Towards Streaming Media Traffic Monitoring and Analysis

Hun-Jeong Kang, Hong-Taek Ju, Myung-Sup Kim and James W. Hong

DP&NM Lab.
Dept. of Computer Science and Engineering
POSTECH, Pohang Korea
Email: {bluewind, juht, mount, jwkhong}@postech.ac.kr
http://dpnm.postech.ac.kr/

Abstract

Recently, a significant increase in streaming media traffic has occurred. This trend is fueled by a large number of Internet users, high bandwidth connections, and improved performance in PCs. It is important to monitor and analyze streaming media traffic. But most enterprises and ISPs cannot obtain exact information about streaming media traffic. Because the majority of streaming media applications use port numbers dynamically allocated, the existing method, which uses well-known port numbers for identifying application of traffic, cannot precisely analyze streaming media traffic. To solve the problem, this paper presents a method and system design for monitoring and analyzing streaming media traffic over RTSP and MMS. Our approach analyzes streaming control messages and extracts information for transferring streaming media data. This information is applied to identifying streaming media traffic. With this approach, we can analyze streaming media traffic and provide information for understanding the behavior of networks and network planning.

Introduction

- Needs of monitoring streaming media traffic
  - ✓ A significant increase of streaming media traffic has occurred
  - ✓ It is essential to monitor and analyze streaming media traffic, for understanding the behavior of networks and network planning
- Difficulty in analysis of streaming media traffic
  - ✓ Streaming applications use dynamically allocated port numbers
  - ✓ Protocols are various, and some of them are proprietary
- Our approach is
  - ✓ to capture the packet over the streaming control protocol
  - ✓ to extract port numbers and protocol used for transferring streaming media data from the payload
  - ✓ to identify streaming media traffic and non-streaming traffic

Streaming Media Traffic Monitoring & Analysis

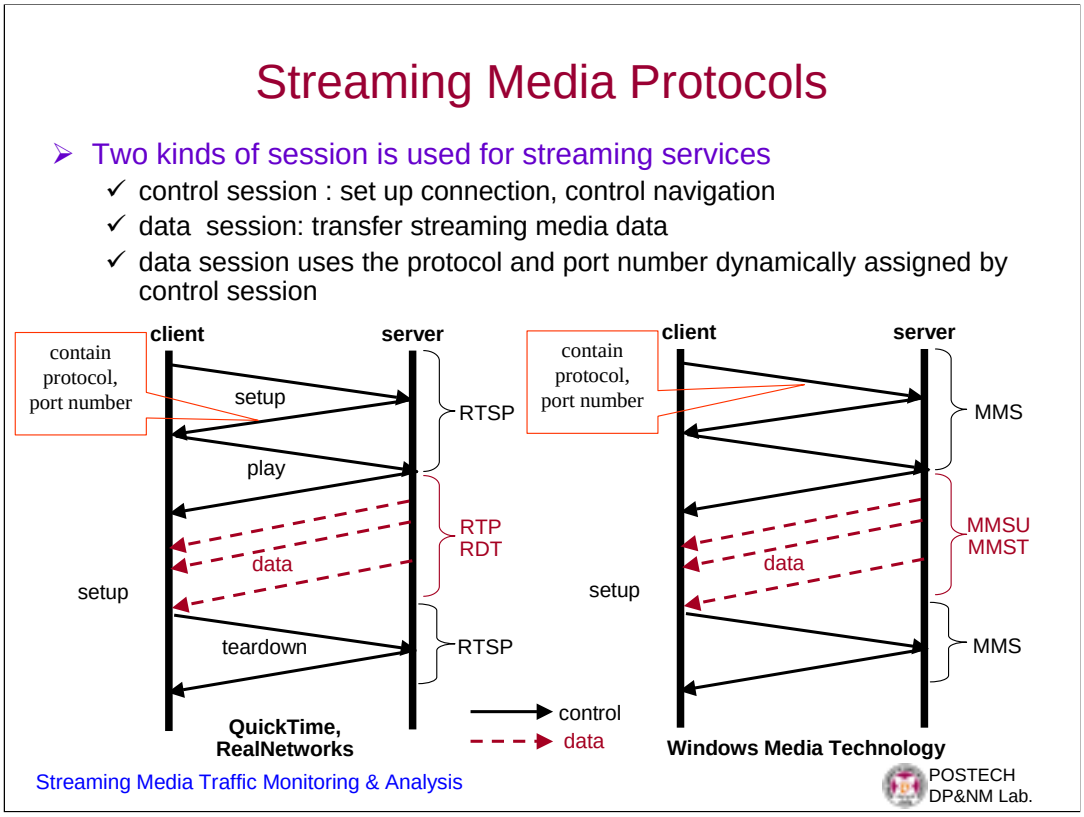
POSTECH
DP&NM Lab.

Recently, a significant increase in streaming media traffic has occurred. Many sites provide streaming contents of broadcast, movies, and music. This streaming media traffic will be more and more dominant in IP networks, due to a large number of users on the Internet, high bandwidth connections, and improved performance in PCs. Therefore, for acquiring information about the status of networks, it is important to monitor and analyze streaming media traffic.

