OPTIMIZED PROTECTION OF STREAMING MEDIA AUTHENTICITY

ZHANG ZHISHOU

NATIONAL UNIVERSITY OF SINGAPORE

2007

OPTIMIZED PROTECTION OF STREAMING MEDIA AUTHENTICITY

ZHANG ZHISHOU
(M.Comp. NUS, B.Eng (*Hons.*), NTU)

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE

2007

ACKNOWLEDGEMENTS

First of all, I would like to take this opportunity to express my heartfelt thanks to my supervisors, Prof. Lawrence Wong Wai Choong and Dr. Sun Qibin, for their tireless support and invaluable intellectual inspiration. I greatly appreciate their willingness to share their seemingly endless supply of knowledge and their endeavor to improve every single word in our papers. I particularly appreciate the support from Dr. Sun Qibin, who is also my manager in the Institute for Infocomm Research. He is my mentor not only in my research, but also in my career and daily life. I can never thank them enough. It is his support and encouragement that make this thesis possible.

I would also like to thank Dr. Susie Wee (Director, HP Labs) and Dr. John Apostolopoulos (Manager, HP Labs), for their invaluable and continuous guidance for my research work, presentation skill and paper writing. I particularly appreciate their tireless effort to improve my paper presentation through many runs of rehearsals. Every single discussion with them gives me so much inspiration and encouragement towards the next excellence. They also made my 3-month visit in HP Labs a fruitful and enjoyable learning journey.

In the course of my study, many other people have helped me in one way or another. I would like to thank Dr. He Dajun, Mr. Zhu Xinglei, Dr. Chen Kai, Mr. Yuan Junli, Dr. Ye Shuiming and Mr. Li Zhi for the discussions, suggestions, and encouragements. Their friendship and support also made my work and life very enjoyable over the years.

Last but not least, there is no way I could acknowledge enough the support from my family. I especially thank my parents and my wife, Wu Xiu, for everything. They are and will always be the driving force that helps me pursuing this long term dream and all the future ones. Thanks you very much. Thank you all!!!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF PUBLICATIONS

Journal Papers:

- Zhishou Zhang, Qibin Sun, Wai-Choong Wong, John Apostolopoulos and Susie Wee, "An Optimized Content-Aware Authentication Scheme for Streaming JPEG-2000 Images Over Lossy Networks," IEEE Transaction on Multimedia, Vol. 9, No. 2, Feb. 2007, pp. 320-331
- Zhishou Zhang, Qibin Sun, Wai-Choong Wong, John Apostolopoulos and Susie Wee, "Rate-Distortion-Authentication Optimized Streaming of Authenticated Video," IEEE Transaction on Circuit and System on Video Technology, Vol. 17, No. 5, May 2007, pp. 544-557
- Zhishou Zhang, Qibin Sun, Wai-Choong Wong, John Apostolopoulos and Susie Wee, "Stream Authentication based on Generalized Butterfly Graph", in preparation.
- Qibin Sun, Zhishou Zhang and Dajun He, "A standardized JPEG2000 image authentication solution based on digital signature and watermarking," China Communication, Vol. 4, No. 5, Oct. 2006, pp. 71-80
- Qibin Sun and Zhishou Zhang, JPSEC: Security part of JPEG2000 standard, ITSC Synthesis Journal, Vol.1, No.1, 2006, pp. 21-30

Conference Papers:

- Zhishou Zhang, Qibin Sun, Wai-Choong Wong, John Apostolopoulos and Susie Wee, "A Content-Aware Stream Authentication Scheme Optimized for Distortion and Overhead," In *Proc*. IEEE International Conference on Multimedia and Expo (ICME), July 2006, Toronto, Canada, pp. 541 - 544 (**Best Paper Award**)

- Zhishou Zhang, Qibin Sun, Wai-Choong Wong, John Apostolopoulos and Susie Wee, "Rate-Distortion-Authentication Optimized Streaming with Multiple Deadlines," In *Proc*. IEEE International Conference on Acoustics, Speech and Signal (ICASSP), April 2007, Hawaii, USA, Vol. 2, pp. 701-704 (**Best Student Paper Finalist**)
- Zhishou Zhang; Qibin Sun; Susie Wee and Wai-Choong Wong; "An Optimized Content-Aware Authentication Scheme for Streaming JPEG-2000 Images Over Lossy Networks," in *Proc*. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2006, Toulouse, France
- Zhishou Zhang, Qibin Sun, Wai-Choong Wong, John Apostolopoulos and Susie Wee, "Rate-Distortion Optimized Streaming of Authenticated Video," In *Proc*. IEEE International Conference on Image Processing (ICIP), Oct. 2006, Atlanta, USA, pp. 1661-1664
- Zhishou Zhang, John Apostolopoulos, Qibin Sun, Susie Wee and Wai-Choong Wong, "Stream Authentication Based on Generalized Butterfly Graph," Accepted by IEEE International Conference on Image Processing (ICIP), Sep. 2007, San Antonio, USA
- Zhishou Zhang, Qibin Sun and Wai-Choong Wong, "A proposal of butterfly-graph based stream authentication over lossy networks," In *Proc*. IEEE International Conference on Multimedia and Expo (ICME), July 2005, Amsterdam, The Netherland
- Zhishou Zhang, Qibin Sun and Wai-Choong Wong, "A novel lossy-to-lossless watermarking scheme for JPEG2000 images," In *Proc*. IEEE International Conference on Image Processing (ICIP), Oct, 2004, Singapore, pp. 573-576

- Zhishou Zhang, Gang Qiu, Qibin Sun, Xiao Lin, Zhichen Ni and Yun Q. Shi, "A unified authentication framework for JPEG 2000," in *Proc.* IEEE International Conference on Multimedia & Expo (ICME), July 2004, Taipei
- John Apostolopoulos, Susie Wee, Frederic Dufaux, Touradj Ebrahimi, Qibin Sun, Zhishou Zhang, "The emerging JPEG-2000 Security (JPSEC) Stamdard," in Proc. IEEE International Symposium on Circuits and Systems (ISCAS), May 2006, Greece, pp.3882-3885
- Xinglei Zhu, Zhishou Zhang, Zhi Li and Qibin Sun, "Flexible Layered Authentication Graph for Multimedia Streaming," Accepted by IEEE International Workshop on Multimedia Signal Processing (MMSP), Oct. 2007, Greece
- Kai Chen, Xinglei Zhu and Zhishou Zhang, "A Hybrid Content-Based Image Authentication Scheme," Accepted by the IEEE Pacific-Rim Conference on Multimedia (PCM), Dec. 2007, Hong Kong, China

LIST OF SYMBOLS

Symbols	Semantics
ε	Packet loss rate in the network.
N	Total number of media packets in a sequence that are considered for authentication or transmission.
P_{n}	The n-th packet in a sequence of N packets, which P_0 is the first and $P_{\mathrm{N-1}}$ is the last.
P_{SIGN}	The signature packet, which could be the first packet or the last packet in the sequence.
Δd_n	Distortion increment of the packet P_n. It is the amount by which the overall distortion will increase if P_n is not received or verified.
B_n	Size of the packet P_n.
T_n	Deadline of the packet P_n. A packet received or verified after T_n is equivalent to loss.
θ	A vector of topology polices of the N packets
θ_n	Topology policy of the packet P_{n}. θ_n is basically a set of target packets of the edges originating from P_{n}.
$\lvert\theta_n\rvert$	Redundancy degree of the packet P_n. It is actually the number of outgoing edges from P_n.
π	A vector of transmission policies of the N packets

π_n	Transmission policy of the packet P_n. It indicates when and how the packet P_n is transmitted. For example, with ARQ, it indicates when the packet is transmitted or re-transmitted.
O_n	The amount of authentication overhead (including hash and signature) appended to the packet P_n.
V_n	Verification probability of the packet P_n.
$V(\theta_n)$	Verification probability of the packet P_n, represented as a function of its topology policy.
ε_n	Loss probability of the packet P_n
$\varepsilon(\pi_n)$	Loss probability of the packet P_n, represented as a function of its transmission policy
ρ_n	Transmission cost (per byte) of the packet P_n.
$\rho(\pi_n)$	Transmission cost (per byte) of the packet $\varepsilon(\pi_n)$, represented as a function of its transmission policy
g	Size (in bytes) of a digital signature, which is usually over hundred bytes
h	Size (in bytes) of a hash value. For example, SAH-1 hash has 20 bytes and MD-5 hash has 16 bytes.
D	Overall distortion of authenticated media at the receiver
$D(\theta)$	Overall distortion of authenticated media at the receiver, represented as a function of the topology policy vector θ.
$D(\pi)$	Overall distortion of authenticated media, represented as a function of the transmission policy vector π

O	Total authentication overhead for all N packets
$O(\theta)$	Total authentication overhead, represented as a function of the topology policy vector θ.
$R(\pi)$	Total transmission cost, represented as a function of the transmission policy vector π
N_R	The number of rows in a butterfly graph or Generalized Butterfly Graph.
N_C	The number of columns in a butterfly graph or Generalized Butterfly Graph. In a butterfly graph, $N_C = \log_2 N_R + 1$
$P_{c,r}$	To indicate the packet located in the c-th column and r-th row in butterfly or GBG graph. It corresponds a packet P_{n} where $n = cN_R + r$.
P_q^l	The q-th packet in layer-l in a layered media format, where layer-0 is the base layer.
φ_n	The determining set, the set of packets on which the packet P_n depends for verification in graph-based authentication method.
ϕ_n	The dependent set, the set of packets which depends on P_n for verification in graph-based authentication method

LIST OF ABBREVIATIONS

Abbreviations	Semantics
ARQ	Automatic Repeat Request (a technique used to re-transmit lost packet)
AVC	Advanced Video Coding (Part-10 of MPEG-4 video coding standard, also known as H.264)
CDMA	Code Division Multiple Access
DAG	Directed Acyclic Graph
DSA	Digital Signature Algorithm
DSL	Digital Subscriber Line
DSS	Digital Signature Scheme
ECC	Error Correction Coding
EMSS	Efficient Multi-chained Stream Signature (A graph-based stream authentication method)
FEC	Forward Error Correction (a technique used to fight against network loss or bit error)
IDA	Information Dispersal Algorithm
IEC	International Electrotechnical Commission
IPTV	Internet Protocol Televisions
ISO	International Standard Orgnization
GBG	Generalized Butterfly Graph
JPEG	Joint Photographic Experts Group
JPSEC	JPEG 2000 Security (ISO/IEC 15444-8)
LAF	Loss-Amplification-Factor (a metrics to measure performance of stream authentication method)
MAC	Message Authentication Code

MD-5	Message Digest algorithm
MTU	Maximum Transmission Unit
MPEG	Moving Picture Experts Group
NAL	Network Abstraction Layer
P2P	Peer-to-Peer
P2PTV	Peer-to-Peer Television
QoS	Quality of Service
RaDiO	Rate-Distortion Optimized streaming technique
RDHT	Rate-Distortion Hint Track
R-D-A Optimized	Rate-Distortion-Authentication optimized streaming technique
RSA	A public key cryptographic algorithm by Rivest, Shamir and Adleman
VCL	Video Coding Layer
VoD	Video-on-Demand
VoIP	Voice over IP
SAIDA	Signature Amortization based on Information Dispersal Algorithm
SHA	Secure Hash Algorithm
SVC	Scalable Video Coding (an extension of AVC to support scalability)
TESLA	Time Efficient Stream Loss-tolerant Authentication
W-CDMA	Wideband Code Division Multiple Access
WLAN	Wireless Local Area Network (IEEE. 802.11 series standards)

SUMMARY

Media delivery and streaming over public and lossy networks are becoming practically very important, which is evident in many commercial services like Internet Protocol Televisions (IPTV), Video-on-Demand (VoD), video conferencing, Voice over Internet Protocol (VoIP) and so on. However, the security issues like authentication are serious concerns for many users. Both the sender and the receiver would like to be assured that the received media is not modified by an unauthorized attacker, and any unauthorized modification should be detected.

