Traitor-Tracing in Online Content Streaming

André Fagerberg
School of Information Technology
Griffith University, Gold Coast Campus, QLD 9726, Australia
andre.fagerberg@griffithuni.edu.au

***Abstract*—In recent years, due to vastly evolving broadband technologies where transmission speeds and applications have completely changed the way we access content online. Studies have shown that multimedia content is accessed in majority through online streaming applications instead of downloading the specific files. This phenomenon poses additional threats of digital rights management (DRM) technologies due to traditional watermarking methods are no longer viable in order to control content spreading and avoid unauthorised content use. General traitor tracing techniques make use of watermarking content with a unique identifier for each user; these methods are not practical in real-time streaming applications. This paper makes use of an existing solution to traitor tracing which monitors the content stream using traffic patterns constructed from routers in a fixed LAN network where monitoring is allowed. A proposed solution is provided which describes how this method can be applied in combination with other network technologies such as mobile agents to expand the system for use in large-scaled online streaming applications.**

***Index Terms*—traitor-tracing, piracy, watermarking, mobile agents, online streaming, DRM.**

I. Introduction

In todays vastly evolving broadband technology, data rates and bandwidths are at a point where streaming content in high quality while using other online applications simultaneously is a reality. This leads to a much more frequent use of streaming applications, rather than downloading the content itself; streaming increases the efficiency of accessing useful information in any networking environment. However, with such possibilities, there are several threatening security issues, such as content usage without authorisation by content holders [1]. For instance, content can be intended for certain users while there is no guarantee that retransmissions or unauthorised use of the content will not occur. Traditional countermeasures include digital rights management (DRM) technologies to securely encrypt and deliver content to intended users; although content providers may implement active management technologies to monitor content usage, there is no way of restraining distribution of decrypted contents [1]. This brings us to the main issue discussed in this paper, which is: how do we trace re-distribution of content, and how do we identify 'traitors' who share or misuse content without the knowledge of the content holder.

Common traitor-tracing techniques involve the process of digital watermarking content by assigning each with its own identity [2]. This method requires every user and content to be uniquely identified prior to distribution, which is not practical as it requires a lot of computations to encode, therefore traditional watermarking methods are not viable in streaming applications where users consume large amount of content [1], [2].

The objective of this paper, while combining several networking techniques is to provide a theoretical solution of an effective traitor-tracing method for tracing multimedia used in online streaming applications.

The paper is organised as follows. Section II introduces a traitor-tracing system that takes use of traffic patterns to monitor content within limited networks where monitoring is allowed; additionally, a brief overview of digital watermarks and mobile agents is provided. Section III defines the problem with current traitor-tracing techniques; section IV provides a detailed description of the proposed method in [1], and finally section V suggests solutions to how this method can be combined with other networking technologies such as mobile agents to be effective in larger scaled streaming applications.

II. Literature Review

A brief description of the proposed traitor-tracing technique is presented in this section as well as an overview of digital watermarking and mobile agents.

A. Traitor Tracing Technique Using Traffic Pattern

A detailed description of a traitor-tracing method can be found in [1]. This method of traitor-tracing is a rather unique technique that monitors and collects samples of traffic patterns during the process of streaming content. The proposed scheme is displayed in Fig. 1. A content server $\boldsymbol{S}$ is responsible for storing and distributing content $\boldsymbol{C}$. Routers $\boldsymbol{R_1}$, $\boldsymbol{R_3}$ and $\boldsymbol{R_4}$ monitors the traffic over the network. For instance, router $\boldsymbol{R_1}$ will start monitoring the traffic flow once a user is streaming some content, it then adds up the sizes of each packet during the stream. This summation takes use of packet filtering from IP addresses to establish the content server's flow. Once the content server has obtained the required information it is sent to the management server $\boldsymbol{M}$ using protocols such as Simple Network Management Protocol (SNMP) and Internet Control Message Protocol (ICMP). The authors of [1] argues that in cases where a user is simultaneously watching two different