But most enterprises and ISPs cannot obtain exact information about streaming media traffic. The majority of existing traffic monitoring systems use well-known port numbers for identifying the application of traffic. This method, however, cannot determine the application of streaming media traffic, because most streaming media applications make use of port numbers which are not well-known but dynamically allocated. As a result, traffic to transfer streaming media is misidentified as unknown traffic in these systems [10, 13].

Another difficulty in analyzing streaming media traffic is that there are different standard streaming service platforms. Streaming protocols vary according to streaming service platform, and most of these protocols are proprietary. As a result, the analysis of streaming media traffic is dependent on a streaming protocol of which specification may not be public. These diversity and closeness of protocols make it difficult to monitor and analyze streaming traffic.

To solve this problem, this paper presents a method and system design for monitoring and analyzing streaming media traffic, including Windows Media Technology [4], RealNetworks [5], and QuickTime [6] traffic. We have a streaming analyzer which parses streaming control protocol such as RTSP [1] and MMS [4]. It analyzes the payload of the packet associated with streaming control protocol and extracts information about the port numbers and protocol used for transferring streaming media data. This information is applied to determine whether unknown traffic is streaming traffic or not. With this approach, we can analyze streaming media traffic and provide information for understanding the behavior of networks and network planning.



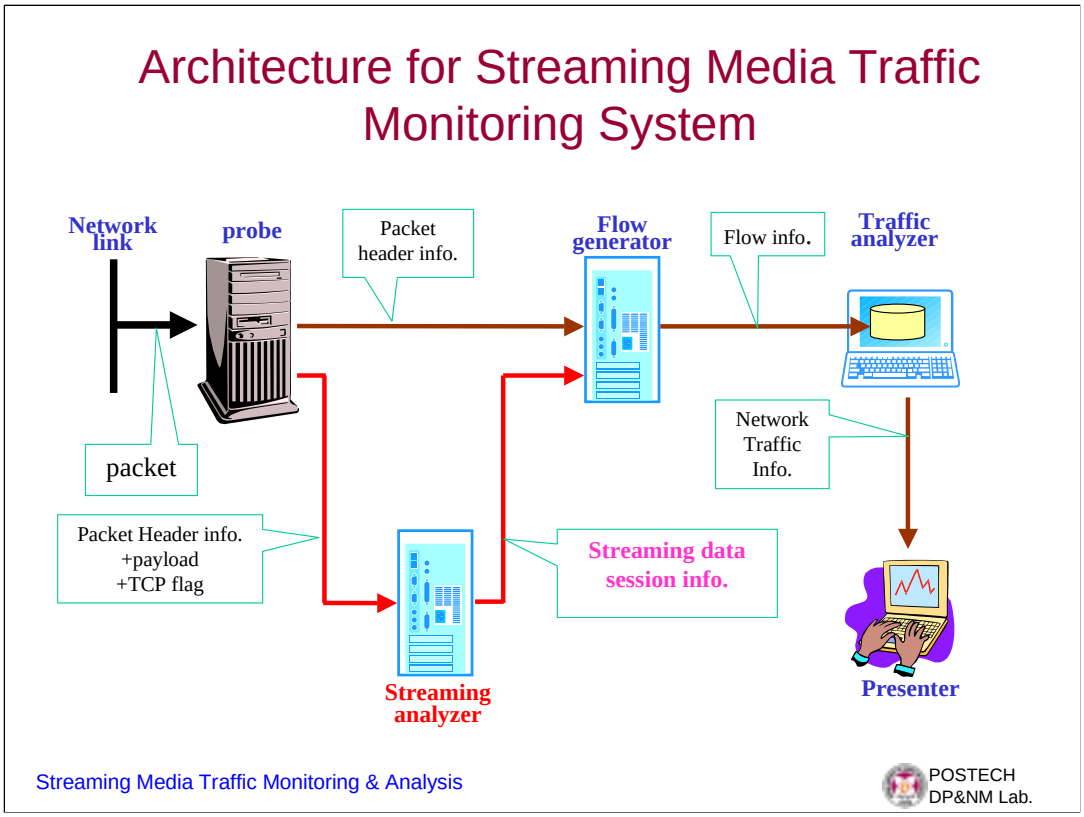
Many streaming service platforms have been introduced to the marketplace, but only three have achieved a substantial user base: Windows Media Technology (WMT) [4], RealNetworks [5], and QuickTime [6]. These platforms differ in their protocols, illustrated in the Table 1.

Streaming Service Platform	Control Session Protocol	Data Session Protocol	Protocol Publicity
RealNetworks	RTSP	RDT	RTSP: IETF RFC 2326 RDT: proprietary
QuickTime	RTSP	RTP	RTSP: IETF RFC 2326 RTP: IEFT RFC 1889
Windows Media Technology	MMS	MMST/ MMSU	proprietary

Table 1 . Comparison of Streaming Media Protocols

Two kinds of streaming sessions are created between the client and server: a ***control session*** and a ***data session***. The control session is responsible for setting up connection and controlling navigation, such as play and pause. This session uses protocols such as RTSP [1] and MMS [4]. The data session sends the streaming media contents to the client over data session protocol, including RDT [5], RTP [3], and MMST/MMSU [4]. We designate each packet related to the control session and data session as a ***streaming control packet*** and a ***streaming data packet***.

