the lengths of the server-side traffic pattern and the user-side traffic pattern, respectively. M is a constant, which depends on U . This calculation cost is much less than the cost of encoding and watermarking.

IV. EXPERIMENT WITH THE REAL NETWORK ENVIRONMENT

A. Experiment Setup

1) *Experiment Parameters*: We performed an experiment with two items of streaming contents encoded in MPEG4 to verify that the stream of content is detected by the proposed method. Fig. 8 shows the topology of this experiment and Table I presents the specification of computers used to construct the streaming system. Other experimental parameters are shown in Table II. In this experiment, since we implemented packet filtering, there is no significantly large background traffic except for the content.

2) *Experimental Scenario*: First, we independently deliver the contents C_1 and C_2 to any user to construct the server-side traffic patterns. While the contents are being delivered, routers R_1 or R_2 will observe the traffic amount until the end. After that, the management server M makes the server-side traffic patterns, X_{S_1} and X_{S_2} , based on the information obtained at the routers.

Next, user U_1 receives one of the contents. In this experiment, it is previously defined that user U_1 receives the content C_1 . During a certain period in the content delivery to the user U_1 , the router R_4 observes the traffic amount. The management server constructs the user-side traffic pattern Y_{U_1} based on information provided by the router R_4 .

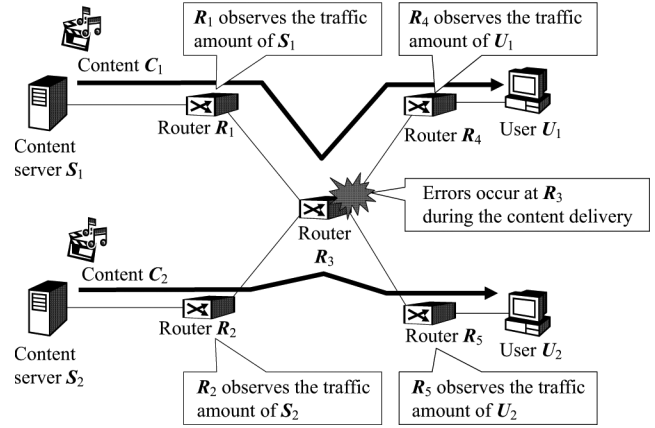


Fig. 8. Topology of the experiment.

TABLE I
COMPUTER CONFIGURATION

Device	Specification
CPU	Intel Pentium4 2.4GHz
Memory	256Mbytes
OS/Kernel	FedoraCore3/Linux2.6.11
NIC	Intel82801DB PRO/100 VE(CNR)
NIC driver	e100 3.3.6-k2-NAPI

TABLE II
PARAMETERS OF THE EXPERIMENT

Parameter	Value
Length of the server-side traffic pattern	180 [sec]
Length of the user-side traffic pattern	20 [sec]
Length of the time-slot	0.2 [sec]
Threshold about the packet size T_P	2 [kbyte]
Bit-rate of the content	750 [kbps] (approx.)
Bandwidth of links	100 [Mbps]

Finally, the management server M compares the user-side traffic pattern Y_{U_1} with two server-side traffic patterns, X_{S_1} and X_{S_2} , to decide which content user U_1 is receiving. In this experiment, since the user U_1 receives the content C_1 , a significant peak should be seen in the similarity graph.

We conducted experiments in not only the ideal wired environment but also the pseudowireless environment where random errors and burst errors occur. The rates of random and burst errors were 10% and 30%, respectively.

B. Experimental Result Without Errors

Fig. 9(a) shows the similarity between the traffic patterns of the server S_1 and the user U_1 . Fig. 9(b) shows the similarity between the traffic patterns of the server S_2 and the user U_1 . The horizontal axis refers to the time-slot and the vertical axis indicates the similarity. In Fig. 9(a), a significant peak greater than the threshold, T_P is found around 400 [slot], which means that the user is receiving the content C_1 . On the other hand, there is no peak in Fig. 9(b). Since it was defined that the user will watch content C_1 , this result is considered to be sufficient.

C. Experimental Result With Random Errors

Fig. 10(a) and (b) shows the results of the experiment where 10% of the random errors occur at intermediate routers.

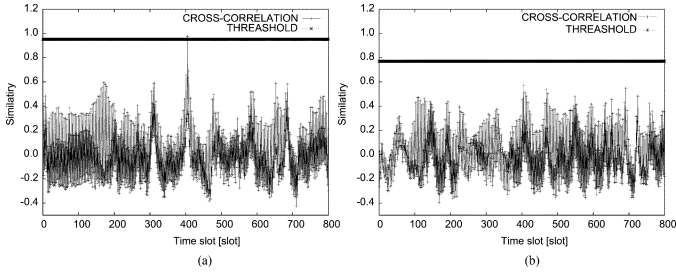


Fig. 9. Similarity without errors (a) between the traffic pattern of S_1 and U_1 and (b) between the traffic pattern of S_2 and U_1 .

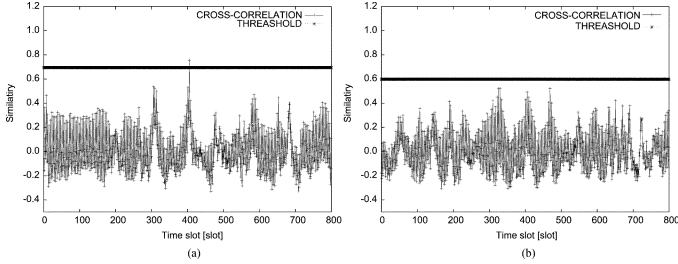


Fig. 10. Similarity with random errors (a) between the traffic pattern of S_1 and U_1 and (b) between the traffic pattern of S_2 and U_1 .

Fig. 10(a) shows the similarity between the traffic patterns of the server S_1 and the user U_1 . Fig. 10(b) shows the similarity between the traffic patterns of the server S_2 and the user U_1 .

As the result of Section IV-B, the proposed method accurately decided that user U_1 is watching the content C_1 , in spite of its lower peak than Fig. 9(a). The size of the peak in Fig. 10(a) is about 0.75, while the size of the peak in Fig. 9(a) is about one. The reason why the proposed method can accurately detect the content stream in spite of the lower peak is the use of the dynamic determination threshold. If the determination threshold was defined as the value which is generally considered to be sufficiently large, we would not be able to find out the peak of the similarity graph.

D. Experimental Result With Burst Errors

Next, we present experimental results in the environment where the burst errors occur. The burst errors keep dropping packets for a few seconds, so that the traffic pattern is distorted partially. This causes an adverse affect on generating the similarity of traffic patterns accurately.