A conventional crypto-based digital signature scheme can be directly applied to a file (file-based) or each packet (packet-based). However, it does not work effectively for streaming media due to three reasons: 1) a file-based method is not tolerant to network loss while streaming media is usually encoded with error-resilience techniques and therefore is tolerant to network loss; 2) a file-based method does not support the paradigm of continuous authentication as packets are being received; 3) a packet-based method imposes extra high complexity and overhead to the processing and the transmission of streaming media, which by itself takes huge computational power and bandwidth.

To tackle the above issues, we first propose a *Butterfly Authentication* method which amortizes a digital signature among a group of packets which are connected as a butterfly graph. It has lower complexity, low overhead and very high verification probability even in the presence of packet loss, because it inherits the nice fault-tolerance property from the butterfly graph. Furthermore, based on the Butterfly Authentication, we also propose a *Generalized Butterfly Graph (GBG)* for authentication, which supports arbitrary number of packets and arbitrary overhead, and at the same time retains the high verification probability of the Butterfly

Authentication. We experimentally show that the proposed Butterfly Authentication and the GBG Authentication methods outperform existing methods.

However, the above methods and all existing methods assume that all packets are equally important and the quality of authenticated media is proportional to the verification probability, which is usually not true for streaming media. Therefore, we propose a *Content-Aware Optimized Stream Authentication* method, which optimizes the authentication graph to maximize the expected quality of the authenticated media. The optimized graph is constructed in such a way that the more important packets are allocated more authentication information and thereby have higher verification probability, and vice versa. Overall, it attempts to maximize the media quality for a given overhead, or conversely minimize the overhead for a given quality.

Stream authentication imposes authentication dependency among packets, which implies that loss of one packet may cause other packets to not be verifiable. Conventional streaming techniques schedule packet transmissions (e.g., through re-transmission or differentiated QoS service) such that more important packets are delivered with high probability. Nevertheless, conventional streaming techniques do not account for the authentication dependencies, and therefore, straightforward combinations of conventional streaming techniques and authentication methods produce highly sub-optimal performance. To tackle this problem, we propose the *Rate-Distortion-Authentication (R-D-A) Optimized* Streaming method that computes the packet transmission schedule based on both coding importance and authentication dependency. Simulation results show that the proposed R-D-A Optimized Streaming method significantly outperforms the straightforward combination when the available bandwidth drops below the source rate.

CHAPTER 1 - INTRODUCTION

This thesis addresses the problem of providing quality-optimized authentication service for streaming media delivered over public and lossy packet networks. The problem has two aspects: security and quality. The former is to ensure that any unauthorized alteration to the media should be detected by a receiver, while the latter is to optimize media quality at the receiver.

1.1 BACKGROUND

Media delivery and streaming over public networks are becoming practically more and more important, enabled by rapidly increasing network bandwidth (especially at the last mile, e.g. DSL, W-CDMA, CDMA2000, WLAN, etc), huge number of users with Internet access (over 1 billion users as of March 2007 [1]), advanced media compression standards [2][6], and advances in network delivery technologies such as content-delivery networks [11] and peer-to-peer (P2P) systems [12][13][14]. This is also evident in many commercial services like Internet Protocol Television (IPTV), Peer-to-Peer Television (P2PTV), Video-on-Demand (VoD), video conferencing, Voice over Internet Protocol (VoIP), and so on. However, security issues like confidentiality and authentication are serious concerns for many users. For instance, the sender would like to be assured that the transmitted media can be viewed by

authorized people only, and the receiver would like to be assured that the received media is, indeed, from the right sender and that it has not been altered by an unauthorized third party. The confidentiality issue has been addressed by various research works in recent years [15][16][17][19]. Recently, ISO/IEC published a new standard called JPEG 2000 Part-8: Secure JPEG 2000 [4], also known as JPSEC. It addresses security services for JPEG 2000 images and at the same time allows the protected image to retain all JPEG 2000 system features like scalability, simple transcodability and progression to lossless. However, JPSEC does not address the packet loss issue. This thesis examines the problem of authenticating streaming media delivered over public and lossy networks.

Throughout this thesis, the term *authentication* implicitly means three things: *integrity*, *origin authentication* and *non-repudiation*. With integrity, a receiver should be able to detect if the received message has been modified in transit, that is, an attacker should not be able to substitute a false message for a legitimate one. Origin authentication enables a receiver to ascertain the origin of the received message, and an attacker should not be able to masquerade as someone else. Non-repudiation means that a sender should not be able to later falsely deny that he sent a message.

Digital signature schemes like Digital Signature Scheme (DSA) [18] are well-known solutions for data authentication. A sender is associated with two keys: a private key and a public key. The private key is used by the sender to sign a message while the public key is used by a receiver to verify a message. For example, if *Alice* wants to send a message to *Bob*, she signs the message using her private key and sends to *Bob* together with the generated signature. *Bob* then uses *Alice*'s public key to verify whether the received message matches the signature. If the message is modified in transit, it will not be able to pass the verification, hence integrity is

ensured. Since the private key is known to *Alice* only, no one else is able to generate a signature matching the message, and therefore *Bob* is able to ascertain it is indeed from *Alice* (origin authentication). Further, *Alice* cannot deny the message is sent by her (non-repudiation).

Digital signature schemes work neither effectively nor efficiently for streaming media, because the typical requirement assumed for data authentication that the received data must be exactly the same as what was sent by the sender, is not appropriate or practical for many uses of media authentication. Conventional digital signature schemes are not tolerant to network loss, and even a single-bit difference may cause the received media not to pass the verification. However, streaming media is usually encoded with error-resilient techniques [5][25] and is tolerant to a certain level of network loss that is unavoidable when it is delivered over an unreliable channel like a UDP connection. When network loss occurs, the received media may have degraded but still acceptable quality. It is desirable that the authentication solution should be able to verify the degraded media, so long as no packet is modified.

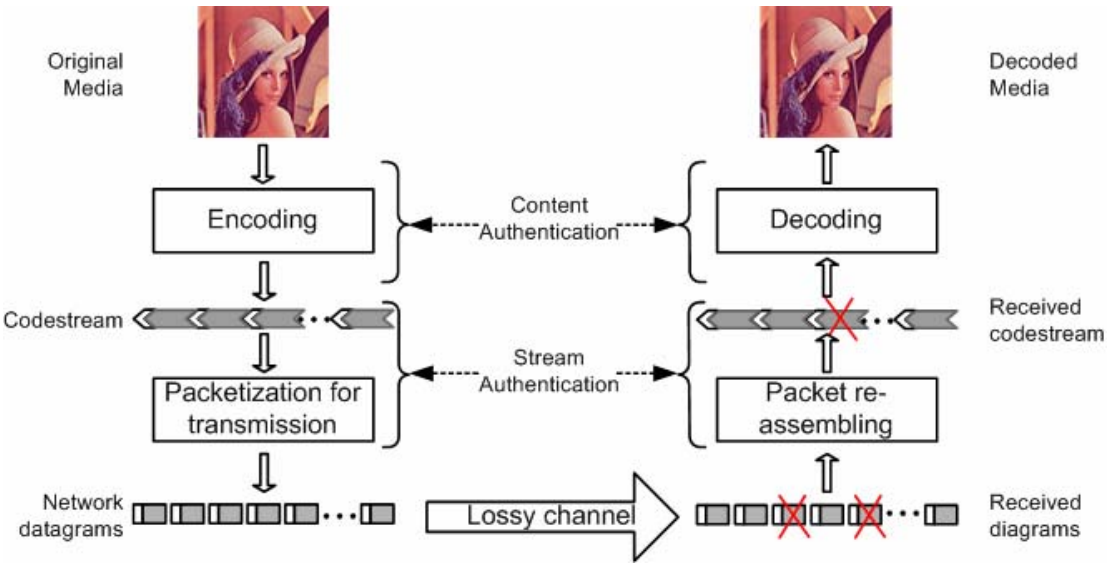

Figure 1-1 – Media transmission over lossy channel

Figure 1-1 illustrates the typical scenario for media communication over a lossy channel. At the sender side, the original media is encoded into a stream, which is basically a sequence of packets. Before network transmission, the packets are then wrapped in datagrams whose size is no larger than the *Maximum Transmission Unit* (*MTU*). A packet might be split into more than one datagram. Throughout the thesis, we denote a "*packet*" as a data unit generated by the media encoding process, and a "*datagram*" as the basic network transmission unit. At the receiver, received datagrams are used to assemble the packets. As the network is lossy, some datagrams may be lost in transit, resulting in corruption of the corresponding packets. Finally, received packets are decoded to reconstruct the media, where various error concealment techniques [5][25] can be applied to recover from the loss.

As illustrated in Figure 1-1, authentication can be achieved at two different levels: content level and stream level. The authentication at content level, also known as content authentication [20][21][22][23][24], has access to the media content. It extracts and signs key features of the media, which are invariant when the media undergoes content-preserving manipulations like re-compression, format conversion and certain levels of network loss. Therefore, content authentication is robust against the distortion introduced by re-compression and channel transmission. However, it is generally more difficult to make useful and mathematically provable statements about the system security for a content authentication method. As shown in Figure 1-2(a), there exists the possibility that an authentic media is falsely detected as unauthentic (i.e., false reject) and an attacked media falsely passes the verification (i.e., false accept).

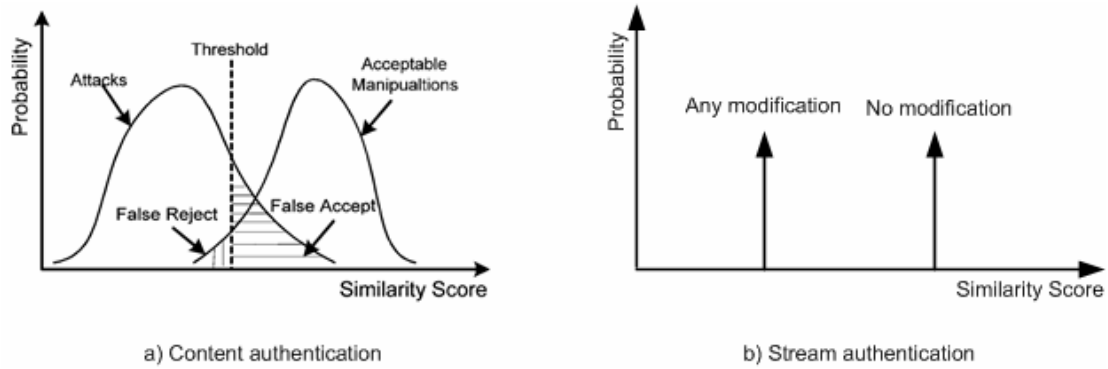

Figure 1-2 – Content Authentication versus Stream Authentication

The authentication at stream level, also known as stream authentication, has access to the packets only. Since stream authentication is achieved using cryptographic hash (like SHA-1) and signature (like DSA) methods [18], it provides a similar level of security to conventional data security techniques, and very importantly provides a mathematically provable level of security. Unlike content authentication, stream authentication has no false rejection or false acceptance, as shown in Figure 1-2(b).