The data session uses the transport protocol and port numbers which are dynamically assigned by the control session protocol. Among the control session protocol, RTSP is a standard protocol as illustrated in Table 1. Therefore, we can trace out the process of setting up a RTSP connection. The client sends a SETUP request to the server with the candidates for a protocol and port number (or a range of port numbers) which will be used for the client to receive streaming media data. Next, the SETUP reply contains the protocol and port numbers chosen by the server. By contrast, MMS is not an open but a proprietary control session protocol. It conceals its protocol specification, and is not a text-based protocol. Although not publicly open, we have discovered by inspecting its packets that the client's request in MMS contains the transport protocol and port numbers used for transferring streaming media data. Therefore, we can find the negotiated protocol and port numbers of data session by checking the RTSP and MMS streaming control packets.



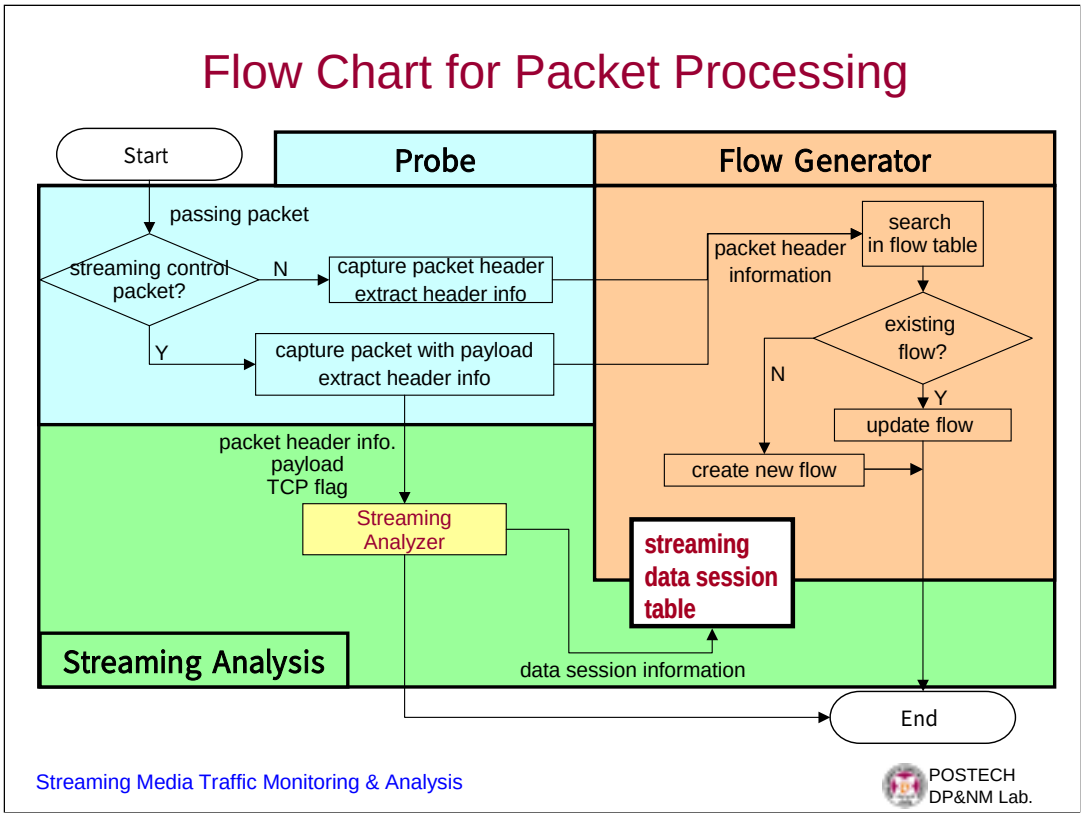
This slide illustrates the overall system architecture for monitoring and analyzing streaming media traffic. The system is composed of five components: a probe, a streaming analyzer, a flow generator, a traffic analyzer, and a presenter.

The probe captures a packet from the network link and extracts information from the packet header. This *packet header information* is offered to the flow generator. If the packet is determined to be a steaming control packet, the probe gives the streaming analyzer a streaming control message which includes the payload.

The streaming analyzer plays an important part in monitoring streaming media traffic. It parses the streaming control message, and discovers the dynamically assigned protocol and port numbers used for streaming data traffic. We will describe the streaming analyzer in more detail in the following slides.

The flow generator makes a flow by collecting a series of packets. Next, it periodically stores flow information into the database. When determining the application of a flow, it references ***streaming data session information*** provided by the streaming analyzer as well as the packet header information.

The traffic analyzer makes analysis tables based on queries about the flow data stored in the database. Then the presenter shows information of streaming media traffic according to the request of a user.