Fig. 11(a) shows the result of the comparison between the server S_1 's and the user U_1 's traffic patterns. Fig. 11(b) shows the result of the comparison between the server S_2 's and the user U_1 's traffic patterns. As the case of the random error, the peak of the similarity graph is significantly lower than that presented in Fig. 9(a). However, thanks to the dynamic determination threshold and the burst error avoidance in Section III-B1, the proposed method accurately detects the peak of similarity.

E. Experiments Using Various Streaming Contents

For validating the effectiveness of the proposed method, in this experiment, five types of videos including documentary, animated show, entertainment, baseball game broadcast, and news broadcast are collected and used for verification. We first confirmed that the proposed method is indeed able to detect the contents of different video types. In addition, we investigated also

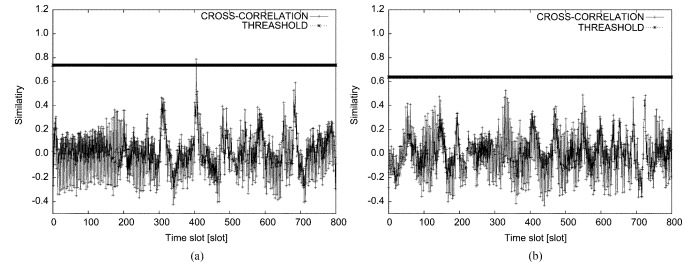


Fig. 11. Similarity with burst errors (a) between the traffic pattern of S_1 and U_1 and (b) between the traffic pattern of S_2 and U_1 .

the situation whereby the contents are relatively similar (in particular in the case of the considered news broadcast instances). Examples of two instances of different news broadcasts, in terms of matching scores, are shown in Fig. 12. The topology, the parameter, and the procedure are the same as those considered in the experiment described in Section IV-A. The horizontal and vertical axes of this figure refer to the number of consecutive segments and maximum similarity, respectively. The two diagonal graphs indicate the case of a user watching the corresponding streaming contents. The figure clearly demonstrates that all maximum similarities exceed the thresholds, which implies that the detection is, indeed, successful. Meanwhile, the remaining two graphs indicate that almost no false-positive occurred for most of the segments. Besides the samples shown in Fig. 12, we also collected an additional 100 news broadcast samples and conducted the same experiment on these. The empirical results suggest a general trend as follows: the higher the user traffic pattern change, the higher the similarity.

F. Experiments Against Flooding Attacks

Indeed, in a real network setting, various types of attacks may target the content server. While it is rather difficult to account for all the attacks in literature in this manuscript, we focus on investigating the impact of the more complicated attacks. For example, denial-of-service (DoS) and distributed DoS (DDoS) attacks are currently common threats against content servers on the Internet. Both DoS and DDoS attacks interrupt network services by means of packets flooding. However, since both these attacks increase network packets on the whole, they do not affect the traffic patterns seriously. Increasing the traffic volume as a whole in this manner is interpreted by our proposed method as adding background traffic on top of the original streaming data. Consequently, our proposed method can preserve the effectiveness of discriminating different contents under DoS and DDoS scenarios. In addition, the Internet routers, in general, have the capability to prevent packet flooding. In this section, we consider the impact of a more complicated scenario, namely the pulsing DoS (PDoS) attack. Indeed, PDoS is currently known as the most representative and intimidating type of DoS attack. Due to the pulse-like traffic generated by PDoS, it is trickier than its more typical DoS or DDoS counterparts. Since the total amount of packets produced by a PDoS attack is relatively less than DoS or DDoS, many security systems are vulnerable to this attack. As a result, the traffic patterns may not preserve their original forms under PDoS attacks. As experimental conditions, we used the same as in Section IV-E, with PDoS environments with cycle of 40 [sec] and duty-ratio of 0.5, respectively. Fig. 13 demonstrates the experimental results of the two different types

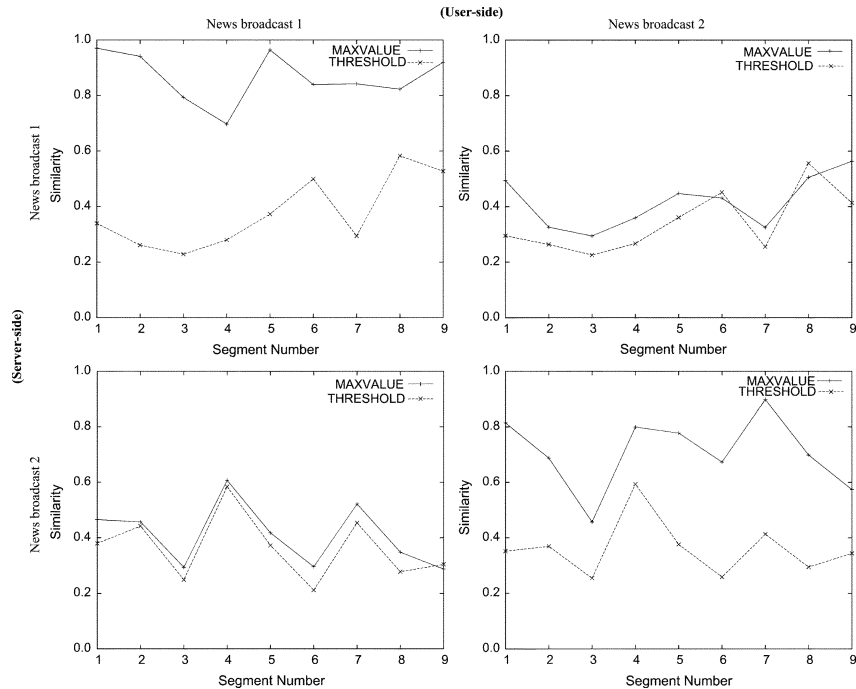


Fig. 12. Matching scores of two streaming contents.