Figure 1-3 illustrates two simple methods to authenticate stream packets. In the first method in Figure 1-3(a), each packet carries its own signature and thereby each received packet is individually verifiable. However, its disadvantage is the high complexity and overhead, as cryptographic signature operations require high computation power and its size is in the order of hundreds of bytes. In the second method, a single signature is computed from a bit string which is the concatenation of all packets. While it has very low complexity and low overhead, it does not tolerate any packet loss, i.e., any packet loss causes all other packets not to be verifiable.

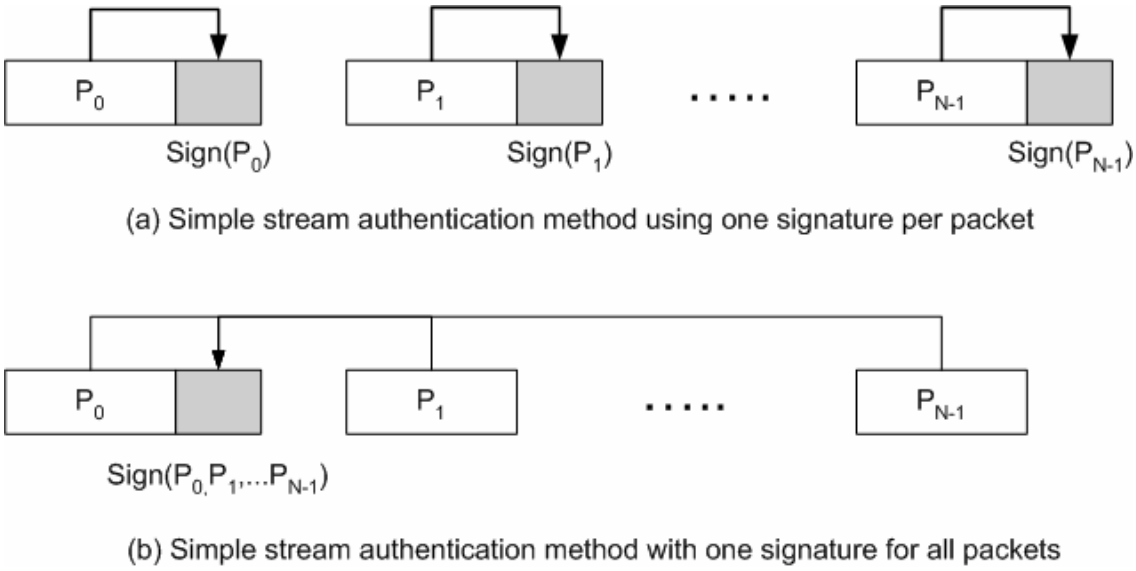

Figure 1-3 - Simple methods to authenticate stream packets

The above two methods are two extreme cases for stream authentication: the first one has very high robustness but also very high complexity and overhead, while the second one has very low complexity and overhead but also very low robustness. More sophisticated methods exist to achieve a trade-off between robustness, overhead and complexity, which can be classified into Erasure-code-based authentication [26][28] and graph-based authentication [29][31][33][34].

The Erasure-code-based authentication method computes a single digital signature from the hash values of the individual packets. To prevent loss of authentication data, Error Correction Code (ECC) algorithm is applied to the digital signature and hash values. The resulting ECC codeword is then divided into segments piggybacking onto the packets transmitted to the receiver. Thus, in the presence of packet loss, the receiver may still be able to recover the authentication data and verify the received packets. The more redundancy added by ECC coding, the more robust it is against packet loss. More details of Erasure-code-based authentication can be found in Section 2.1.2.1.

Graph-based stream authentication connects packets as a Directed Acyclic Graph (DAG), where packets correspond to nodes. A directed edge from packet *A* to packet *B* is realized by appending *A*'s hash (one-way hash) to *B*. There is only one packet carrying the digital signature (which is referred to as signature packet), and each packet has at least one directed path to the signature packet. At the receiver side, the lost packets are removed from the graph and a packet is verifiable if it has at least one directed path to the signature packet. In order to increase the robustness against packet loss, we have to add more redundant edges in the graph. An example of graph-based stream authentication is given in Figure 1-4. The graph-based authentication has low complexity, because it requires only one signature operation for all packets and one hashing operation per packet. In addition, it has either lower sender delay or lower receiver delay, depending on whether the signature packet is the first or last one to be sent. This thesis examines graph-based stream authentication in more details.

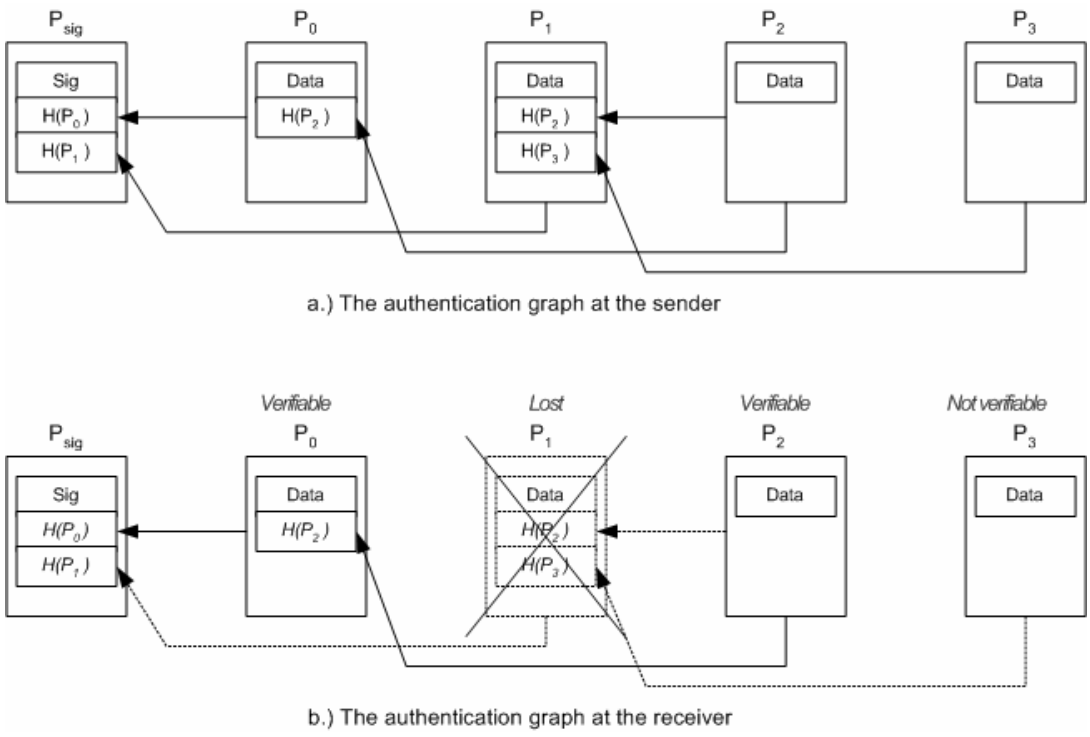

Figure 1-4 – An example of graph-based stream authentication

1.2 PRELIMINARIES

1.2.1 Security Related Concepts

Authentication, Integrity and Non-repudiation

Usually authentication is associated with data integrity, origin authentication, and non-repudiation, because these issues are very often related to each other: Data which has been altered should effectively have a new source; and if the source cannot be determined, then the question of alteration cannot be settled either. Typical methods for providing data authentication are digital signature schemes (DSS) and message authentication codes (MAC). Digital signatures use an asymmetric (public/private) key pair, while MACs use a symmetric (private) key. Both DSS and MAC techniques build upon the use of one-way hash functions.

In one-to-one communication scenario, the symmetric key (i.e., with MAC) is shared by the sender and the receiver, and is unknown to any third party. Thus, the receiver is assured that the received message is indeed from the sender as long as the MAC matches the received message, since the sender is the only party (besides the receiver) who knows the key. However, in one-to-many communication scenario, the symmetric key is shared by more than two parties and the MAC can be generated by any party who has the key. Thus, there is no way for a receiver to be assured of the origin of the received message. The asymmetric key (i.e., DSS) works for both one-to-one and one-to-many communication scenario, because only the sender has the private key used to generate the signature.

Further, DSS provides non-repudiation but MAC cannot. In the case of asymmetric key (i.e., DSS), the sender's private key, which is used to generate

signature, is not known to any other party. Thus, the signature generated with DSS cannot be forged and non-repudiation is automatically provided. However, with symmetric key (i.e., MAC), the same key is used by a sender to generate a signature and also used by a receiver for verification. Given a signature, it is not possible to tell who generated it.

One-way Hash Function

A one-way hash function or cryptographic hash works only in one direction to generate a fixed-length bit-string for any given data with arbitrary size. These hash functions guarantee that even a one-bit change in the data will result in a totally different hash value. Therefore, the use of a hash function provides a convenient technique to identify if the data has changed. Further, by "one-way" we mean that it is computationally easy to compute a hash from a message and it is computationally infeasible to find the message for a given hash. Typical hash functions include MD5 (128bits) and SHA-1 (160bits).

A good one-way hash function is also collision-free, i.e., it is hard to generate two messages with the same hash value. One-way hash is quite a primitive operation in the cryptography world. For example, a digital signature is usually generated from a hash value (one-way hash) computed from a message, instead of directly generated from the message.

Message Authentication Code

A message authentication code (MAC) is a one-way hash function with the addition of a secret key. To prevent an attacker from both changing the data and replacing the original hash value with a new one associated with the new data, keyed hash functions

are used where the hash is computed from a combination of the original data and a secret key. As discussed previously, due to the nature of symmetric key, MAC does not provide origin authentication in one-to-many communication scenario, and it does not provide non-repudiation, either.

Digital Signature Scheme

The digital signature scheme (DSS) includes 1) a procedure for computing the digital signature at the sender using the sender's private key, and; 2) a procedure for verification of the signature at the receiver using the associated public key. Computing a digital signature is very computationally expensive, and depends on the length of the data being signed. Therefore, instead of directly signing the data, the typical approach is to compute a one-way hash of the data and then sign the hash value. Public key DSS is a common technology and has been adopted as an international standard for data authentication, where the private key is used for signature generation and the public key is used for signature verification. The generated signature is usually about 1024bits. As discussed previously, the asymmetric key pair enables DSS to provide integrity, origin authentication and non-repudiation at the same time.

Media Data versus Media Content

Given a specific type of multimedia (e.g., image), the term media "data" refers to its exact representation (e.g., binary bitstream) while the term media "content" refers to the semantics of the same data representation. The term *semantics* refers to the aspects of meaning that are expressed in a language, code, or other form of media representation. For example, after lossy compression the original and reconstructed

media data is different, however the media content or media semantics should be the same (e.g., the same people are visible in both the original and reconstructed image). Semantics measurement is generally subjective, and is a function of the specific applications. For example, matching or similarity score is the most common one used in pattern recognition.

Incidental Distortion versus Intentional Distortion

Incidental distortion refers to the distortion introduced from coding and communication like compression, transcoding, and packet loss, etc. Intentional distortion refers to the distortion introduced by malicious attacks like image copy-paste (e.g., changing the text in a picture), packet insertion, etc. In some applications, the goal of the authentication scheme is to tolerate incidental distortions (i.e., all affected media caused by incidental distortions will still be deemed as authentic media) while rejecting or identifying intentional distortions. Sometimes, the intentional distortion is also referred to as attack.