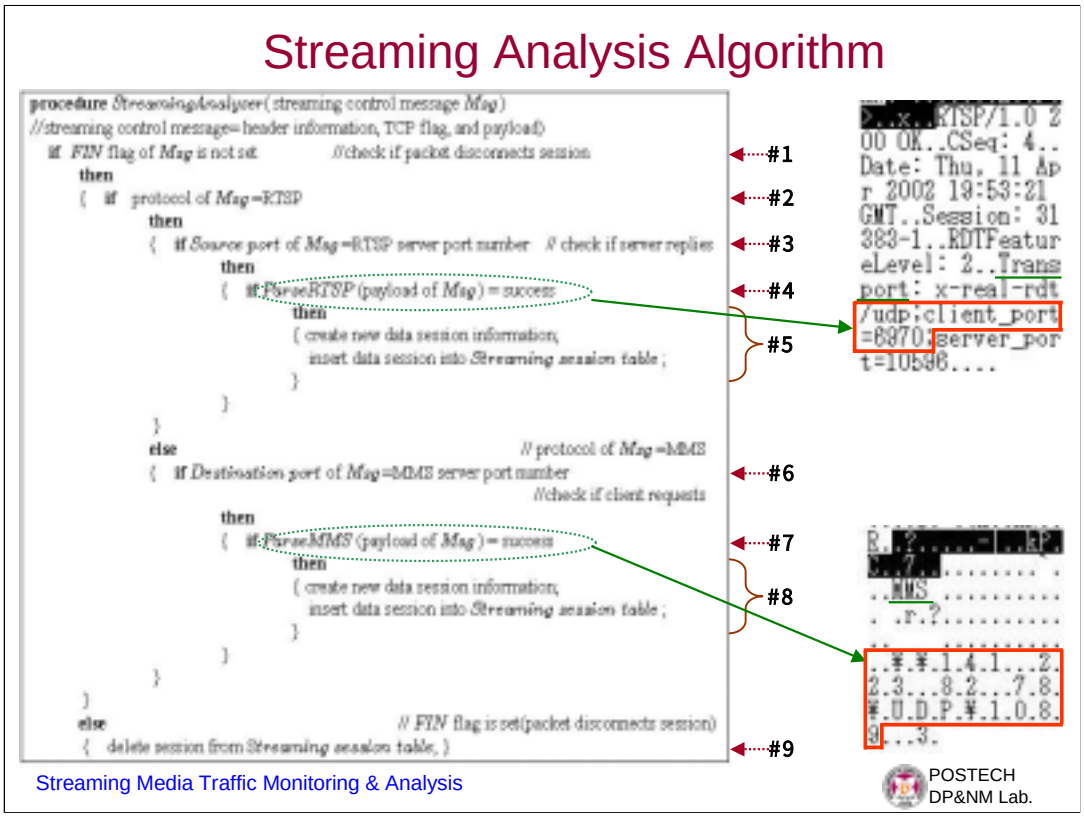


The slide illustrates the flow chart in which a packet is captured and processed. The overall procedure consists of three modules: the probe, flow generation, and streaming analysis

The probe module takes in charge of capturing a packet. When finding a packet passing through the probing point, this module ascertains whether the packet is a streaming control packet or not. If so, the entire packet is captured for analyzing the payload of the packet. If not, only the packet header is captured. After capturing, the module extracts the packet header information, such as the source/destination IP addresses and port numbers regardless of the kinds of packet. This packet header information is provided to the flow generation module.

The flow generation module makes information based on flows. A flow represents a series of packets traveling between ‘‘interesting’’ end points [9]. In this paper, we define a flow as a sequence of packets with the same 5-tuple: source IP address, destination IP address, source port, destination port, and protocol number. Whenever receiving the packet header information, this module determines if there is a flow to which the packet belongs by searching the flow with the same 5-tuple of the packet. According to the search result, the module updates the existing flow or creates a new flow if one does not exist.

Note that the packet header information is not sufficient when the flow generation module generates a flow related to streaming data packets. In general, identifying the application of traffic is based on well-known port numbers. However, streaming media applications use port numbers dynamically allocated, which is unknown in advance. Therefore, we need information to determine whether an unknown traffic is streaming traffic or not. The streaming analysis module provides this information, and the algorithm for obtaining the information illustrated in the next slide.