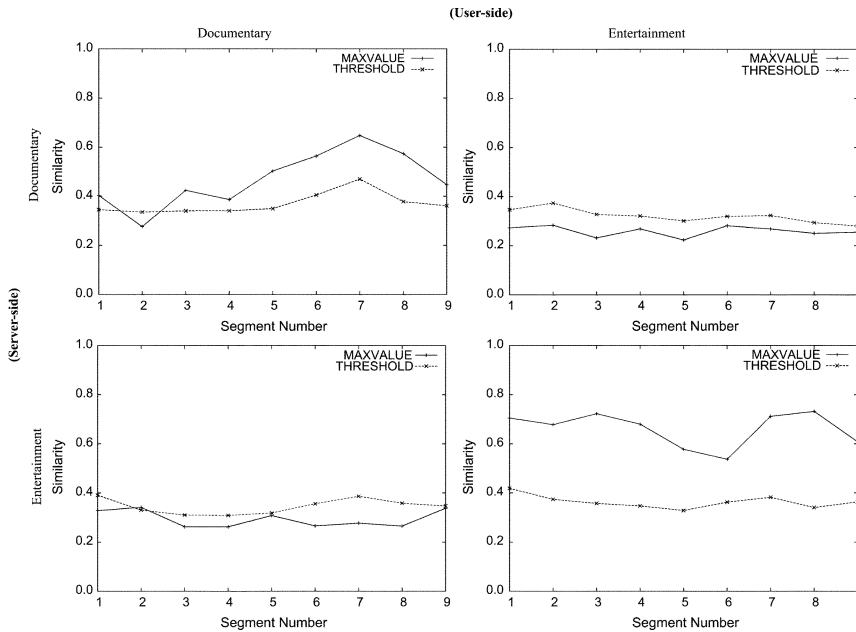


Fig. 13. Matching scores of two streaming contents under the PDoS attack.

of contents, documentary and entertainment. Regardless of the position number of the same content, most of the similarities exceed the thresholds. For the opposite case, most of the similarities remain below the thresholds. Thus, the proposed method can also be applied in such an environment affected by more subtle attacks such as PDoS.

G. Other Issues

Our proposed scheme can be applied to many VBR-based codecs, e.g., H.264 and so forth. Indeed, the most recent video coding standard is H.264 SVC, which adjusts the sending ratio according to the change of the available network bandwidth. In

case of contents being encoded with H.264 SVC codec, our proposed method is also applicable. In principle, the content server can know the streaming rate on the user-side before sending the content. Therefore, we can effectively compare the user-side traffic patterns with the server-side one. In order to evaluate the performance of the proposed scheme using H.264 SVC, we carried out an experiment in which contents are prepared with different bit-rates on the server for emulating the functionality of H.264 SVC. In other words, four kinds of H.264 AVC bit-rates are prepared, namely (250, 500, 750, and 1000 [kbps]). The effectiveness of the proposed method is verified at nine positions for each bit-rate, as shown in Figs. 14 and 15. As evident from these results, almost all the points of maximum detection

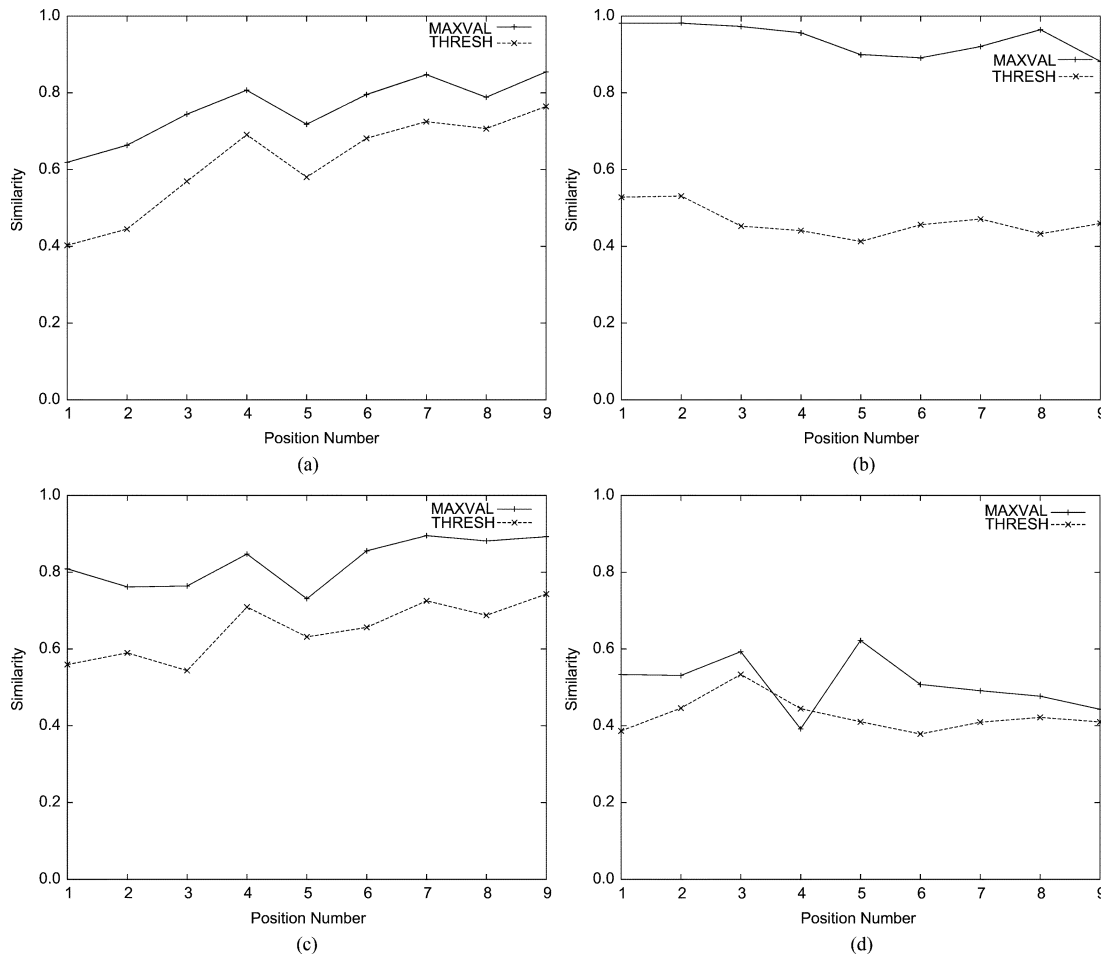


Fig. 14. Matching scores of the H.264 contents "Documentary" (a) for 1000 [kbps], (b) for 750 [kbps], (c) for 500 [kbps], and (d) for 250 [kbps].

values exceed the corresponding threshold levels. This verifies that the proposed approach detects the illegal relay regardless of variable bit-rates. However, as shown in Figs. 14(d) and 15(b), the maximum values approach and occasionally go below the corresponding threshold values. This is because of the fact that the traffic pattern of each content becomes relatively flat at a specific bit-rate. As a consequence, the proposed method fails to correctly detect illegal content in these cases. In particular for the documentary and entertainment streaming scenarios, the success rates achieved by our proposed method still maintain 97% and 88%, respectively. Thus, in general, the proposed method still maintains good performance even in different bit-rate environments.