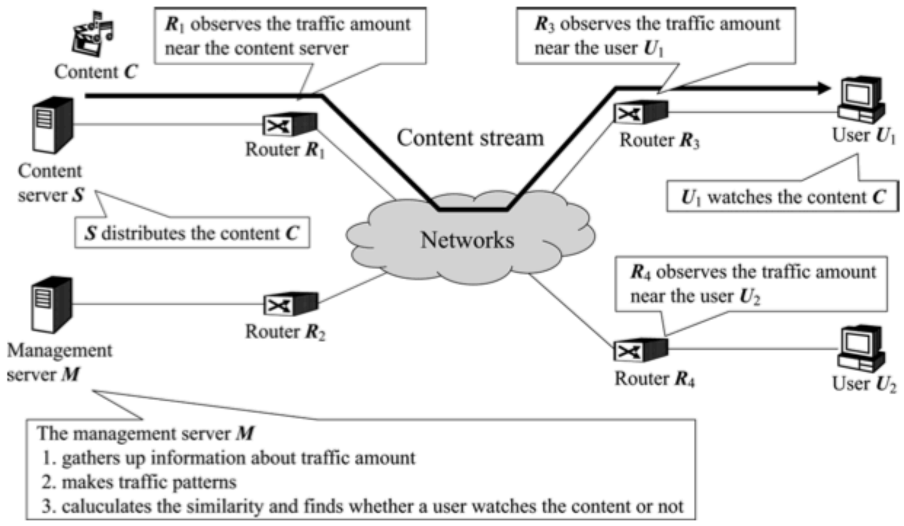
Content C
R_1 observes the traffic amount near the content server
R_3 observes the traffic amount near the user U_1
Content server S
Router R_1
Content stream
Router R_3
User U_1
U_1 watches the content C
S distributes the content C
Networks
R_4 observes the traffic amount near the user U_2
Router R_2
Management server M
Router R_4
User U_2
The management server M
1. gathers up information about traffic amount
2. makes traffic patterns
3. caluculates the similarity and finds whether a user watches the content or not

Fig. 1. Outline of the proposed scheme [1].

contents, the system is able to differentiate the two by inspecting the separate port numbers, since streaming applications occupy different port numbers for transmission.

Thereafter, the management server $\boldsymbol{M}$ is responsible for constructing traffic patterns at both ends of the transmission i.e. both the user side through routers $\boldsymbol{R_3}$, $\boldsymbol{R_4}$and on the server side through router $\boldsymbol{R_1}$'s packet information. These traffic patterns are constructed during the time of content $\boldsymbol{C}$ being delivered. The management server $\boldsymbol{M}$ then compares the two traffic patterns at both ends to determine which user is viewing the content, in other words, if user $\boldsymbol{U_1}$ is watching content $\boldsymbol{C}$ then $\boldsymbol{U_1}$**'s** traffic pattern is matching the pattern obtained at $\boldsymbol{R_1}$ during the delivery of content $\boldsymbol{C}$ [1]**.** With this data, the system is able to determine that $\boldsymbol{U_1}$ receives $\boldsymbol{C}$, and it can also detect if $\boldsymbol{U_1}$ retransmits the stream of content to user $\boldsymbol{U_2}$ because traffic patterns will indicate that the same content stream $\boldsymbol{C}$ traverses the router [1].

Section II-B provides a brief overview of digital watermarking.

B. Digital Watermarking

Digital watermarking is the process of embedding a hidden message related to multimedia content such as images, videos, and songs. The technique is closely related to steganography, which is used as a cover to hide the contents existence. Digital watermarking was developed during the past 15 years and it is now used for several different applications in traitor tracing [3]. Due to the increase in illegal distribution of multimedia content and the importance of digital rights management (DRM), several watermarking techniques have been established in an attempt to counter such activities [2], [3]. One of the first watermarking applications was used for TV broadcast monitoring, which allowed content providers to track when a specific program was being broadcast.

A very important factor is owner identification. Being able to identify the owner of a digital content such as video or image is a crucial task in cases related to copyright infringement. Therefore, instead of appending copyright notices within each content, watermarking can be used to embed copyright in the content itself.

The most relevant property of watermarking for this paper is the transaction tracking technique. The watermark embedded in digital content can be used to record how many transactions of a specific content have been taken place in the history of its existence [2], [3]. For example, digital watermarking can be used to record the identity of every legal recipient's copy of a movie by embedding a unique watermark within each copy. If the movie is then redistributed to a third party whom does not have permission to view the movie, the movie producers can then identify which recipient is the traitor.