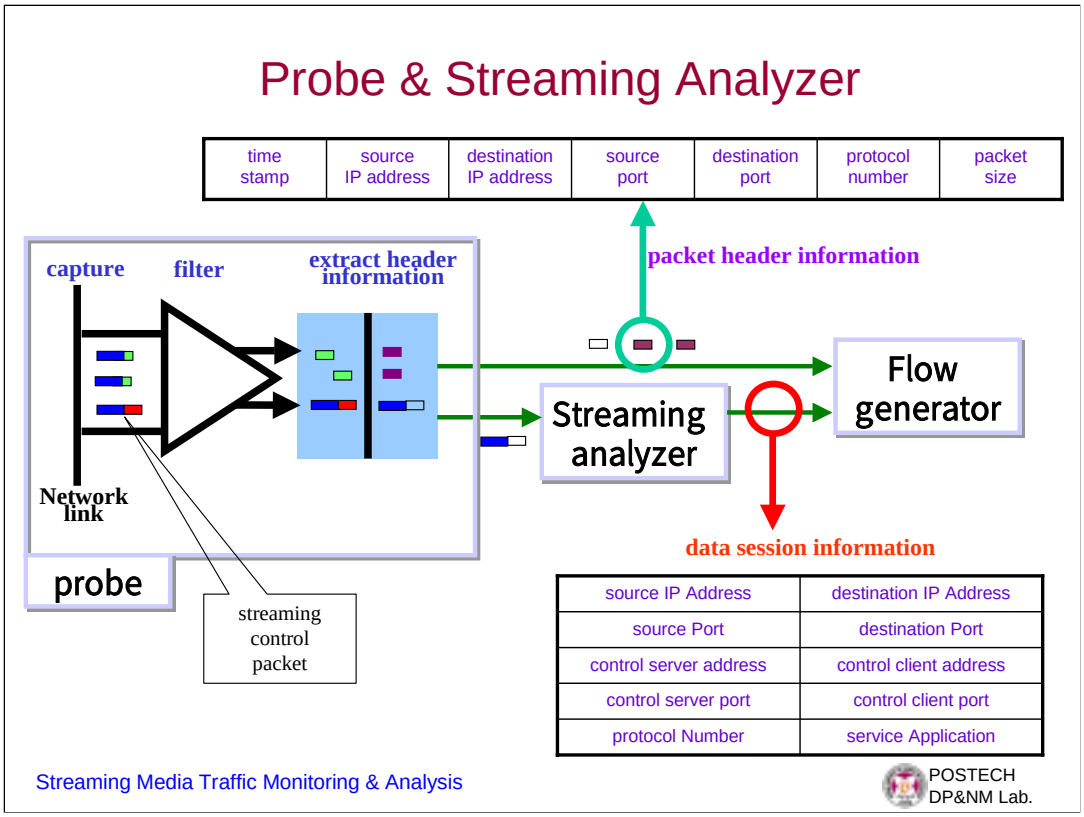
The ***Streaming Analyzer*** procedure described in the slide receives a *streaming control message*, including the packet header information, TCP flag, and TCP payload. Because the TCP flag is excluded from the packet header information, it is input apart from the packet header information.

Analyzing a streaming control message is performed according to the protocol. Hence, we can reduce the analysis overhead by selecting the packet which contains dynamically assigned protocol and port numbers. First, the procedure checks if the FIN flag is set; that is, the streaming session is disconnected. If not, the module determines whether the packet contains the information about streaming data session or not. In the case of RTSP in which a server uses a well-known port number (i.e., 554), the negotiated protocol and port numbers are contained in the server reply packet. Accordingly, the procedure ascertains whether the source port of the packet is 554 in #3 in the slide, in order to select the server reply packet. By contrast, the protocol and port numbers assigned by MMS appear in the packet sent from the client to the server. So, the destination port number is verified if it is MMS server port number (i.e., 1755) in #6, for the purpose of choosing the client request packet.

The payload of the selected packet is parsed according to each control protocol in #4 or #7. In RTSP, the analyzer searches for string components: "RTSP", "OK", "Transport:", ";client_port=", the number or range of numbers, and ";". These string components appear in the payload of the SETUP method reply. By contrast, MMS is a proprietary protocol. Although we do not know clearly about the specification of MMS, we can analyze by searching for strings: "MMS", 'URL-string format', "TCP" or "UDP", and the port number. Unfortunately, the payload formats of the MMS packets may be vary according to implementation. For the future work, we are continuously looking for the more reliable parsing method than matching strings..

After parsing, the procedure confirms if the dynamically allocated protocol and port numbers are found, in #4 or #7. If so, the information about the streaming data session is stored into the ***streaming data session table*** that has information on active streaming data sessions, in #5 or #8. As a result, the unknown-traffic can be identified as traffic carrying streaming media data.

When a streaming session is disconnected, information on the streaming data session must be removed. This information is normally deleted from the steaming data session table in #9 when the TCP FIN is set for the control session in #1. However, the FIN packet might never be captured because of effects such as packet losses or route changes [10]. In such cases, the information is removed from the table by selecting sessions which do not show any activity for a certain period time.