Content Authentication

The term "content authentication" refers to verifying that the meaning of the media (the "content" or semantics) has not changed, in contrast to data authentication which considers whether the data has not changed. This notion is useful because the meaning of the media is based on its content instead of its exact data representation. This form of authentication is motivated by applications where it is acceptable to manipulate the data without changing the meaning of the content. Lossy compression is an example.

Stream Authentication

The term “Stream authentication” refers to a process to verify that a sequence of packets (or a stream) transmitted over a public and lossy network has not been altered by an unauthorized third party, while tolerating packet loss to occur in transit. The basic idea is to amortize a digital signature among a group of packets to reduce complexity and overhead, and at the same time remain robust against packet loss. Stream authentication can be classified into erasure-code-based stream authentication and graph-based stream authentication. The former applies Error Correction Coding to protect authentication data (digital signature and hash values) from network loss, while the latter adds redundant paths to the Directed Acyclic Graph (DAG) to protect authentication data from network loss.

Authenticated Media

Authenticated media is defined as the media decoded from received and authenticated packets only. That is, a packet received but not verified will be equivalent to loss. This definition prevents packets from alteration and assumes packet loss is not malicious. Note that packet loss could be due to various factors like congestion and transmission bit error. Throughout this thesis, we assume packet loss is not malicious due to: 1) It may not be possible to tell whether a packet loss is caused by network or by a malicious attacker; 2) Media stream is tolerant to packet loss, which might be concealed using various error-resilience and error-concealment techniques.

1.2.2 Media Coding and Streaming

This section gives a brief overview of the latest media formats including the JPEG 2000 image coding standard and the H.264/AVC video coding standard.

JPEG 2000 Images Coding Standard

JPEG 2000 [2] is the latest image coding standard by the Joint Picture Expert Group (JPEG), which is to provide a new image representation with rich set of features, all supported within the same compressed bit-stream. The JPEG 2000 standard can address a variety of existing and emerging applications, including server/client image communication, medical imagery, military/surveillance, and so on. Compared with the baseline JPEG standard, the JPEG 2000 standard supports the following set of features:

- Improved compression efficiency
- Lossy to lossless compression
- Multiple resolution representation
- Embedded bit-stream (progressive decoding and SNR scalability)
- Titling
- Region-of-Interest (ROI) coding
- Error resilience
- Random codestream access and processing
- A more flexible file format

The JPEG 2000 standard employs Discrete Wavelet Transform (DWT) to transform an image into *resolutions* and *sub-bands*, followed by *quantization*. The quantized coefficients are then arranged into *codeblocks*. Figure 1-5 illustrates how an

image of 256x256 pixels is decomposed into three resolutions and each sub-band consists of codeblocks of 64x64 coefficients.

The quantized coefficients are coded in two tiers. In Tier-1, each codeblock is encoded independently. The coefficients are bit-plane encoded, starting from the most significant bit-plane all the way to the least significant bit-plane. Furthermore, all bit-planes except the most significant one are split into three sub-bit-plane passes (*coding passes*), where the information that results in largest reduction in distortion will be encoded first. Each coding pass is associated with a distortion increment, the amount by which the total distortion will decrease if the coding pass is correctly decoded towards the reconstructed image.

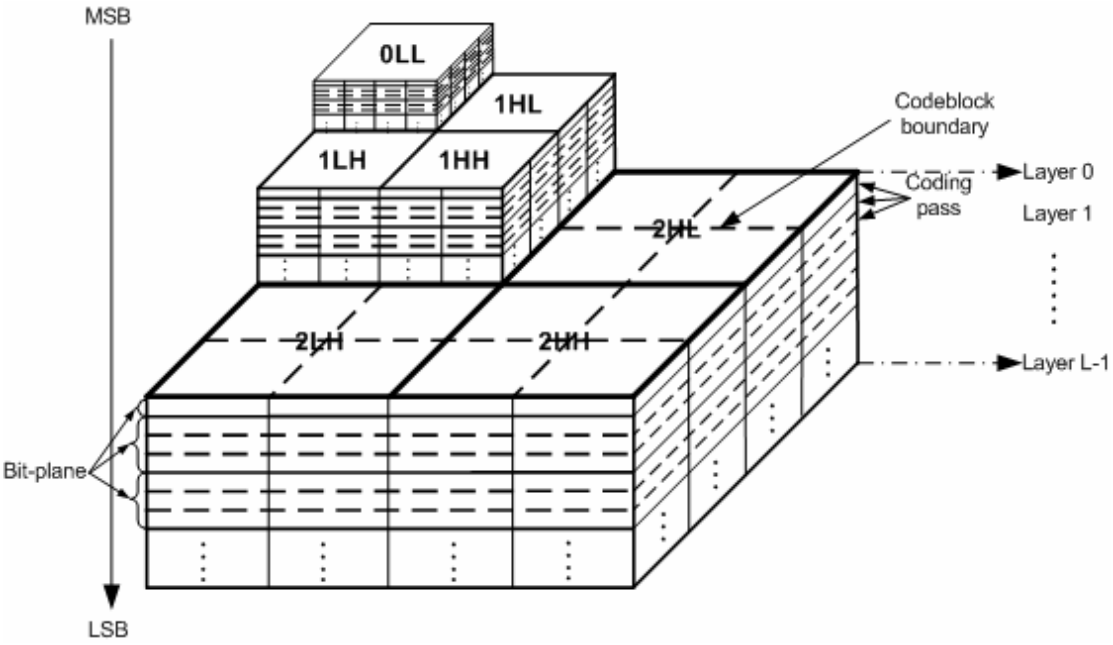

Figure 1-5 – JPEG 2000 resolutions, sub-bands, codeblocks, bit-planes and coding passes

The Tier-2 coding introduces another three structures, *layers*, *precincts* and *packets*. The layers enable SNR scalability and each layer includes a number of consecutive coding passes contributed by individual codeblocks. The precinct is a collection of spatially contiguous codeblocks from all sub-bands at a particular

resolution. All the coding passes that belong to a particular precinct and a particular layer constitute a packet.

The distortion increment of a packet is the summation of the distortion increments of all coding passes that constitute the packet. Furthermore, within the same precinct, a high-layer packet will depend on all the lower-layer packets for decoding (i.e., simple linear dependency). The distortion increment, together with dependency relationship, is used to measure the importance of a packet in a JPEG 2000 image.

More details of JPEG 2000 standard can be found in [3].

H.264/AVC Video Coding Standard

H.264/AVC [6][7] is the latest international video coding standard by ITU-T Video Coding Expert Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). This new standard is designed for higher compression efficiency and network-friendliness. Therefore, the H.264/AVC standard can be used for applications like video broadcasting, video conference, video-on-demand, video streaming service, multimedia messaging service, and so on. Compared with prior video coding standards, H.264/AVC has many new features, some of which are highlighted as follows:

- Higher compression efficiency, achieved by using various motion compensation techniques like quarter-sample-accurate, variable block-size and multiple reference pictures.
- Enhanced error-resiliency, achieved by using techniques like Network abstraction layer (NAL), parameter set structure, flexible slice size, flexible macroblock ordering (FMO), and so on.

The H.264/AVC has a Video Coding Layer (VCL), which is designed to efficiently represent the video content, and a Network Abstraction Layer (NAL), which formats the VCL representation of the video in such a way that it is convenient and efficient to be transported by different networks.

In the VCL layer, a *picture* is partitioned into fixed-size *macroblocks* (a 16x16 rectangular area), which are the basic building blocks of the standard. A *slice* is a sequence of *macroblocks* which are processed in the raster-scan order. A picture may be split into one or several slices. Slices are self-contained in the sense in that a slice can be correctly decoded without the use of data from other slices in the same picture. The slices can be coded with different coding types as follows:

- **I-Slice:** A slice in which all macroblocks are coded using intra predication, i.e., prediction from the samples in the same picture.
- **P-Slice**: In addition to the coding types in I-Slice, a P-Slice also has some macroblocks coded using inter-predication (i.e., prediction from the samples in different pictures) with at most one motion-compensated predication signal per predication block.
- **B-Slice**: In addition to the coding types in P-Slice, a B-slice has some macroblocks coded using inter-prediction with two motion-compensated prediction signals per prediction block.

The coding dependency is very complicated in H.264/AVC, because any I-slice, P-slice or B-slice may be used for prediction of some other slices. This is exacerbated by the fact that a slice may depends on more than one slice.

The Network Abstraction Layer (NAL) is to provide “network friendliness” to enable simple and effective customization of the use of VCL for a broad variety of systems. The NAL structure of H.264/AVC facilitates the ability to map VCL data to

transport layers such as RTP/UDP/IP for real-time wire-line and wireless network service, File format, H.32X and MPEG-2 systems for broadcasting service.

The coded video data is organized into NAL units, each of which is effectively a packet that contains an integer number of bytes. The NAL units can be classified into VCL and non-VCL NAL units. The VCL NAL units contain the data that represent the values of the samples in the video pictures, and the non-VCL NAL units contain any associated additional information such as parameter sets and supplemental enhancement information.

Similar to the JPEG 2000 packets, each VCL NAL unit is associated with a distortion increment, the amount by which the total distortion will decrease if the NAL unit is correctly decoded. In addition, the NAL units also have inter-dependency. For example, a VCL NAL unit may depend on a non-VCL NAL unit containing parameter set information, and a VCL NAL unit containing a P-Slice or B-Slice may depends on some other NAL units for motion compensation. Therefore, the importance of each packet can be measured by the distortion increment associated with each NAL unit and the dependency relationship among them.

More details of H.264/AVC can be found in [7] and [8].

Media Delivery versus Media Streaming

The term “media delivery” refers to a process where every media packet is simply transmitted once to a receiver, which is not adaptive to network condition and packet importance. All packets are treated equally for network transmission. Media delivery is typically used for static media like JPEG 2000 image data that has no strict timing requirement. The term “media streaming” refers to a more sophisticated process where the sender actively schedules packet transmission based on network condition

and packet importance. For instance, the sender could allocate more transmission opportunities to more important packets, or actively prune less important packets when network is congested. Further, packet transmission is scheduled to satisfy timing requirement. Media stream is more appropriate for media like H.264/AVC video, where each frame must be delivered before a specific deadline in order to ensure a smooth play out at the receiver.

1.2.3 Channel Model

Throughout the thesis, the channel is modeled as an independent time-invariant packet erasure channel. Time-invariant channel means that the packet loss probability and delay are independent of the time when the packet is injected into the channel. The term "Packet erasure channel" refers to a transmission channel where a packet can be either received correctly or lost in transmit. It models the end-to-end communication channel based on UDP/IP, which is often used to for media communication. In the UDP/IP protocol stack, the MAC header and the IP header include a checksum field for error detection and correction. A packet received with error will be dropped and it appears be lost to the application layer. Only packets received correctly are passed to the application layers. Therefore, from application point of view, a UDP/IP-based channel can be considered as a packet erasure channel.

Packet loss is most likely caused by buffer overflow at intermediate routers at the time of congestion or caused by active packet dropping by routers to avoid network congestion. If a packet is not lost, its forward trip time, from the time it is sent out to the time it is delivered to the receiver, consists of the queuing delay in the intermediate routers and the propagation delay in the network link. Usually, the forward trip time follows a Shifted Gamma distribution.