Section II-C provides a brief overview of mobile agents.

C. Mobile Agents

Mobile agents are applications that can migrate between machines in heterogeneous networks; they can do so on their own at any given time due to their autonomous property. They are used for implementation of distributed applications in both wired and wireless networks [4]. The use of mobile agents bring some powerful advantages over traditional client/server configurations, they can significantly reduce the workload and latency of networks, provide greater scalability, and provide further flexibility in the development and maintenance of applications. The most prominent difference is that mobile agent systems can move to the location of where the data source resides and use local method calls, whereas in traditional systems data must be transported to the running process through remote procedure calls, which is a much slower process [4], [5].

The agent's ability to sense their environments and react dynamically to changes is a useful property in many intrusion detection systems [6].

Java-based mobile software agents can be used to monitor multiple levels of networked computers including: traffic packets, processes, system, and user activities. These types of agents can detect changes in the patterns of networks, and provide detailed information of packets entering and leaving a network [6].

Mobile agents are mostly independent of computer systems and the transport layer; therefore they have the potential to provide system integration with optimal conditions, as network computing is profoundly heterogeneous [4].

Section III contains a critical analysis of the issues in current traitor-tracing systems [1].

III. Problem Statement

The problem with current traitor-tracing techniques is that there are no efficient ways of tracing multimedia content used in online streaming applications. There exist schemes that are effective for individual redistribution of content that contains watermarks, where the content owner can trace their copy and find the traitor who shared it. However, there are ways where several users may collude against watermarking techniques by distorting the watermark through combining multiple copies of the content into one; in doing so, there is no way of detecting which user the watermark belongs to [1], [7]. Furthermore, there are limitations in the watermarking procedure due to the high load of encoding many contents; therefore traditional watermarking techniques are inefficient for applications that use streaming technology [2].

The proposed solution discussed in [1] introduces a technique that can be used to monitor traffic patterns in an attempt to more efficiently detect and trace traitors. The main problem with the proposed method is that the technique can only be deployed in limited networks where monitoring routers and their nodes are allowed. The system is most likely only suited for corporate LANs rather than large scaled online streaming applications such as YouTube and Vimeo. The authors of [1] suggests that their system is appropriate for several streaming applications, such as Helix and QuickTime servers offered by Real Networks and Apple.

For this method to be feasible outside of limited networks, one major issue must be addressed regarding the monitoring of packets at the router level, i.e. how do we monitor a users router at a remote location if there is no software installed on the user-side to allow this. Especially since the proposed system is dependant on comparing traffic patterns constructed by the sum of packets passing through users routers. The system needs to be combined with other technologies to allow monitoring at the user-side, for example, the use of mobile agents.

Section IV provides a more detailed description of an existing solution to traitor-tracing in limited networks.

IV. Existing Solution

This existing solution is focused around variable bitrate (VBR) streaming content to collect traffic patterns in order to identify traitors in a fixed LAN environment where monitoring is allowed. A full description of the method can be found in [1].

A. Defining the Traffic Pattern

In order to collect and compare the traffic pattern at the server side router $\boldsymbol{S}$ and the user side router $\boldsymbol{U}$, a time slot $\boldsymbol{t}$ must first be established so that the system knows when to start collecting data [1].

In this method, $\boldsymbol{S}$ monitors and collects the traffic pattern for the full duration of content delivery and its pattern is denoted $\boldsymbol{X_S}$ while $\boldsymbol{U}$'s pattern is expressed as $\boldsymbol{Y_U}$. The observation length of $\boldsymbol{U}$ is shorter than $\boldsymbol{S}$ [1].

Once $\boldsymbol{Y_U}$ has been collected it is used as a partial pattern to be compared with $\boldsymbol{X_S}$, and since $\boldsymbol{X_S}$ is a pattern of the entire piece of content and $\boldsymbol{Y_U}$ is only a partial, it can be placed on top of $\boldsymbol{X_S}$ for comparison by shifting it from left to right until a match in the two patterns is found; when the partial pattern reaches the rightmost side of the server-side pattern, a value of similarity can be obtained ($\boldsymbol{S} - \boldsymbol{U} + 1$). In cases where this value is large indicates that the user-side pattern is very similar to the server-side pattern, which allows the system's management server $\boldsymbol{M}$ to determine that a specific user is watching the content. This scenario is referred to as a "match", and the system relies on a threshold $\boldsymbol{T_R}$ when deciding whether the value is significant enough [1]. A dynamic threshold is therefore implemented, which is discussed in the following section.