Next, our system only needs to transfer a small piece of information from each user-side to the management server. The impact on networks can be considered rather small, e.g., if we need to monitor 1000 end nodes and the user-side traffic segment is 20 s long which is equivalent to 2 KB, the total management overhead generated by the proposed system is about 2 MB. This finding indicates that bottlenecks may not occur or can even be avoided by setting management servers on different locations of the considered networks.

V. PERFORMANCE ANALYSIS WITH SIMULATIONS

We conducted simulations to verify the performance of the proposed method using the NS-2 [35]. In the following section,

simulation results to verify the influence of the random errors and background traffic are presented. Additionally, a simulation with different contents, such as movies, animations, and news, was conducted to confirm the applicability of the proposed method regardless of types of contents. In the simulation, we used the precision, the recall, and the f-measure as the index to evaluate the performance. First, the introduction of these indices are shown.

A. Indices to Verify the Performance of the Proposed Method

The precision and the recall 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. For instance, the precision approaches a small value when the system inadvertently determines that a user is watching the content despite the fact that the user is NOT actually watching it. On the other hand, the recall has a small value, when the system inadvertently missed a user who is watching the content. Large values of these indices indicate high performance of the method. However, because these indices generally have a trade-off relationship, we also use the f-measure, which is a comprehensive measurement, to evaluate the performance of our approach. The f-measure also indicates high performance, when it has a large value. These indices are

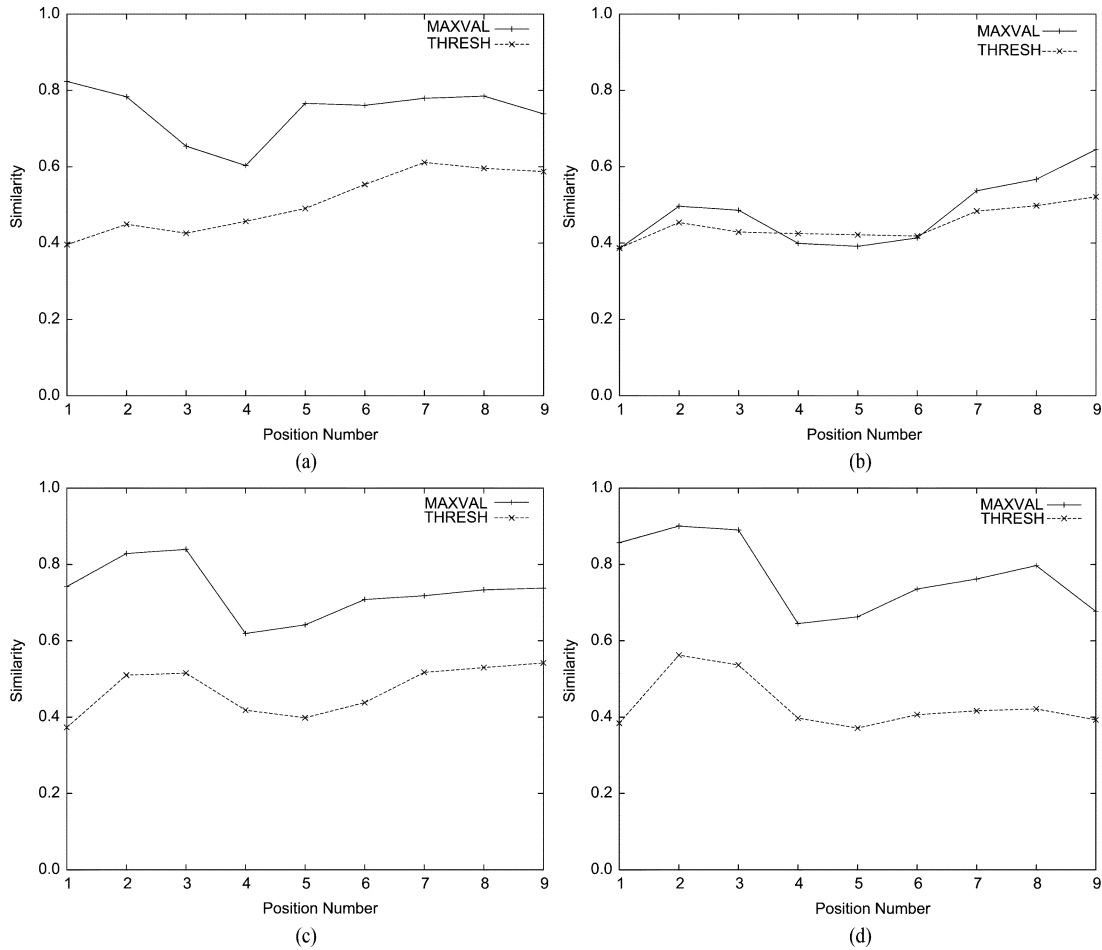


Fig. 15. Matching scores of the H.264 contents “Entertainment” (a) for 1000 [kbps], (b) for 750 [kbps], (c) for 500 [kbps], and (d) for 250 [kbps].

expressed as the following equations:

$$\text{precision : } Pr = \frac{\text{Cor}}{\text{All}} \times 100 [\%] \quad (5)$$

$$\text{recall : } Re = \frac{\text{Cor}}{\text{Wat}} \times 100 [\%] \quad (6)$$

$$\text{f - measure : } F = \frac{2 \times Pr \times Re}{Pr + Re} [\%]. \quad (7)$$

Here, Cor is the number of users who are “correctly” declared to be watching the content. All is the number of users who are declared to be watching the content, including the false positives. Wat is the number of users who are actually watching the content.

In Section V-B, we provide an explanation about the influence of random errors. Section V-C presents the influence of background traffic and Section V-D gives an account for the proposed method regardless of the content types.

B. Simulation #1: Verification of the Influence of Random Errors

1) *Objective of the Simulation:* Random errors, which transform the user-side traffic pattern causing a pattern actually matching the server-side traffic pattern, may be considered as an “unmatched” pattern by mistake. Hence, we conducted a simulation where random errors occur in the middle of the content stream to confirm this phenomenon.

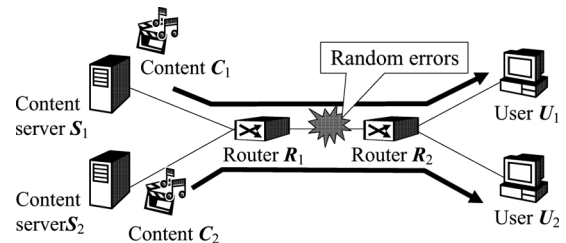


Fig. 16. Topology of simulation #1.