For a media delivery scenario (for static media like image), only loss probability is considered, because packets do not have strict timing requirement. For example, in an image communication system, all packets of an image share the same deadline, which is usually quite relaxed. In this case, packet delay is less important. However, for a media streaming scenario (for media like video and audio), packets have more strict deadlines and they must be delivered before their respective deadline to ensure a smooth play out at the receiver. A packet received after its deadline is equivalent to loss. Therefore, both packet loss probability and delay have to be considered. The effective packet loss probability ε is computed by Eq(1.1), where t is packet arrival time and τ is the deadline.

$$\varepsilon = \Pr(lost) + (1 - \Pr(lost)) \Pr(t > \tau \mid not\ lost) \tag{1.1}$$

1.2.4 Attack Model

Packets transmitted over public network can be captured and modified by unauthorized party. The possible attack can be summarized as follows:

Packet Modification

A packet can be modified or replaced with another packet, which may lead to changed streaming media content. For example, when a packet corresponding to a region of an image is modified, the image transmitted by the sender and the image viewed by the receiver may have different semantic meaning. Packet modification should be detected by a receiver.

Packet Insertion

The attacker can insert new packets that do not belong to the original streaming media, which may also change the media content. For example, for video streaming, the attacker can insert one or more packets corresponding to additional frames, and the receiver may perceive additional motion that is not present in original media. Packet insertion should be detected by a receiver.

Packet Removal

The attacker can also remove packets that belong to the original stream media. However, it is difficult to tell whether packet loss is due to "packet removal" attack or due to network loss. As various error-resilience and concealment techniques [5][25] are available to recover from packet loss, the streaming media can still be reconstructed in the presence of certain levels of packet loss, although with degraded quality. Therefore, we assume packet loss is not malicious throughout this thesis.

1.2.5 Performance Metrics

The performance metrics for stream authentication are summarized as follows:

Verification probability: Probability that a received packet is also verifiable. Ideally, all received packets can be verified, however, this requires high overhead and computational complexity, motivating the need for alternative techniques which also provide high verification probability but at significantly lower costs. Verification probability is used to measure robustness against network loss. A higher verification probability indicates higher robustness against packet loss.

Complexity: The computational resources required to sign the stream packets at the sender and to verify the received packets at the receiver. As the media stream

typically involves a huge amount of continuous packets, this requirement becomes even more critical when the receiver is a mobile device with limited computational capabilities. Since signature generation and verification are computationally expensive, the complexity is usually closely related to the number of signature operations required at the sender and receiver.

Overhead: The additional information associated with the authentication information transmitted along with the media packets. The additional information may include MAC values, digital signatures, or hashes. It is important to minimize this overhead, especially in settings where the available network bandwidth is limited.

Sender delay: The additional delay placed on a packet before it can be transmitted because of the authentication processing (e.g., processing a block of packets). In real-time communication scenarios, a high sender delay often requires a large buffer at the sender. For pre-recorded media, e.g., in Video-on-Demand and media broadcast scenario, the signing process can be done off-line, and hence sender delay does not matter.

Receiver delay: The delay from the time when a packet is received to the time when it can be verified by the receiver. A high receiver delay often requires a large buffer at the receiver. For streaming media, usually each packet has an associated playout deadline after which it becomes useless, therefore it is desirable to design the authentication method such that the receiver delay does not result in a packet missing its deadline.

Media quality: The quality of authenticated media at the receiver side, measured as Peak Signal-to-Noise Ratio (PSNR). Assuming packets are equally important, the media quality is proportional to the verification probability. However, in reality, the importance of media packets varies a lot, leading to a situation where media received

with higher verification probability might have lower media quality than the same media received with lower verification probability, depending on whether or not the important packets pass the verification. For media, we argue that media quality is a more meaning performance metric, because the quality matters to human perception.

Note that verification probability and overhead are competing goals. It is desirable to have minimal overhead and maximal verification probability. To increase the verification probability, one has to add more redundancy for authentication data, leading to higher overhead. Thus, a good authentication scheme has to balance these two metrics. Similarly, the sender delay and receiver delay are also competing goals. In graph-based stream authentication, if a signature packet is the first packet to be sent, sender delay is high and receiver delay is low. Otherwise, sender delay is low and receiver is high.

1.3 MOTIVATIONS

In general, the motivation of this thesis is to provide a practical solution to authenticate media with optimized quality at the receiver in the presence of network loss. A media stream comprises a sequence of packets transmitted over the network. A received packet can be classified into four categories: 1) *verified packet*; 2) *lost packet*; 3) *modified packet*; and 4) *unverifiable packets*. The authenticated media is reconstructed from verified packets only. The lost packets are due to network condition such as congestion and fading channel, which is out of the scope of this thesis. The modified packets and unverifiable packets are discarded due to security concerns. The unverified packets are due to incomplete authentication data, caused by network loss. Recall that authentication data piggybacks onto the packets, and

therefore packet loss will lead to partial loss of authentication data. This thesis studies the problem of how to minimize the number of unverifiable packets, in order to maximize the quality of authenticated media at the receiver.

More specific motivations of this thesis are summarized as follows:

1.3.1 Optimized Verification Probability

In a simple media delivery scenario, packets are simply transmitted over a network and, therefore, every packet is subjected to network loss with a certain probability. Packet loss leads to incomplete authentication data, which in turn leads to unverifiable packets. In order to reduce the number of unverifiable packets (i.e., to increase verification probability), we need to make the authentication data more robust against packet loss by adding more redundancy (i.e., more overhead). Therefore, a trade-off exists in stream authentication between overhead and verification probability. In general, the greater the overhead is, the higher the verification probability. Ideally, it is desirable to have an authentication method with very low overhead and at the same time very high verification probability. But in reality, we have to find a balance between overhead and verification probability. The problem is to find an optimal operation points such that the verification probability is maximized for a given overhead, or conversely the overhead is minimized for a given verification probability.

1.3.2 Optimized Media Quality

Media packets have unequal importance, i.e., some packets have higher importance than the other packets in terms of their impact to the quality of the reconstructed

media. For example, in a video stream, loss of an I-frame packet will have error propagation to all subsequent packets until the next I-frame. Therefore, I-frame packets are much more important than P-frame and B-frame packets. Similarly, P-frame packets are more important than B-frame packets. Further, two packets of the same type may also have different importance, depending on video motion. Therefore, we argue that media quality should be used to measure the performance of an authentication method for media, instead of verification probability, because media quality matters to human perception. It is desirable to have an authentication method such that more important packets have high verification probability, and vice versa. In other words, the packet importance information is utilized to design an authentication method such that more important packets are allocated with more overhead and hence have higher verification probability. This will result in more optimized allocation of overhead and subsequently better media quality at the receiver.

1.3.3 Alignment of Coding Dependency and Authentication Dependency

It is well known that packets inherit coding dependency from the media encoding process, i.e., a packet may need another packet for decoding. For example, in a JPEG 2000 stream, a packet in a higher layer depends on the corresponding packets in the lower layers for decoding. On the other hand, stream authentication also imposes dependency among packets, that is, verification of one packet may depend on some other packets. If the coding dependency is not aligned with authentication dependency, the receiver might get some packets which are decodable but unverifiable, or verifiable but not decodable. However, a packet is useful if and only if it is both decodable and verifiable. An example of misalignment is when two packets (A and B) have corresponding coding and authentication dependency, that is,

A depends on *B* for decoding and *B* depends on *A* for verification. Loss of one packet causes the other packet to be useless (either unverifiable or not decodable). Therefore, the problem is to design the authentication method that aligns the authentication dependency with the coding dependency.

1.3.4 Joint Streaming and Authentication

The Rate-Distortion Optimized (*RaDiO*) Media Streaming technique [38][41] schedules packet transmissions based on packet coding dependency, coding importance, packet size and playout deadline. Stream authentication ensures that only authenticated packets are used to reconstruct the media at the receiver, and it imposes authentication overhead and dependency among packets. It is desirable to have R-D optimized quality for the authenticated media. However, straightforward concatenation of *RaDiO* and stream authentication produces highly sub-optimal performance because *RaDiO* does not account for authentication dependency and overhead. For example, given two packets *A* and *B*, where *A* has higher coding importance than *B* and *A* depends on *B* for verification, *RaDiO* will assign more transmission opportunities to packet *A* and fewer opportunities to packet *B*. However, packet *A* cannot be verified unless packet B is received. To get optimized media quality, we have to take into consideration authentication overhead and dependency, besides the original attributes used by *RaDiO*.

1.4 MAJOR CONTRIBUTIONS

The major contributions of the thesis are summarized as follows:

1.4.1 Butterfly Authentication

For stream authentication, there exists a trade-off between the overhead and the verification probability. It is desirable to find a balance point where the verification probability is maximized for a given overhead, or the overhead is minimized for a given verification probability. Thus, we need to design an authentication graph to connect the packets in such a way that each received packet has a path to the signature packet with high probability. Towards this goal, we propose a Butterfly Authentication method based on butterfly graph which is commonly used in communication networks for parallel computer and distributed system. In a butterfly graph, every node/packet is connected to two other packets/nodes that are independent of each other for authentication (referred to as *independency property*), and thereby maximize its verification probability.

We experimentally show that the butterfly graph is a near-optimal graph when the overhead is around 2 hashes per packet. We developed a greedy algorithm to build an authentication graph step by step. At each step, the greedy algorithm considers all possibilities of adding one more edge to the graph and chooses the edge that gives the highest average verification probability. At some point when the overhead reaches 2 hashes per packet, the resulting graph is similar to a butterfly graph in the sense that it has the independency property. In addition, experimental results also show that the Butterfly Authentication has higher verification probability than existing graph-based methods when overhead at 2 hashes per packet.

1.4.2 Generalized Butterfly Graph Authentication

Although the Butterfly Authentication method has higher verification probability than existing graph-based methods, it also has several limitations: 1) The number of nodes in a butterfly graph is not flexible; 2) The overhead of butterfly graph is not flexible either (i.e., fixed at 2 hashes per packet); 3) The signature packet grows with the total number of packets in the graph. If the signature packet is bigger than a MTU, it has to be segmented for transmission – increasing its loss probability and negatively impacting all other packets. To overcome the above 3 limitations of Butterfly Authentication, we propose the GBG authentication graph.

Compared with the Butterfly Authentication graph, the GBG graph supports arbitrary overhead and arbitrary number of packets, and the signature packet does not grow with the total number of packets. The GBG framework supports a wide range of possible authentication graphs, and the problem of finding the best authentication graph for a given situation corresponds to a design problem. The input parameters include the total number of packets, packet loss rate and overhead budget. The output parameters include the number of rows/columns, edge placement and the number of transmissions for the signature packet.

In addition, we also propose a new evaluation metric called the *Loss-Amplification-Factor (LAF)*, defined as the ratio of effective loss rate over packet loss rate. The effective loss rate accounts for both lost packets and packets which are received but not verifiable. The LAF measures the extent to which the authentication method exacerbates the effective loss rate. Ideally, the LAF would equal to 1, i.e., all received packet are verifiable. The closer the LAF for an authentication method is to 1, the greater its robustness is against network loss.

Experimental results demonstrate that the GBG authentication method has significant performance improvement over existing graph-based methods like EMSS and Augmented Chain.

1.4.3 Content-aware Optimized Stream Authentication

We argue that a more appropriate evaluation metric should be the media quality (i.e., Peak Signal-to-Noise Ratio, or PSNR) for a stream authentication method, because the quality (instead of verification probability) matters for human perception. If it is assumed that all packets are equally important, the verification probability is actually equivalent to media quality. However, media packets usually have varying importance, and some packets are more important than the others. This naturally motivates us to allocate more redundant authentication information for the more important packets in order to maximize their verification probability and thereby maximize the overall media quality.