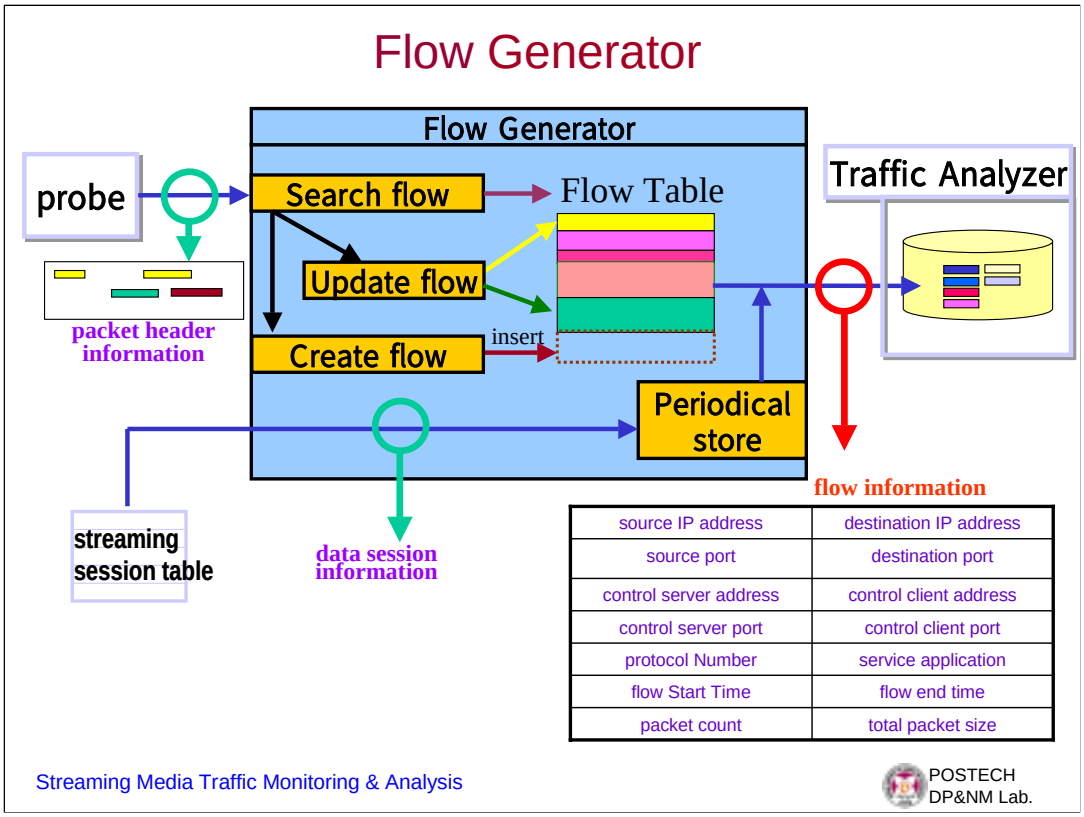
This slide illustrates the functions and parameters of the probe and streaming analyzer. The probe captures an entire packet associated with streaming control protocols in order to parse the payload of the packet. By contrast, it captures only the header of general packets so as to reduce the system overhead. This differentiated capturing is performed through configuring and applying the filter expression of BPF (BSD packet filter) [11].

Another function of the probe is to extract information from the packet header and to send it to the flow generator or streaming analyzer. The format of packet header information appears in the slide. The time stamp is the time when packet is captured.

The probe captures all kinds of packet, and processes them as described above. This method purposes to avoid false-rejecting error, namely, the real streaming data packet is misidentified not to be associated with the streaming session. Consider a streaming data packet which belongs to a new streaming data session which is not yet listed in the streaming data session table. Then the packet may pass the probing point without being captured. Although there is the extra overhead, the system can make a correct analysis.

Using the algorithm for streaming analysis described in the previous slide, the streaming analyzer parses streaming control message received from the probe. After parsing, it can find out the information on the newly created steaming data session. This information is stored into the streaming session table and referenced by the flow generator.

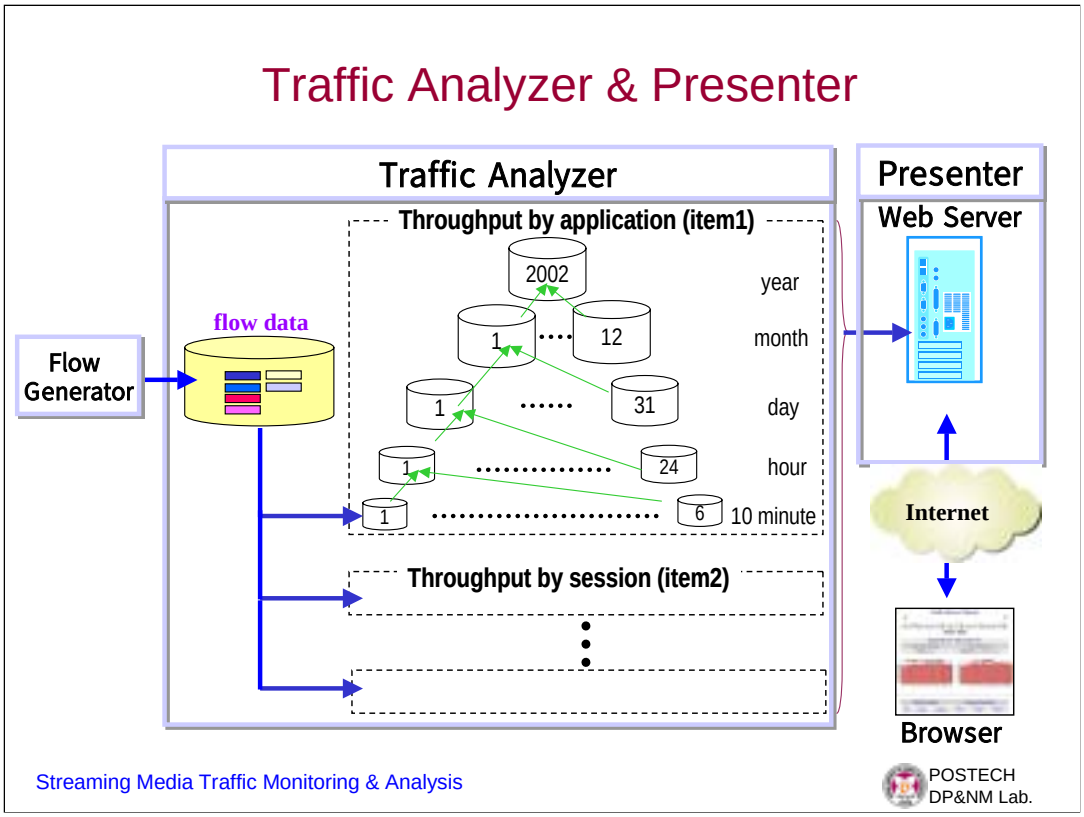
The slide shows the format of *streaming data session information*. In this format, the control session server/client address and port are IP address and the port number of the control session that create that data session, respectively. By making use of these items, the system is aware of the relationship between the control and data sessions. And a service application is currently out of WMT, QuickTime, or RealNetworks.