2) *Simulation Setup:* Fig. 16 shows the topology of this simulation. As a typical case, we use a simple dumbbell-type network to verify the influence of random errors as much as possible. As shown in Fig. 16, the content servers S_1 and S_2 deliver the content C_1 and C_2 , respectively. C_1 and C_2 are normal movies, which are selected without any specific purpose or intention. Random errors occur on the link between routers R_1 and R_2 . In this simulation, we conducted experiments changing the rate of random errors from 0% to 50%. Experiments were conducted for 200 times for a certain rate. In each experiment, the time to start observations of the user-side traffic amounts were changed not to use a particular pattern. Additionally, the simulations with fixed discrimination thresholds are conducted for comparison with the proposed method. We call this method with the fixed threshold the “static method.” Generally, when a value

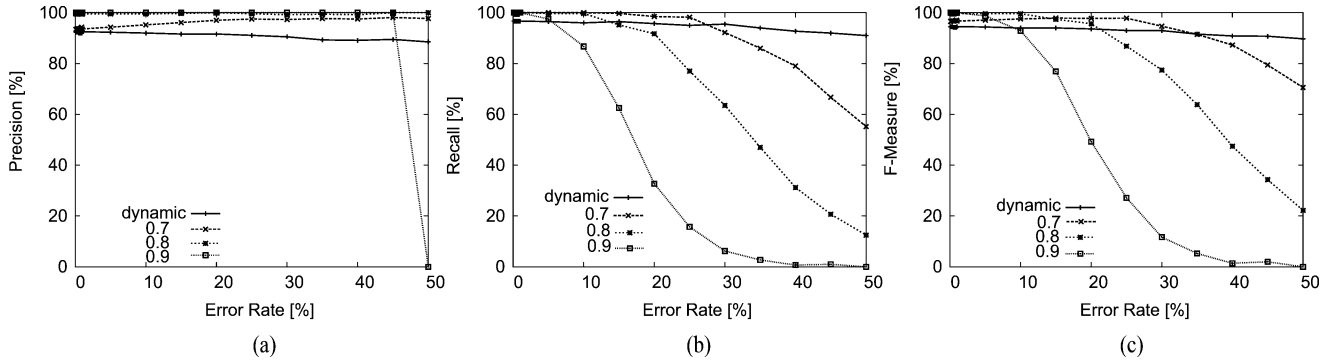


Fig. 17. Results of the simulation #1: (a) precision, (b) recall, and (c) f-measure.

of similarity is in the range 0.7–1.0, similarity of two patterns is considered to be high. Therefore, we used 0.7, 0.8, and 0.9 as a static determination threshold. The parameters are shown in Table III. In this simulation, in contrast with the transferring of video streams over realistic environments, the pseudo-VBR packet flows generated by NS-2 have only about one third of the actual speed. For gaining the same amounts of traffic from the user-side, we, therefore, used three times the length targeted at realistic environment. This is why the length of the user-side traffic pattern is changed to 60 [sec], i.e., three times more than the real length specified in Table II.

3) *Simulation Results:* Fig. 17(a)–(c) shows the results of the simulation about precision, recall, and f-measure, respectively. The horizontal axis indicates the random error rate. As shown in Fig. 17(a), the precision of the static method is significantly higher. On the other hand, as shown in Fig. 17(b), the recall of the proposed method is significantly higher. Furthermore, the f-measure of the proposed method is also higher than one of the static methods. This is because the static method can find a stream of content only in the ideal environment where random errors rarely occur. In the case that random errors exist, since the distortion of the user-side traffic patterns make the peak in values of the similarity comparatively small, it is difficult for the static method to correctly decide that such distorted patterns are parts of the content stream. On the other hand, the proposed method achieves a stable performance. The precision, the recall, and the f-measure of the proposed method have large values for any error rate. This is because the threshold is dynamically calculated based on the statistical information of the relations.

With the knowledge of the estimated value of the random error rate, we can previously define the determination threshold. But the condition of the network generally changes momentarily. Thus, according to Fig. 17(c), we concluded that the proposed method with the dynamic determination threshold has a superior performance, especially in an environment that includes wireless networks, since there is no need to adjust the determination threshold for the changing random error rate.

C. Simulation #2: Verification of the Influence of Background Traffic

1) *Objective of the Simulation:* The proposed method implements packet filters about IP addresses and packet types, so that large amounts of traffic except for the content stream are not included in the traffic pattern. Moreover, even if a small

TABLE III
PARAMETERS OF THE SIMULATION #1

Parameter	Value
Length of the server-side traffic pattern	3000 [sec]
Length of the user-side traffic pattern	60 [sec]
Length of the time-slot	0.2 [sec]
Threshold about the packet size, T_P	100 [kbyte]
Bit-rate of the content	250 [kbps] (approx.)
Bandwidth of links	100 [Mbps]

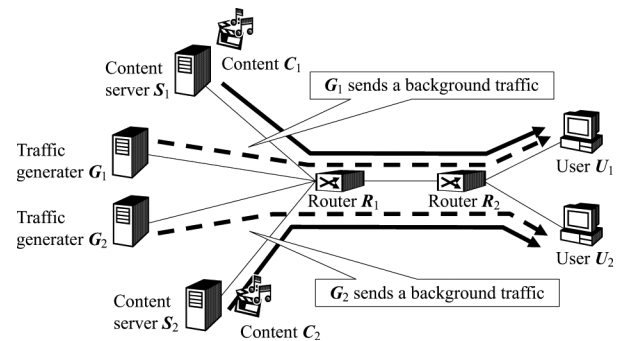


Fig. 18. Topology of the simulation #2.

background traffic is mixed in the user-side traffic, the proposed method correctly finds the content stream by using the similarity and dynamic threshold. But, in this paper, supposing the case that the packets for controlling the delivery and other applications' traffic are involved in the user-side traffic, we implemented simulations to verify the performance of the proposed method in the environment influenced by the background traffic.

2) *Simulation Setup:* Fig. 18 shows the topology of this simulation. We use a simple dumbbell-type network like Section V-B2 to verify only the influence of random errors. As shown in Fig. 18, the content servers S_1 and S_2 deliver the content C_1 and C_2 , respectively. The traffic generators G_1 and G_2 send pareto ON-OFF model traffic as background traffic.