Towards this goal, with awareness of media content, we formulate an optimization framework to compute an authentication graph to minimize the expected distortion (i.e., to maximize the expected quality) at the receiver, given an overhead and packet loss rate. In other words, the distortion-overhead performance of an optimized authentication graph lies in the convex hull of the set of all distortion-overhead performances. As an example, we show how to compute an optimized authentication graph for a JPEG 2000 image, and how to align the coding dependency (between different layers) with the authentication dependency. In view that the optimization process has high computational complexity, we also propose a simplified authentication graph construction method that requires much lower complexity.

Through system analysis and simulation, we demonstrate that the proposed scheme achieves an R-D curve of the authenticated media which is very close to the R-D curve when no authentication is required, and it substantially outperforms existing schemes at all bit-rates and packet loss rates.

1.4.4 Rate-Distortion-Authentication Optimized Streaming

When conventional *RaDiO* techniques are applied to a media stream protected using graph-based authentication method, they produce highly sub-optimal performances for authenticated video, because they optimize rate and distortion only, where the "rate" includes the data rate of coded media data and the "distortion" is measured by the difference between the original media and the media decoded from all received packets. For a media stream protected with a graph-based authentication method, each packet is associated with two more parameters: *authentication importance* and *overhead size*. Authentication importance is the additional expected distortion increment due to the unverifiable packets caused by packet loss, and the overhead size is the size (in bytes) of the authentication data appended to this packet. Conventional *RaDiO* techniques do not consider authentication and, therefore, are referred to as *authentication-unaware* techniques.

We propose the *Rate-Distortion-Authentication* (*R-D-A*) optimized streaming technique to achieve optimized quality for authenticated media. The *R-D-A* Optimization is defined as the rate-distortion optimization for authenticated video, where the "rate" includes data rate for coded media data and the authentication overhead and the "distortion" is measured by the difference between the original media and the authenticated media. The *R-D-A* optimization method is able to achieve optimized performance for authenticated video as it accounts for authentication

importance and overhead size. Considering that *R-D-A* Optimization has high complexity, we subsequently propose a low-complexity algorithm, whose performance is close to the fully-blown algorithm. Lastly, we also propose an extension to the *R-D-A* Optimization framework that considered multiple deadlines associated with each packet.

We conduct simulations to compare the *R-D-A* Optimized method and the authentication-unaware method. The experimental results demonstrate that the *R-D-A* Optimized method outperforms the authentication-unaware methods. Indeed, the *R-D-A* Optimized method is the only method that works at low bandwidth. The authentication-unaware methods do not work at low bandwidth, as the media quality drops quickly to unacceptable level when the bandwidth drops below the source rate.

1.5 THESIS OUTLINE

The rest of this thesis is organized as follows:

Chapter 2 gives an overview of related works, including various methods for stream authentication and the state-of-the-art media streaming techniques. For stream authentication, we describe and compare both the erasure-code-based and graph-based authentication. Then, we describe the Rate-Distortion Optimize (*RaDiO*) streaming techniques.

Chapter 3 proposes authentication methods based on butterfly graph and generalized butterfly graph (GBG). In this chapter, we assume a media delivery scenario where all packets are treated equally for network transmission. In this context, the media quality is proportional to verification probability. First, we propose the *Butterfly Authentication* method connecting packets as a butterfly graph. Analysis

and experimental results show that it outperforms existing methods in term of verification probability for a given overhead. Further, based on butterfly graph, we also propose a more flexible graph structure called *Generalized Butterfly Graph* (*GBG*), which supports an arbitrary number of packets and arbitrary overhead budget.

Chapter 4 proposes a Content-aware Optimized Authentication method, by taking into consideration varying packet importance. First, we describe a distortion-overhead optimization framework used to construct an authentication graph, where more important packets are allocated with more overhead, hence, have higher verification probability. The resulting authentication graph is able to give a maximized expected quality (or minimized expected distortion) of the authenticated media. Second, based on the distortion-overhead optimization framework, we also propose a *Content-aware Optimized Authentication* method as well as a simplified authentication graph with low complexity. Thirdly, we present analysis and simulation results to validate the proposed methods.

Chapter 5 deals with media streaming scenario, where the transmission schedule is adaptive to network condition and packet importance. We describe the proposed *Rate-Distortion-Authentication* (*R-D-A*) Optimized streaming method. The packet transmission schedule is computed by accounting for the authentication dependency and overhead, besides the coding importance and packet size. Simulation results show that the proposed *R-D-A* method gives higher media quality than straightforward concatenation of *RaDiO* and stream authentication. Specifically, at low transmission rate (when bandwidth is lower than source rate), the proposed *R-D-A* method is still working, while all other methods fail.

Chapter 6 concludes the thesis and gives suggestions on future directions. One possible direction is to further examine the problem of joint authentication and

streaming. In Chapter 4 we have described how to construct authentication graph, assuming transmission policy is fixed (i.e., one transmission per packet). In Chapter 5 we describe how to schedule packet transmission, assuming authentication graph is fixed. If the graph construction is jointly done with packet scheduling, we can get even better media quality at the receiver.

CHAPTER 2 - OVERVIEW OF STREAM AUTHENTICATION AND MEDIA STREAMING TECHNIQUES

This chapter gives a brief overview of the related work, including existing work on stream authentication and media streaming. The former is to assure the receivers that the received packets are not modified by unauthorized attacker and they are, indeed, from the claimed sender, even in presence of network loss. The latter is to compute transmission schedule (when and how to transmit individual packets) to maximize the expected media quality at the receiver. In this chapter, Section 2.1 reviews the existing stream authentication methods and Section 2.2 reviews the existing media streaming methods.

2.1 STREAM AUTHENTICATION TECHNIQUES

A media stream is nothing more than a sequence of media packets and, therefore, stream authentication is nothing more than authenticating a sequence of media packets. However, media stream has its own properties, making it difficult to straightforwardly apply traditional cryptographic methods. First, a media stream potentially has a huge number of packets and, therefore, it cannot afford to have too high complexity or overhead for authentication. A counter example is to apply one digital signature per packet, as illustrated in Figure 1-3(a), which has prohibitively

high complexity and high overhead. Second, a media stream tolerates packet loss up to a certain level and thereby requires the authentication methods to tolerate packets loss as well. In other words, when packet loss occurs in the network, the receiver should be able to verify the received packets with high probability. A counter example is to apply one digital signature for all packets, as illustrated in Figure 1-3(b), where any packet loss will cause all other packets not to be verifiable. Thirdly, a media stream should be verified in a continuous manner. In streaming scenarios like VoD, VoIP, video conference and video broadcast, it is desirable that the packets can be continuously verified as they are delivered to the receiver. The receiver delay measures the gap between the time a packet is received and the time it is verified. A low receiver delay ensures a smooth media play out at the receiver. It is not acceptable that received packets cannot be verified until the last packet is delivered, as with the case in Figure 1-3(b). In addition, for live-encoded media like video conference and VoIP, it is desirable that the sender could continuously sign and transmit the packets generated by the encoder.

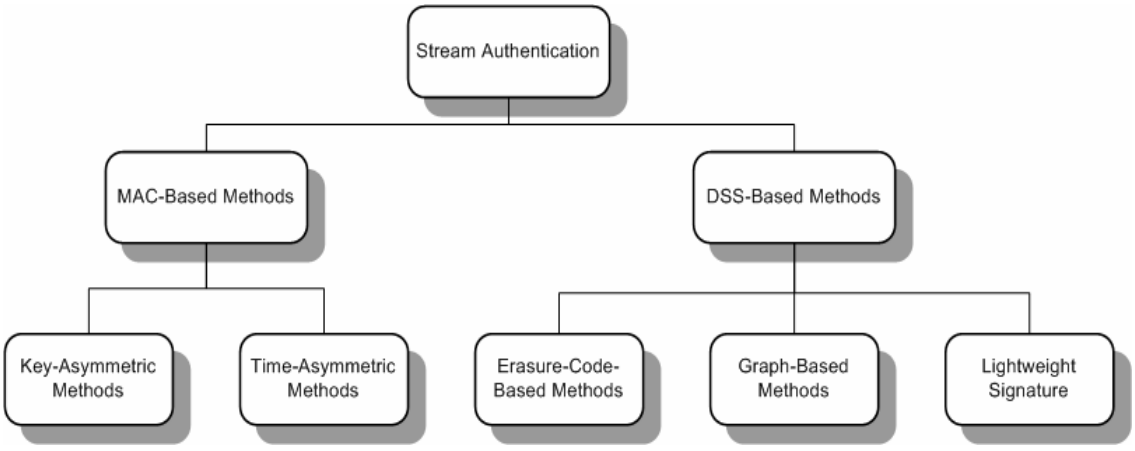

Figure 2-1 - Classification of existing stream authentication methods

Figure 2-1 gives various types of stream authentication methods. Initially, the *MAC-based* Stream Authentication method directly applies MAC to authenticate

stream packets in one-to-one communication scenarios. A common key is shared by the sender and the receiver, where the sender uses the key to generate one MAC per packet and the receiver uses the key to verify the received packet. However, in one-to-many communication scenarios where the common key is shared by all entities, the packet can be modified and the associated MAC can be re-computed by any entity, and therefore it is not possible verify the integrity of the received packet. To tackle this problem, researchers have proposed *key-asymmetric* variants and *time-asymmetric* variants of the MAC-based method. The *key-asymmetric* methods introduce asymmetry in the key material, where the entire key is known to the sender and each receiver has only the partial key material. Therefore, the receivers can verify the packets but cannot generate valid hashes on behalf of the sender. The disadvantage is that the key-asymmetric methods are vulnerable to collusion attack, where a set of receivers collaborate to reconstruct the entire key material. The *time-asymmetric* methods use time as a source of asymmetry, where the sender keeps changing the keys for authentication, and announces a key which is no longer used to generate MACs. In other words, by the time a key is announced to all receivers, it is no long used by the sender and all packets authenticated with it are already delivered to the receivers. However, the time-asymmetric methods raise new security vulnerability related to time synchronization disturbance.

The DSS-based Stream Authentication methods apply digital signature to authenticate stream packets. It works in both one-to-one and one-to-many communication scenarios, due to the asymmetric keys used by DSS. In addition, it simultaneously provides integrity, origin authentication and non-repudiation, while the MAC-based method provides integrity only. The disadvantage is that it has very high complexity and overhead, because a digital signature has much higher

complexity and overhead than a MAC. The *lightweight signature* methods are proposed to reduce the complexity but not the overhead. To reduce both complexity and overhead of the DSS-based methods, researchers has proposed to amortize one digital signature among a group of packets, using some auxiliary data like hashes, recognizing the fact that a hash has much lower complexity and lower overhead than a digital signature. Based on the way a signature is amortized, we can classify the methods into *graph-based* methods and *erasure-code-based* methods. The *graph-based* methods connect packet as a directed acyclic graph using hash chaining, while the *erasure-code-based* methods encode the digital signature and packet hash with erasure code. However, both methods create inter-dependency between the stream packets, i.e., loss of one packet may cause other packets unverifiable. Many methods in this category have been proposed to add redundancy to the authentication data in order to fight against packet loss. More details can be found in Section 2.1.2.

2.1.1 MAC-based Stream Authentication

In a one-to-one communication scenario, MAC can be directly applied to stream packets, i.e., one MAC per packet. This is a very efficient solution, due to the following reasons: 1) MAC is fast to generate and verify, hence low complexity; 2) MAC is small in size, hence low overhead; 3) Each packet individually verifiable, hence high robustness against loss; 4) A received packet can be verified instantly, hence lower receiver delay.