Whenever receiving packet header information, the flow generator looks up the ***flow table*** to search for an existing flow to which the packet belongs. If a matched flow exists, the packet is added to the flow by updating the flow, which increases the packet count and packet total size, and substitutes the flow end time. If not, a new flow is made of the packet header information. Next, the flow is inserted into the flow table. Flows in the table are periodically stored into the database.

When storing flows into the database, the flow generator selects streaming media traffic. By referencing information in the streaming data session table, it identifies whether an unknown traffic is related to a data session. Because storing flow mostly requests system overhead [7, 13], flows only related to streaming media traffic should be stored. The slide also presents the flow information format stored into the database. The items, control server/client address and port, come from streaming data session information. Integrating these items makes it possible to obtain an analysis based on the session. In the case of a control session flow, these items are filled up with IP address and port number of control session. The flow start time is the time stamp of the first packet in the flow, and the flow end time is updated whenever the packet in the same flow is received. The packet count and packet total size are number and length of packets to belong to the flow, respectively.

As mentioned previously, it is possible to commit a false-rejecting error. The problem caused by a delay when information in the streaming data session table is applied too early or late. This phenomenon occurs in a short delay relative to the period of updating database. In order to solve this problem, we leave the unknown traffic in the flow table, though it is once judged to be non-streaming traffic. As a result, it has opportunity to be checked twice if it is streaming media traffic. By applying information in the streaming data session table at least one period of updating database, we can handle a false-rejecting error.



The traffic analyzer makes an analysis of streaming media traffic by querying flow data stored in the database. It can analyze streaming media traffic in the session level. It is possible for a streaming service to open several sessions. Traffic analyzer can find out and analyze sessions separately. In addition , it integrates information of sessions to belong to one streaming service. For example, it can analyze the traffic volume exchanged in the control and data sessions related to one steaming service, respectively.

The slide illustrates a set of tables made by the traffic analyzer according to an analysis item such as throughput. The example set has a database update time, 10 minutes. In order to provide a short response time, the flow generator prepares traffic data which have been analyzed to be shown. Next, each set has time-series database tables in the cause of temporal analysis, such as hourly, daily, monthly, and yearly tables. Whenever periodically receiving flow information from the flow generator, the traffic analyzer updates these tables in a sequence of hour, day, and so on. When updating the tables, it selects top N entries for the purpose of reducing the system requirements and access time.

The presenter shows analyzed data corresponding request of user through web server. For using traffic prepared by the traffic analyzer, it can provide a user with information in a fast access time.

Related Work

- Fluxoscope, FlowScan
  - ✓ Flow-based traffic analysis
  - ✓ Uses a heuristic method
- Mmmdump
  - ✓ Parse streaming control protocol to get dynamically assigned port number
  - ✓ Changes the packet filter
- Our approach
  - ✓ Captures streaming control packets
  - ✓ Extracts port numbers and protocol used for transferring streaming media data from the payload
  - ✓ Identifies streaming media traffic and non-streaming traffic

Over the years, A few systems have been developed to monitor and analyze streaming media traffic. We survey some systems using the method of streaming analysis that is connected with our approach. Here, we handles only streaming media traffic which consists of traffic related to most popular streaming service platforms: WMT, RealNetworks, QuickTime.

Fluxoscope [8] and ***Flowscan*** [9] are flow-based traffic analysis systems. Their monitoring target related to streaming media is the traffic over RTSP. And they are interested in the distribution. They have a heuristic method which works as follows. The system remembers ongoing streaming control sessions. When a flow is seen with an unknown port number on both ends, it checks to see whether an active streaming control connection exists between the same two hosts. If so, it assumes that the flow corresponds to a streaming data session.

mmdump [10] is a tool for monitoring multimedia media traffic on the Internet. This tool is used to investigate the characteristics of streaming media traffic over RTSP. Accordingly, it mainly examines such characteristics as the packet length distribution and packet arrivals for streaming. But the tool does not analyze MMS traffic that is considered to be most widely used streaming service platform. The tool contains a parsing module for RTSP protocol. It parses the control messages to extract the dynamically assigned port numbers. The parsing module then dynamically changes the packet filter to allow packets associated with these ports to be captured. The results of analysis are provided in the off-line.

Our approach intends to find out the distribution of streaming media traffic. It monitors and analyzes traffic over RTSP and MMS. We can learn on network usage such as the streaming traffic rate of whole and the kind of streaming service, including WMT. Our method extracts information about the data session from the payload of streaming control packet, and identifies streaming media traffic.