The pareto ON-OFF model is generally used to represent the model of Internet traffic. In the pareto ON-OFF model, a certain rate of traffic flows during the "ON time." But, during the "OFF time," no traffic flows. The distribution of ON-OFF time and the traffic amount in ON time are defined by the following probability density function:

$$p(x) = \alpha k^\alpha x^{-(\alpha+1)}, \quad \alpha, k > 0, x \geq k. \quad (8)$$

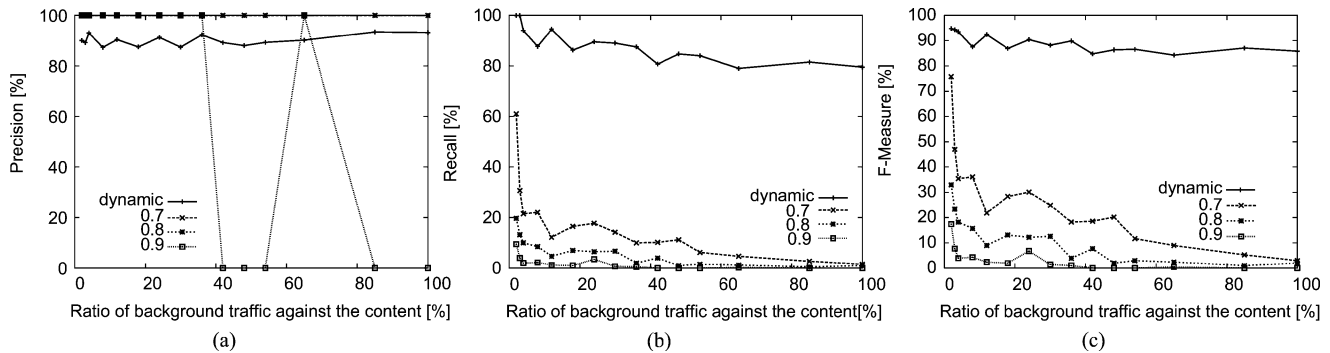


Fig. 19. Results of the simulation #2; (a) precision, (b) recall, and (c) f-measure.

TABLE IV
PARAMETERS OF THE SIMULATION #2

Parameter	Value
Length of the server-side traffic pattern	1080 [sec]
Length of the user-side traffic pattern	60 [sec]
Length of the time-slot	0.2 [sec]
Threshold about the packet size, T_P	100 [kbytes]
Bit-rate of the content	250 [kbps] (approx.)
Pareto parameter, α	1.2
Bandwidth of links	100 [Mbps]

Here, α is a parameter which decides the shape of the hem of the distribution and k is a coefficient which changes according to the traffic size. Since $\alpha = 1.2$ is typically used to represent Internet traffic, we also use this value in this simulation [36].

In this simulation, we conducted experiments with changing the rate of traffic during ON time from 0% to 100% of the content traffic's rate. For instance, 0% means half of the bit rate of the content stream. Here, such a large amount of background traffic does not actually exist, because of packet filtering. However, we conducted this experiment to verify the performance in severe cases. Experiments were conducted 200 times for a certain rate. At this time, the time to start an observation of the user-side traffic amount was changed not to use a particular pattern. Additionally, like Section V-B2, simulations with fixed discrimination thresholds were conducted for comparison. The parameters are shown in Table IV.

3) *Simulation Result*: Figs. 19(a)–(c), respectively, show the results of the simulation about precision, the recall, and the f-measure. As shown in Fig. 19(a), the difference between the maximum and minimum values in the results of the static method is significantly large. The static method is considered to lack stability. This is because the static method can correctly determine only the case in which the peak of similarity graph is especially large and is easily detected. Briefly, the results of the static method are correct for high rates, but it misses most of the user traffic that must be actually detected. If we obtain a high recall with the static method, the precision is sacrificed, and vice versa. On the other hand, the proposed method has a stable performance, according to Fig. 19(a) and (b). Moreover, the recall of the proposed method is much more higher than that of the static method. Consequently, as shown in Fig. 19(c), the performance of the proposed method is superior to that of the static one.

There are two reasons why the static method cannot perform effectively. First, the background traffic is added to the

user-side traffic amount, so that the user-side traffic pattern is transformed. This is the same phenomenon as with random errors. Another important factor is the jitter or fluctuations in the delay. For example, when the processing of a certain packet gets delayed, the amount of the time-slot in which the packet is actually involved decreases.

The proposed method correctly finds out the content stream, even if the user-side traffic pattern deteriorates because of the fluctuation of the delay. This is because the proposed method dynamically defines the determination threshold based on the statistical analysis of the similarity.

D. Simulation #3: Verification of the Generality Concerning the Contents

1) *Objective of the Simulation*: The proposed method makes use of the traffic pattern, which is unique to each piece of content. However, actually, detection faults should be taken into account, because the part of the traffic pattern of a certain content may be similar to the parts of another content. There is no evidence that the traffic pattern is completely unique to content even if it is divided into small parts. Alternatively, in this paper, we conduct a simulation using a variety of contents to verify the performance of the proposed method. In the simulation, the contents are compared with each other and the performance indices are calculated.

2) *Simulation Setup*: The topology, the parameter, and the procedure are almost the same as the simulation in Section V-B, except for random errors. We do not have any error in this simulation. The contents used in the experiment are chosen from 24 example items for each experiment. These consist of pseudomovie data used in NS-2, and were made from news, movies, and other sources.

3) *Simulation Result*: Fig. 20 shows the f-measure for this simulation. The horizontal and vertical axes mean the pair of contents and the f-measure, respectively. There is a pair whose value is smaller than 80%, but the average of the values is about 93%.

The overall traffic pattern of the content stream exhibits dissimilar shapes compared with other traffic patterns in a unique way. This is due to the fact that the traffic pattern represents the volume of the traffic. It is, however, important to note that there could be a partially similar segment in the traffic pattern with other traffic patterns, which would cause a false positive detection of the content stream. This is, however, very rare but it will deteriorate the precision.

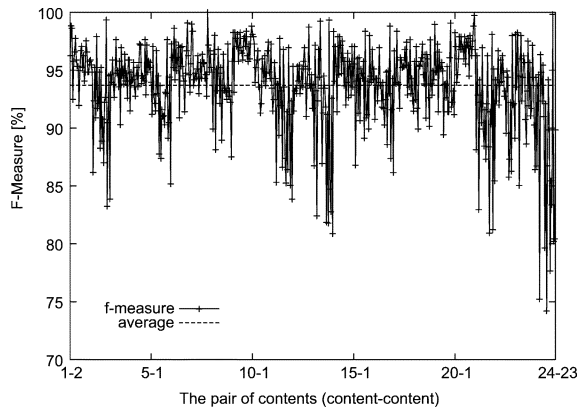


Fig. 20. Result of the simulation #3 (f-measure).

For further improving the accuracy of the proposed method, researchers in this domain, in future, may take into consideration the way to generate traffic patterns which can distinguish contents more clearly. We can think the following four approaches: 1) using multiple picked patterns from the user-side, 2) automatically adjusting the period of a time slot, 3) absorbing the delay jitter, and 4) shaping the traffic patterns according to network bandwidth.