B. Dynamic Threshold Detection

In all transmissions of data there exists jitter and loss of packets, such events influence this method of collecting data for constructing traffic patterns. For instance, if packets are lost during transmission of the content, the user-side traffic pattern will fluctuate and the value of similarity becomes smaller. Using a static threshold is not suitable for such systems because depending on the amount of jitter and packet loss during a particular content transmission, the threshold must be able to adapt to such circumstances; therefore a dynamic threshold is required [1].

The majority of similar data analysed by this method is generally very small, but when a similarity occurs between two patterns the value is significantly larger and such value is called an "outlier" [1]. The equation used to calculate the dynamic threshold T_R is defined as follows:

$$T_R = \min\left(\mu_R + 4\sigma_R\right)$$

The values $\boldsymbol{\mu_R}$ and $\boldsymbol{\sigma_R}$ are the mean and the variance of similarity. Every traffic pattern constructed at the user-side has its own $\boldsymbol{\mu_R}$ and $\boldsymbol{\sigma_R}$ values. It is possible to balance the trade-off between false-positives and detection ratios by adjusting the coefficient of $\boldsymbol{\sigma_R}$ [1]. A larger value of this coefficient guarantees a low false-positive ratio but at the cost of a lower detection ratio, and a lower value result in the reverse effect [1]. The probability that data values are greater or equal to $\boldsymbol{T_R}$, and are not outliers, is approximately 6%; these occurrences are classified as false-positive. Similarity values greater than $\boldsymbol{T_R}$ are considered outliers, which signify a correct match of the traffic patterns [as seen in Fig. 2] [1].

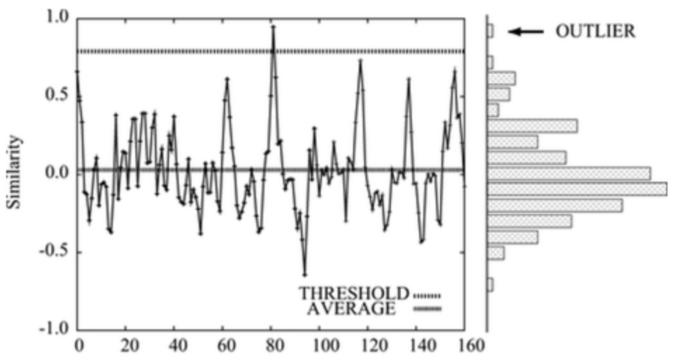

Fig. 2. Similarity graph of matching traffic pattern [1].

This method uses a management server to calculate similarities between traffic patterns; therefore no additional computations are needed on the server providing the content. The total number of multiplications needed to calculate the similarity $\boldsymbol{M}$ is estimate to ($\boldsymbol{S} - \boldsymbol{U} + 1$) x $\boldsymbol{M}$. Where $\boldsymbol{S}$ and $\boldsymbol{U}$ are the full length of the server/user-side traffic patterns. $\boldsymbol{M}$ is a constant that is dependant on $\boldsymbol{U}$. The main advantage of this network topology is that the total cost of these calculations is much less than those of traditional methods including encoding and watermarking [1].

The following section provides a proposed traitor-tracing method that can be used in *any* online content streaming environment.

V. Proposed Solution

The proposed method in this section combines other network technologies to provide an effective solution to online content streaming that can be applied in *any* environment. This method applies the same techniques discussed in section IV and is proposed in [1], which is a method of tracing traitors in a fixed LAN environment where monitoring is allowed. To overcome some of the limitations of their method, this proposed system takes use of mobile agents in a unique manner.

The following section describes the proposed method using mobile agents.

A. Using Mobile Agents

The existing method discussed in section IV is limited to fixed environments where monitoring at nodes is allowed, therefore it is not a scalable solution to traitor-tracing for online applications since there is no way of constructing a traffic pattern at the user-side to be matched with the server-side pattern.