Comparison of Related Work

	Fluxoscope FlowScan	mmdump	our approach
Target streaming traffic	Traffic over RTSP	traffic over RTSP	traffic over RTSP, MMS
Purpose	determine traffic distribution	investigate characteristic of traffic	determine traffic distribution
Analysis method	heuristic	analytic	analytic
Capturing overhead	capturing all packet	changing packet filter	capturing all packet
Analyzing overhead	low	parsing control protocol	processing control packet
Accuracy	inaccurate	partially accurate	accurate
Analysis	on-line	off-line	on-line

The table makes a comparison between our approach and the systems outlined previously. ***Fluxoscope*** and ***Flowscan*** capture all kinds of packets, so capturing overhead is high. But they do not have the additional process that exactly analyzes streaming media traffic. The heuristic method does not burden the system with particular overhead to analyze streaming media traffic. On other hand, this analysis may provide inaccurate information. The reason is that traffic seen with an unknown port number may be non-streaming traffic when there is an active streaming control connection between two connected hosts.

mmdump differs in using the packet filter from that of other systems. It captures only packets of interest by changing the packet filter. This method can reduce the resource requirements and capturing overhead. But it is also a burden to frequently compile and change the packet filter. Further, it may miss streaming data packets because it captures only packets in the filtering list that includes the port number to be captured. Before parsing the message of a new streaming control session and updating the filtering list, packets, which belongs to a new streaming data session, may pass without being captured. In addition, some streaming data packets pass the probing point after the deletion of port numbers from the filtering list. In these cases, the analysis results of ***mmdump*** are not accurate.

Our approach captures all packets for the purpose of avoiding a false-rejecting error which is mentioned in ***mmdump***. Next, it performs packet processing for determining the dynamically assigned port number and protocol. By using this method, our approach can produce precise information of streaming media traffic, although it suffers capturing and analyzing overhead.

Conclusion

- Proposed a method and system design for monitoring and analysis of streaming media traffic
  - ✓ analyze streaming control message
  - ✓ extract information about streaming data session (dynamically selected protocol and port number)
  - ✓ determine whether unknown traffic is streaming traffic or not using session information
- Meaning
  - ✓ boost analysis of traffic from packet level to session level
  - ✓ Raise analysis from transport level to application level
- Future Work
  - ✓ to extend the proposed analysis method to other types of traffic

In this paper, we presented a method and system design for monitoring and analyzing streaming media traffic. This method analyzes a streaming control protocol message and extracts information about the streaming data session. The extracted information includes dynamically selected protocol and port numbers, which are used for determining whether unknown traffic is streaming traffic or not. This approach makes it practical to monitor unknown streaming traffic.

We can discover the meaning in that this method boosts analysis of traffic from the packet level to the session level. It does not simply extract header information, but also makes it possible to analyze traffic per session by learning the session information. In addition, it overcomes the problems with existing approaches that use only well-known port numbers of TCP or UDP for identifying application of traffic. By analyzing application messages, this method discovers the status of application and raises the analysis application level.

We are currently implementing a streaming analysis system. We plan to use it to trace out behaviors of streaming media traffic. And we are planning to extend the proposed analysis method to other traffic such as H.323 [12] that use a similar session setup.

References

[1] H. Schulzrinne, A. Rao, and R. Lanphier, “Real Time Streaming Protocol (RTSP),” RFC 2336, April 1998.

[2] H. Schulzrinne, S. Casner, R. Frederick, V, and Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” RFC1889, January 1996.

[3] J.Craig Lowery, “Using Dell PowerApp.cache for Caching and Splitting Media Streams,” http://www.dell.com/us/en/esg/topics/power_ps3q01-lowery.htm, May 2001.

[4] Microsoft, Windows Media Technology,

http://www.microsoft.com/windows/windowsmedia/default.asp.

[5] Real Networks, Real Media Technology, http://www.realnetworks.com/.

[6] Apple, QuickTime, http://www.apple.com/quicktime/.

[7] S. Hong, J. Kim, B. Cho, and J. Hong, “Distributed Network Traffic Monitoring and Analysis using Load Balancing Technology,” 2001 Asia-Pacific Network Operations and Management Symposium, Sydney, Australia, September 2001, pp. 172-183.

[8] S. Leinen, Fluxoscope, http://www.switch.ch/lan/stat/fluxoscope/.

[9] Dave Plonka, FlowScan, http://net.doit.wisc.edu/~plonka/FlowScan/.

[10] Jacobus van der Merwe, R amon Caceres, Y ang-hua Chu, and Cormac Sreenan “mmdump- A Tool for Monitoring Internet Multimedia Traffic,” ACM Computer Communication Review, 30(4), October 2000.

[11] S. McCanne and V. Jacobson. “The BSD packet filter: A new architecture for user level packet capture,” Proceedings of the Winter 1993 USENIX Conference, January 1993, pp. 259–270.

[12] ITU-T, “Recommendation H.323: Visual Telephone Systems and Equipment for Local Area Networks Which Provide a Non-guaranteed Quality of Service,” 1996.

[13] James W. Hong, Soon-Sun Kwon and Jae-Young Kim, “WebTrafMon: Web-based Internet/Intranet Network Traffic Monitoring and Analysis System,”Computer Communications, Elsevier Science, Vol. 22, No. 14, September 1999, pp. 1333-1342