VI. CONCLUSION

Thanks to broadband technology, streaming technology is used in many kinds of applications. However, a control method for the steaming content delivery is required to prevent abuse of the content. Traitor-tracing technology is one of these technologies and is used to observe the usage of content. General tracing methods have limitations, because of a high load to produce many contents and the watermark's limitation. To advance the security of content delivery, we proposed a method, which takes advantage of the traffic pattern. To evaluate the performance of the proposed method, simulations and actual experiments were conducted. Finally, we obtained satisfactory results, verifying the effectiveness of our approach.

As discussed in Section II, the proposed method can also be used with previous DRM technologies. Moreover, the calculation cost of the proposed method is less than those of encoding and watermarking schemes used in general traitor-tracing techniques. We believe that the combined usage of the proposed technique and the conventional traitor-tracing techniques can achieve higher efficiency and accuracy than the stand-alone usage. In the combined usage, one can first use the proposed method to detect the possibility of an unauthorized content use. Next, the conventional traitor-tracing method is applied which requires calculation capability to conduct precise investigations.

Indeed, we have to figure out how to protect the confidential information from being accessed by attackers. From a commonsense perspective, we must deploy security features such as encryption, access control, and so forth. Of course, using watermarking schemes for streaming contents is one of the most popular ideas of implementing security. In addition, packet monitoring for the communication that takes place between routers may also be a practical solution. Our proposed method falls into this second category. If there are attackers who have exploited knowledge of our system, they may resort to changing

the bit-rate of the content. As shown in Section IV-G, the experimental results demonstrate that our scheme does not depend on bit-rates. Furthermore, the most effective way an attacker can manipulate against our system is that the VBR content is changed into a constant bit-rate stream. This is analogous to copying the content into a USB memory and bringing it out from the server illegally. Indeed, this is a focus of physical intrusion detection mechanisms and we are, unfortunately, unable to solve this problem by using network technologies.

The future work is to advance the determination accuracy. The segmentation method of packets and the measurement of similarity of traffic patterns may be further improved.

REFERENCES

- [1] Z. Yang, H. Ma, and J. Zhang, "A dynamic scalable service model for sip-based video conference," in *Proc. Ninth Int. Conf. Computer Supported Cooperative Work in Design*, May 2005, vol. 1, pp. 24–26.
- [2] M. Shimakawa, D. P. Holed, and F. A. Tobagi, "Video-conferencing and data traffic over an ieee 802.11g wlan using dcf and edca," in *Proc. Int. Conf. Communications (ICC)*, May 2005, vol. 2, pp. 16–20.
- [3] R. H. Koenen, J. Lacy, M. Mackay, and S. Mitchell, "The long march to interoperable digital rights management," *Proc. IEEE*, vol. 92, pp. 883–897, 2004.
- [4] A. M. Eskicioglu, J. Town, and E. J. Delp, "Security of digital entertainment content from creation to consumption," *Special Issue on Image Security, Signal Process. Image Commun.*, vol. 18, no. 4, pp. 237–262, Apr. 2003.
- [5] A. M. Eskicioglu and C. Brooklyn, "Protecting intellectual property in digital multimedia networks," *Computer Publication*, vol. 36, no. 7, pp. 39–45, Jul. 2003.
- [6] E. I. Lin, A. M. Eskicioglu, R. L. Lagendijk, and E. J. Delp, "Advances in digital video content protection," *Proc. IEEE*, vol. 93, no. 1, pp. 171–183, Jan. 2005.
- [7] Y. Nishimoto, A. Baba, T. Kurioka, and S. Namba, "A digital rights management system for digital broadcasting based on home servers," *IEEE Trans. Broadcast.*, vol. 52, no. 2, pp. 167–172, Jun. 2006.
- [8] F. Hartung and F. Ramme, "Digital rights management and watermarking of multimedia content for m-commerce applications," *IEEE Commun. Mag.*, vol. 38, no. 11, pp. 78–84, Nov. 2000.
- [9] W. Luh and D. Kundur, "New paradigms for effective multicasting and fingerprinting of entertainment media," *IEEE Commun. Mag.*, vol. 43, no. 6, pp. 77–84, Jun. 2005.
- [10] D. Boneh and M. Franklin, "An efficient public key traitor tracing scheme," in *Advances in Cryptology (Crypto'99)*, 1999, pp. 338–353.
- [11] B. Chor, A. Fiat, M. Naor, and B. Pinkas, "Tracing traitors," *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 893–910, May 2000.
- [12] A. Fiat and T. Tassa, "Dynamic traitor tracing," *J. Cryptology*, vol. 14, no. 3, pp. 211–223, 2001.
- [13] R. S. Naini and Y. Wang, "Sequential traitor tracing," *IEEE Trans. Inf. Theory*, vol. 49, no. 5, pp. 1319–1326, May 2003.
- [14] C. I. Podilchuk and E. J. Delp, "Digital watermarking: Algorithms and applications," *IEEE Signal Process. Mag.*, vol. 18, no. 4, pp. 33–46, Jul. 2001.
- [15] D. Kundur and K. Karthik, "Video fingerprinting and encryption principles for digital rights management," *Proc. IEEE*, vol. 92, no. 6, pp. 918–932, Jun. 2004.
- [16] B. Chen and G. W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1423–1443, May 2001.
- [17] S. Craver, N. Memon, B. L. Yeo, and M. M. Yeung, "Resolving rightful ownerships with invisible watermarking techniques: Limitations, attacks, and implications," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 573–586, May 1998.
- [18] M. Barni and F. Bartolini, "Data hiding for fighting piracy," *IEEE Signal Process. Mag.*, vol. 21, no. 2, pp. 28–39, Mar. 2004.
- [19] K. Su, D. Kundur, and D. Hatzinakos, "Statistical invisibility for collusion-resistant digital video watermarking," *IEEE Trans. Multimedia*, vol. 7, no. 1, pp. 43–51, Feb. 2005.
- [20] M. Kutter, S. Voloshynovskiy, and A. Herrigel, "The watermark copy attack," in *Proc. Electronic Imaging '99, Security and Watermarking of Multimedia Contents II*, San Jose, Jan. 2000, vol. 3971, pp. 371–380.