Mobile agents have the ability to travel autonomously across the web [8]; therefore the proposed method can take use of such characteristics in order to collect data at the user-side. When a user requests access to a specific content through an online streaming application, a java-based mobile agent can be assigned to transport itself to that users host-network. The agent's job is then to start gathering packets at the user-side in order to construct a traffic pattern to be compared with the server-side pattern. The use of mobile agents for constructing the traffic pattern can be used to reduce the workload of the management server during heavy traffic due its unique ability to perform computations locally [9]. This will not only prevent the management server from overloading, it will also allow the traffic pattern to be more accurate since the calculations are made on the local host where the content is being streamed. However, there are some limitations; the mobility of an agent may be restricted by hosts with dynamically assigned network addresses, or by a high security configured firewall [8]. We must assume, for theoretical purposes, that hosts in the proposed method do not possess these limitations.

Next is a description of the mobile agents and their requirements in constructing traffic patterns.

1) Agent Requirements

These agents monitoring and calculating the traffic pattern on the user-side takes use of UNIX tools like as 'snoop' which is an inbuilt packet analyser tool in oracle Solaris OS used to gather all incoming packets sent from the content server where the streamed media resides [6]. The agent detects changes in the numbers and sizes of incoming packets while using the same protocols used in the original method [1] (SNMP and ICMP). These intelligent agents must also be programmed to perform the similarity calculation discussed in section IV-B. Once the similarity calculation is complete and an outlier is detected, the resulting traffic pattern is encapsulated and sent back to the management server together with the agent in order to determine whether the user is a 'traitor' or not.

VI. Conclusion

Online streaming applications have changed the way we view content online. However, additional obstacles have emerged with these technologies. Different methods are required to control and prevent abuse of content delivery online. To prevent unauthorised use of content online, traitor-tracing techniques are used, however, traditional methods including watermarking are no longer viable solutions in this new streaming era. This paper describes a new efficient method of traitor-tracing that takes use of traffic patterns collected by routers in a fixed network where monitoring is allowed. In addition, a proposed solution of how this method can be expanded into a large-scale online streaming application is provided. The proposed system combines other network technologies such as intelligent mobile agents to overcome some of the issues posed in the original method. However, there are some limitations regarding the mobility of mobile agents as discussed in section V-A where hosts that have dynamically assigned addresses or configured firewalls may prevent the system from functioning.

Further research must be conducted concerning the mobility of mobile agents and their ability to gain access to requesting hosts in order to collect and calculate traffic patterns that can ultimately be used to prevent misuse of multimedia content in online streaming applications.

References

[1] H. Nakayama, A. Jamalipour and N. Kato, 'Network-based traitor-tracing technique using traffic pattern', Information Forensics and Security, IEEE Transactions on, vol 5, iss 2, pp. 300--313, 2010.

[2] D. Jarnikov, E. Hietbrink, M. Arana and J. Doumene, 'A watermarking system for adaptive streaming', pp. 375--377, 2014.

[3] M. Barni, T. Kalker and H. Kim, 'Digital watermarking', 4th ed. Siena: Springer, 2005.

[4] M. Eid, H. Artail, A. Kayssi and A. Chehab, 'Trends in mobile agent applications', Journal of Research and Practice in Information Technology, vol 37, iss 4, pp. 323--351, 2005.

[5] P. Rajguru and S. Deshmukh, 'Analysis of Mobile Agent', Journal of Global Research in Computer Science, vol 2, iss 11, pp. 6--10, 2011.

[6] D. Dasgupta and H. Brian, 'Mobile security agents for network traffic analysis', vol 2, pp. 332--340, 2001.

[7] T. Laarhoven, J. Doumen, P. Roelse, B. Skoric and B. de Weger, 'Dynamic Tardos traitor tracing schemes', Information Theory, IEEE Transactions on, vol 59, iss 7, pp. 4230--4242, 2013.

[8] Y. Aridor and M. Oshima, ‘Infrastructure for mobile agents: requirements and design’, IBM Research, Tokyo Research Laboratory, pp. 38--49, 1998.

[9] D. Lange and M. Oshima, 'Seven good reasons for mobile agents', Commun. ACM, vol. 42, no. 3, pp. 88--89, 1999.