MAC, however, is not suitable for one-to-many communication scenarios, due to the symmetric key used. A naïve solution is to have a separate key between the sender and each receiver, which obviously have many disadvantages like high complexity and high overhead. The sender has to compute one MAC for each

receiver, using the key shared with that receiver. In order to use MAC in one-to-many scenario, it is desirable to have some asymmetric mechanism which allows receivers to verify received data without being able to generate valid authentication on behalf of the sender. There are some existing methods to use MAC in one-to-many scenarios, which can be classified into two categories: *key-asymmetric* methods [51][55][56] [57][58][59][60][61] and *time-asymmetric* methods [62][63][64][33][65][66][67][68].

The key-asymmetric methods introduce asymmetry in the key material used to authenticate data. In particular, the entire key material is known to the sender, and only partial key material is known to a receiver. As such, the receivers can verify the received packets but cannot generate valid MAC on behalf of the sender. For example, in the method proposed by Canetti, *et al* [55], the sender holds a set of l keys and attaches to each packet l MACs – each MAC computed with a different key. Each receiver holds a subset of the l keys and verifies the MAC according to the key it holds. Appropriate choice of subsets ensures that with high probability no coalition of up of w colluding bad receivers know all the keys held by a good receiver, where w is an input parameter. However, since the receivers hold partial key material, the key-asymmetric methods are vulnerable to collusion attack, where a set of receivers collaborate to reconstruct the key material used by the sender.

The time-asymmetric methods use time as a source of asymmetry. For example, in Time Efficient Stream Loss-Tolerant Authentication (TESLA) [33], the receivers are synchronized with the sender's clock and are instructed when to accept a specific key as being used to verify received packets. A key is disclosed to the receivers only after the sender finishes using it to generate MACs. In other words, by the time a key is disclosed, it is no longer used by the sender to generate MACs. Therefore, an attacker is unable to forge a message on half of the senders. The sender

must be very careful about the gap from the time a key is last used for signing to the time the same key is disclosed. If the gap is not sufficiently large, it is possible that a packet is still making its way to the receivers when the key is disclosed, and therefore gives an attacker the opportunity to forge the packet. Note that the time-asymmetric methods raises new security vulnerability related to time synchronization disturbance.

Note that neither key-asymmetric methods nor time-asymmetric methods provide non-repudiation, due to its nature of symmetric key.

2.1.2 DSS-based Stream Authentication

DSS is mature solution for data authentication to simultaneously provide integrity, origin authentication and non-repudiation. Since DSS uses asymmetric key pair, it works in both one-to-one and one-to-many communication scenarios. At any time, the private key, which is used to generate the signature, is known to the sender only. The receivers know the public key only, and therefore they can verify received packets but cannot generate a valid signature on half of the sender. However, DSS has much higher complexity than MAC. According to a benchmarking result [69], a RSA 1024 signature operation takes 6430000 cycles, a RSA 1024 verification operation takes 290000 cycles, and a keyed-hash based message authentication code (or HMAC) has the complexity of 26.4 cycles per byte. That is, for a given packet of 200 bytes, the time required by a RSA 1024 signature operation is 1217 times of that required by a HMAC operation. And similarly, the time required by a RSA 1024 verification operation is 55 times of that required by a HMAC operation. In addition, DSS also has much higher overhead than MAC. For example, a RSA signature is at least 128 bytes while a HMAC is only around 20 bytes.

Chapter 1 describes two simple solutions of using DSS for authenticating stream packets, as illustrated in Figure 1-3. The first solution has very high overhead, high complexity and high verification probability (or robustness), while the second one has very low overhead, low complexity and low verification probability. It is desirable to have a practical solution with reasonable overhead and complexity, where received packets are verifiable with reasonably high probability. Towards this goal, there are two possible approaches: one approach is to propose a lightweight signature with lower complexity and overhead; the other approach is to amortize one digital signature among a group of packets.

There exists a few lightweight signature methods [29][31][32][65]. Gennaro and Rohatgi [31] propose using a one-time signature and one-time public key to authenticate each packet. Although one-time signature has lower complexity, it still has high overhead, which is in the order of several hundred bytes per signature. Later on, Rohatgi [32] proposed using a combination of *k*-time signatures and certificates for the *k*-time public keys (created with a regular signature scheme) to authenticate packets. Despite its improvement over the one-time signature scheme, this method still requires an overhead of the order of several hundred bytes. Wong and Lam [29] proposed extensions to the Feige-Fiat-Shamir digital signature scheme, also known as *eFFS*, to speed up both the signature generation and verification operations. However, the public key and private key in *eFFS* are huge in size. Perrig [65] proposes another one-time signature scheme called *BiBa*, with significantly faster signature and verification operations. However, *BiBa*'s security is dependent on the time synchronization between the sender and the receivers. The authentication rate (number of packet verified per unit time) is limited by the maximum allowable time synchronization error. A *BiBa* signature is still very large, around 100-200 bytes and

the public key is of the order of 10KB, which is exacerbated by the fact that the public key has to be transmitted at regular intervals.

The other approach is to amortize one digital signature among a group of packets, using some auxiliary data like hashes of packets, recognizing the fact that the digital signature has much higher complexity and overhead than hash. According to [69], a RSA 1024 signature operation has a complexity that is 3457 times more than a MD-5 hash operation and a RSA 1024 verification operation has a complexity that is 155 times more than a MD-5 hash operation. Based on the way a signature is amortized, existing DSS-based methods can be further classified into two categories: *Erasure-code-based* and *graph-based* methods, which are explained in more details in Section 2.1.2.1 and Section 2.1.2.2 respectively. Both types of methods try to add redundancy to the authentication data (including signature and hashes), in order to fight against packet loss.

2.1.2.1 Erasure-code-based Stream Authentication

Erasure-code-based methods apply erasure codes to the authentication information (including hash and digital signature) to fight against packet loss. The Signature Amortization based on IDA algorithm (SAIDA) [26][27] proposes to use Information Dispersal Algorithm (IDA) [37], as illustrated in Figure 2-2.

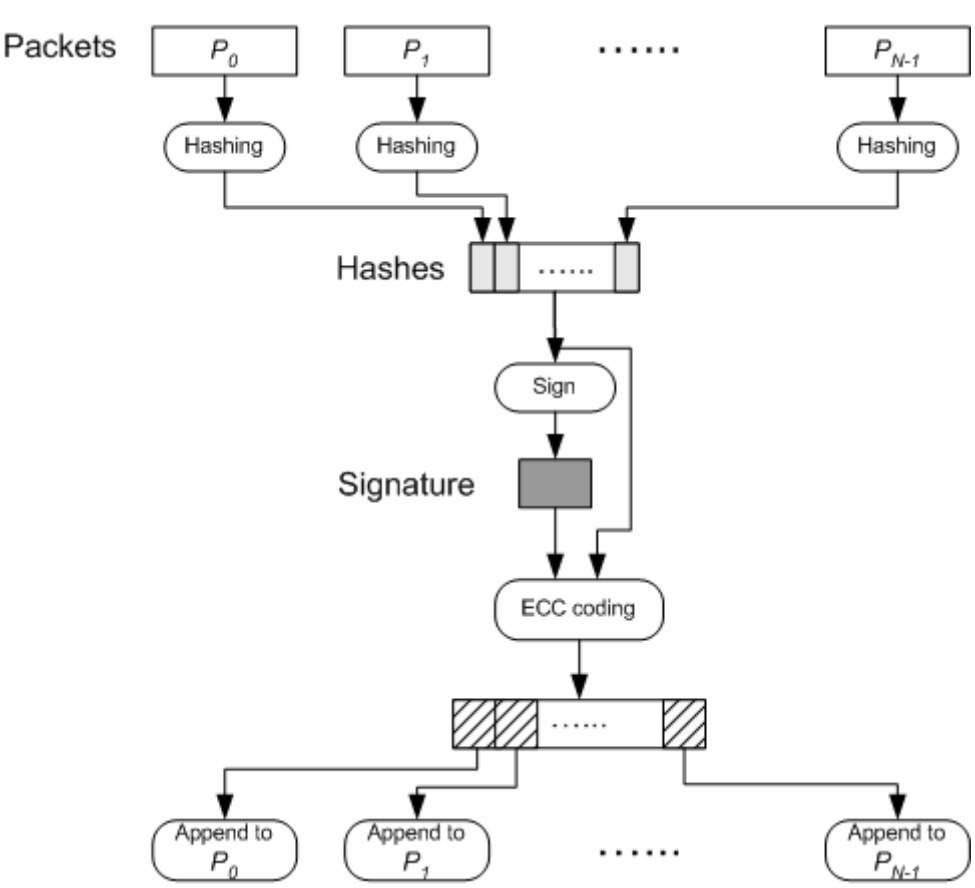

Figure 2-2 – Illustration of Erasure-code-based stream authentication

The Erasure-code-based authentication is applied in the following steps:

- A stream of packets is first divided into blocks. Suppose each block has N packets, namely, P_0, P_1… P_{N-1}. The steps below are performed on every group.
- A hash value $H(P_i)$, where i=0,1…,N-1, is computed for each packet using one-way hash function like SHA-1 or MD-5. The concatenation of all hash values is indicated using $F=H(P_0)||H(P_1)||\ldots||H(P_{N-1})$, where || indicates concatenation operation.
- IDA erasure coding is applied to F, generating N segments, namely, $F_0', F_1', ..., F_{N-1}'$. Out of the N segments, any m segments can be decoded to recover the original bit string F, where m is a pre-determined parameter of erasure coding.

- A digital signature S is generated from the hash of F using a digital signature scheme, i.e., $S=SIGN(H(F))$, where *SIGN(.)* represents digital signing operation.
- The same IDA erasure coding is applied to S, generating N segments, namely, $S'_0, S'_1, ..., S'_{N-1}$. Similarly, any m out of the N segments can be decoded to recover the original digital signature S.
- The hash segment F'_i and signature segment S'_i are appended to packet P_i, where $0 \leq i < N$.

After signing, the packets are transmitted to the receiver, where the signature S and hash string F are recovered as long as at least m packets are received out of N packets in a group. With S and F recovered, the receiver is able to verify all received packets within the group.

The trade-off between verification probability and overhead is governed by the parameter m, which is the minimum number of packets required for recovering the authentication information. A smaller value of m implies higher overhead and high verification probability, and vice versa.

Pannetrat and Molva [28] also proposed a similar method that has three variants: Unbuffered Sender Scheme, Double Buffered Scheme and Single Buffered Scheme. The Unbuffered Sender Scheme piggybacks the hash segments and signature segments of current block onto the packets of the next block. It has no sender delay because a packet can be sent out once its hash value is computed. The receiver delay, however, is equivalent to two blocks of packets, as the received packet cannot be verified until sufficient packets in the next block are received. On the contrary, the Double Buffered Scheme piggybacks the hash segments and signature segments of current block onto the packets in previous block. It has a sender delay equivalent to two blocks and no receiver delay. The Single Buffered Scheme piggybacks the hash

segments and signature segments onto the packets in same block. In this case, both sender delay and receiver delay are equivalent to one block of packets.

Although Erasure-code-based methods are efficient in trading off verification probability and overhead, it has high complexity because erasure coding and decoding are expensive operations. This is especially a concern where the sender or the receivers are small devices like PDA or cell phone.