- [21] M. Dobashi, H. Nakayama, N. Kato, Y. Nemoto, and A. Jamalipour, "Traitor tracing technology of streaming contents delivery using traffic pattern in wired/wireless environments," in *Proc. IEEE GLOBECOM*, Nov. 2006, pp. 1–5.
- [22] B. N. Park, W. Lee, and J. W. Kim, "A license management protocol for protecting user privacy and digital contents in digital rights management systems," *IEICE Trans. Inf. Syst.*, vol. E88-D, no. 8, pp. 1958–1965, Aug. 2005.
- [23] Y. Matias and A. Shamir, "A video scrambling technique based on space filling curves," in *Proc. Advances in Cryptology (CRYPTO'87)*, 1988, vol. 293, pp. 398–417.
- [24] H. K. Chang and J.-L. Liu, "A linear quadtree compression scheme for image encryption," *Signal Process. Image Commun.*, vol. 10, no. 4, pp. 279–290, Sep. 1997.
- [25] H. Cheng and X. Li, "Partial encryption of compressed images and videos," *IEEE Trans. Signal Process.*, vol. 48, no. 8, pp. 2439–2451, Aug. 2000.
- [26] A. Pommer and A. Uhl, "Selective encryption of wavelet-packet encoded image data: Efficiency and security," *Multimedia Syst.*, vol. 9, no. 3, pp. 279–287, 2003.
- [27] X. Wang, "Mpeg-21 rights expression language: Enabling interoperable digital rights management," *IEEE Multimedia*, vol. 11, no. 4, pp. 84–87, Oct./Dec. 2004.
- [28] I. J. Cox, M. L. Miller, and A. L. McKellips, "Watermarking as communications with side information," *Proc. IEEE*, vol. 87, pp. 1127–1141, Jul. 1999.
- [29] C. E. Shannon, "Channels with side information at the transmitter," *IBM J. Res. Develop.*, vol. 2, pp. 289–293, 1958.
- [30] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 1897–1905, Sep. 1998.
- [31] W. Trappe, M. Wu, Z. J. Wang, and K. J. R. Liu, "Anti-collusion fingerprinting for multimedia," *IEEE Trans. Signal Process.*, vol. 51, no. 4, pp. 1069–1087, Apr. 2003.
- [32] R. J. Anderson and F. A. P. Petitcolas, "On the limits of steganography," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 474–481, May 1998.
- [33] S. Voloshynovskiy, S. Pereira, V. Iquise, and T. Pun, "Attack modeling: Towards a second generation watermarking benchmark," *Signal Process. Elsevier*, vol. 81, no. 6, pp. 1177–1214, Jun. 2001.
- [34] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York: Wiley Interscience, 2000.
- [35] VINTproject, Network simulator 2 [Online]. Available: <http://www.isi.edu/nsnam/ns/>
- [36] V. Paxson and S. Floyd, "Wide area traffic: The failure of poisson modeling," *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 226–244, Jun. 1995.



Hidehisa Nakayama (M'06) received the B.E., M.S., and Ph.D. degrees in information sciences from Tohoku University in 2000, 2002, and 2005, respectively.

He is a Senior Assistant Professor and currently working for Tohoku Institute of Technology, Sendai-shi, Japan. He has been engaged in research on intelligent sensor technology, wireless mobile ad hoc network, computer networking, character string analysis, pattern recognition, and image processing.

Dr. Nakayama is a member of the IEEE Communications Society, the Institute of Electronics, Information and Communication Engineers (IEICE), and the Information Processing Society of Japan (IPSJ). He received the Paper Award for Young Researcher of the IPSJ Tohoku Chapter in 2000, the Best Paper of Pattern Recognition Award in SCI 2003, and the eighth Network System Award of the IEICE Technical Committee on Network Systems in 2009.



Abbas Jamalipour (S'86–M'96–SM'00–F'07) received the Ph.D. degree from Nagoya University, Japan.

He is the author five books, nine book chapters, and over 200 technical papers, in the field of mobile communications. He disseminated fundamental concepts of the next-generation mobile networks and broadband convergence networks; some are being gradually deployed by industry and adopted by ITU-T. He is currently leading the Wireless Networking Group (WiNG) at the University of Sydney, Australia, and

has an Adjunct Professorship position at Macquarie University.

Dr. Jamalipour is a Fellow of IEAust, an IEEE Distinguished Lecturer and a Technical Editor of several scholarly journals including *IEEE Communications*, *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY*, *IEEE Communications Surveys & Tutorials*, *ETRI Journal*, etc. He was the Editor-in-Chief of the *IEEE Wireless Communications* from 2006 to 2008. He has been an organizer or chair in many international conferences and currently is the General Chair of the IEEE Wireless Communications and Networking Conference (IEEE WCNC2010). He is a voting member of Conference Boards, Education Board, and the Online Contents of the IEEE Communications Society. He has been a reviewer for several international research bodies including the Australian Research Council, NSERC (Canada), NSF (USA), European Science Foundation, Science Foundation (Ireland), and the Kentucky Science and Engineering Foundation's R&D Excellence Program (USA).



Nei Kato (M'03–A'04–SM'05) received the M.S. and Ph.D. degrees in information engineering from Tohoku University, Japan, in 1988 and 1991, respectively.

He joined the Computer Center of Tohoku University, Sendai-shi, Japan, in 1991, and has been a full professor in the Graduate School of Information Sciences since 2003. He has been engaged in research on computer networking, wireless mobile communications, network security, image processing, and neural networks. He has published more than 180 papers in

journals and peer-reviewed conference proceedings.

Dr. Kato currently serves as vice chair of IEICE Satellite Communications TC, secretary of IEEE Ad Hoc & Sensor Networks TC, a technical editor of *IEEE Wireless Communications* (2006–), an editor of *IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS* (2008–), and an associate editor of *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY* (2009–). He served as a co-guest-editor for the *Special Issue on Wireless Communications for E-Healthcare*, *IEEE Wireless Communications Magazine*, a workshop cochair of VTC2010-Fall, a symposium cochair of GLOBECOM'07, ICC'10, ChinaCom'08, ChinaCom'09, and WCNC2010 TPC vice Chair. His awards include the Minoru Ishida Foundation Research Encouragement Prize (2003), the Distinguished Contributions to Satellite Communications Award from the IEEE Communications Society, the Satellite and Space Communications Technical Committee (2005), the FUNAI Information Science Award (2007), the TELCOM System Technology Award from Foundation for Electrical Communications Diffusion (2008), and the IEICE Network System Research Award (2009). Besides his academic activities, he also serves as member of the expert committee of Telecommunications Council, Telecommunications Business Dispute Settlement Commission Special Commissioner, Ministry of Internal Affairs and Communications, Japan, and as the chairperson of ITU-R SG4, Japan. He is a member of the Institute of Electronics and Information and Communication Engineers (IEICE).