2.1.2.2 Graph-based Stream Authentication

Gennaro and Rohatgi [31] propose a *Simple Hash Chain* for stream authentication, as illustrated in Figure 2-3. A digital signature is amortized by forming a hash chain. Starting from the last packet, each packet has its hash appended to a previous packet and the first packet is signed using a digital signature. This method has very low overhead, only around one hash per packet on average, but it also has very low verification probability, as any packet loss will break the chain and all subsequent packets are not verifiable. It has a sender delay up to N packets and no receiver delay.

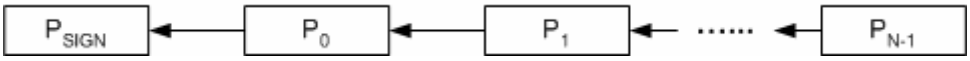

Figure 2-3 – Simple Hash Chain

Perrig, *et al* propose the *Efficient Multi-Chained Stream Signature* (*EMSS*) [33], which is basically an extension to the Simple Hash Chain method [31]. The basic idea is to add more redundant paths to the hash chain such that even in the presence of packet loss, the received packets still have a path to the signature packet with high probability. As illustrated in Figure 2-4, each packet has its hash appended to m (where m=2 in this example) packets that are randomly selected from the immediately following L packets (where $m < L$), and the last packet is signed.

Compared to Simple Hash Chain, EMSS has higher overhead and also higher verification probability. Further, since P_{SIGN} is located at the end of the chain and all edges are pointing forward, EMSS has no sender delay and the receiver delay is up to N packets.

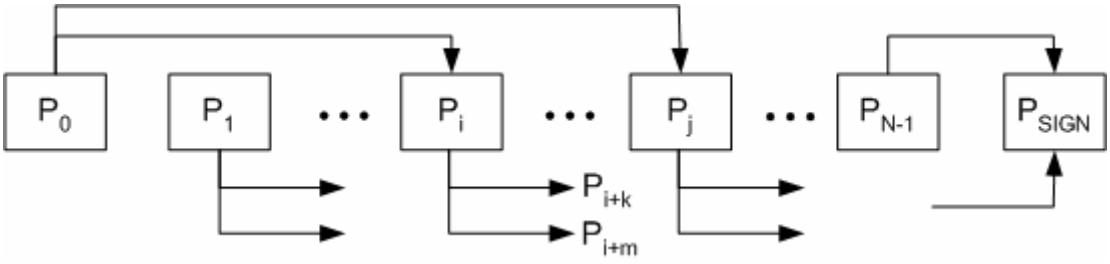

Figure 2-4 – Efficient Multi-Chained Stream Signature (EMSS)

Golle and Modadugu [59] propose another method called *Augmented Chain*, which is a systematic method of putting edges in strategic locations so that the chain can resist a burst loss. Figure 2-5 shows an example of an augmented chain $C_{a,p}$ with a=2 and p=5. There are two types of edges in Augmented Chain, global edges and local edges. The global edges connect the packets whose indices are multiple of p, while the local edges connect the packets in their own locality. It can survive a burst loss of up to $p(a\text{-}1)$ packets in length. The last packet is the signature packet. The overhead is 2 hashes per packet on average. The sender delay is equivalent to p packets and the receiver is equivalent to N packets.

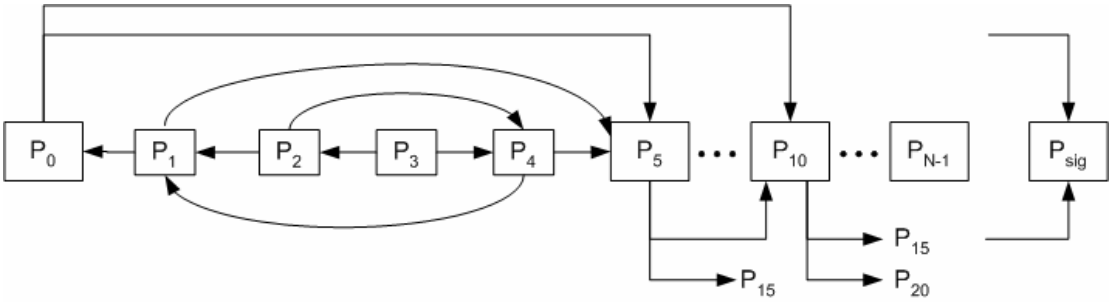

Figure 2-5 – Augmented Chain (a=2 and p=5)

Miner and Staddon [35] propose the *Piggybacking* method similar to Augmented Chain to resist multiple burst losses. In the Piggybacking scheme, N packets are partitioned into r equal-size priority classes. The signature packet is the first packet in the highest priority class. It is assumed that the packets in the highest priority class are spaced evenly throughout the block so that two consecutive packets of the highest priority class are located exactly r packets apart. The Piggybacking method is designed in such a way that packets in the i-th priority class can tolerate burst loss of size at most $k_i r$, where k_i is a parameter dependent on the configuration of the hash chains.

Wong and Lam propose Tree Authentication method [29] that employs Merkle's authentication trees [30]. Figure 2-6 shows a binary authentication tree, where a rectangle represents a packet and a cycle represents a hash value computed from the underlying packet or hash values. A digital signature is generated from the root of the tree. Each packet carries the digital signature and the hash values in the sibling nodes along its path to the root node. Its advantage is that each packet is individually verifiable, i.e., the verification probability is 1. Given N packets, each packet carries one signature and $\log_2 N$ hash values, leading to a very high overhead. It has no receiver delay and the sender delay is equivalent to N packets.

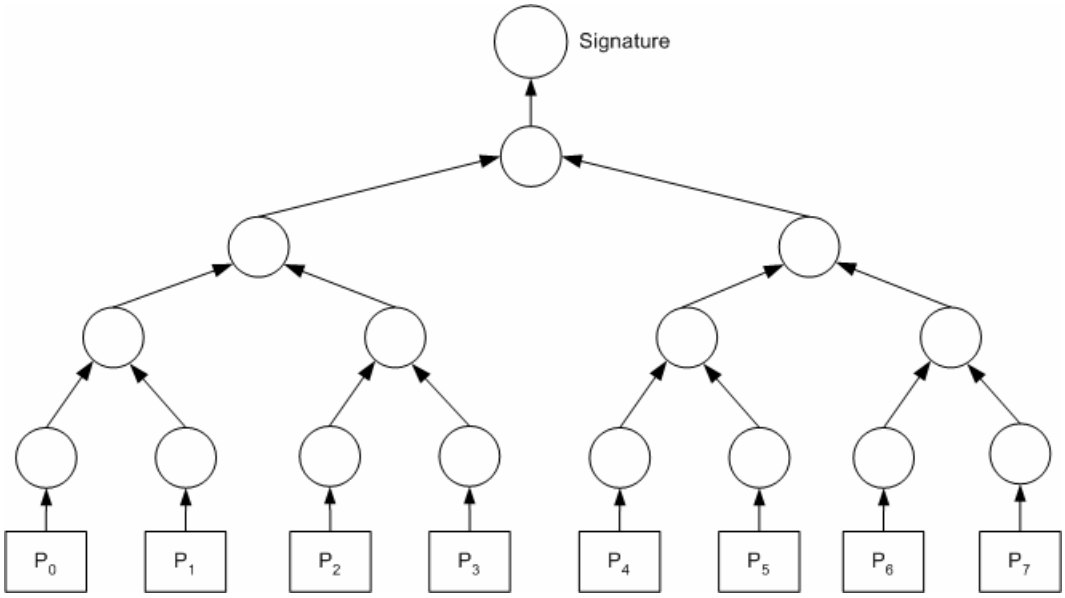

Figure 2-6 – Tree Authentication (degree = 2)

2.2 OPTIMIZED MEDIA STREAMING TECHNIQUES

Media streaming is a technique used to deliver media data (which is usually compressed) to a receiver in real-time over a network. It allows the media data to be transmitted in a continuous stream and played as it arrives. That is, the receiver does not have to wait to download the entire media data before seeing the video and hearing the sound. With advances in media compression and network technologies, media streaming techniques are becoming increasingly more important, which is evident in emerging commercial applications like IPTV, Video-on-Demand, VoIP, Video conference, and so on.

There exist a very diverse range of media streaming applications, with very different operation conditions and properties. For example, media streaming can be for one-to-one or one-to-many (multicast or broadcast) communication scenarios, the media may be pre-encoded (e.g., Video-on-Demand) or encoded in real-time (e.g., VoIP and video conference), the channel may be static or dynamic depending on

whether or not the channel characteristics (e.g., bandwidth, delay and loss rate) change over time, the channel may support QoS or just best-effort transmission. The above conditions impose different technical challenges and different user requirements for media streaming techniques.

Since the raw media data is usually huge in size and occupies large storage space and bandwidth, it is often compressed before being transmitted over a network. There exist various standards (JPEG, JPEG 2000, H.261/3/4, MPEG-1/2/4) or proprietary (RealNetworks [82] and Microsoft Windows Media [83]) media compression techniques. While the compression significantly reduces the size of media data (hence saving storage space and bandwidth), it also makes the streaming media sensitive to network loss. In the compressed domain, some packets are very important and the loss of such important packets will cause drastic negative impact on media quality. For example, in the GOP structure shown in Figure 2-7, a P-frame packet depends on the previous I/P-frame packet, and the B-frame packet depends on both previous and next I/P-frame packets for decoding. In this case, the loss of an I-frame causes all frames not to be decodable within the GOP, while the loss of a B-frame does not affect other frames at all. Some existing methods prioritize the media packets and apply differentiated treatment to the packets using DiffServ [70][71][72], Forward Error Control (FEC) [73][74][75] and Automatic Repeat Request (ARQ) [76][77][78][78].

Furthermore, when network bandwidth is a scarce resource and the source rate is higher than the channel rate, the sender may need to actively prune the media data in order to avoid congestion in the network. Based on the packets' priority, the sender carefully selects which packets to transmit and how to transmit, in order get optimal quality at the receiver. Scalable media compression techniques like JPEG 2000 [2]

and Scalable Video Coding (SVC) [9] are designed to enable simple transcoding with lower complexity. The lower-priority packets are discarded to make the resulting media data match the channel rate.

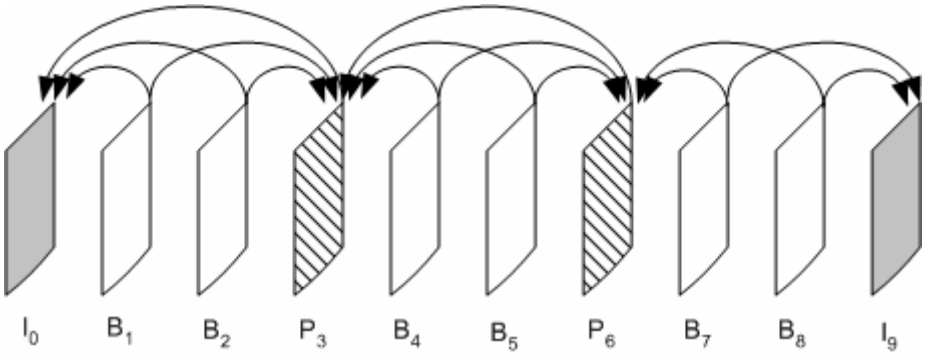

Figure 2-7 – Example of predication dependency between frames in a GOP

A recent advance in media streaming is called *Rate-Distortion Optimized* (*RaDiO*) streaming [38], which computes packet transmission schedule to maximize media quality at the receiver, given the knowledge of packets' importance and network conditions. In other words, the goal of *RaDiO* is to find an optimal transmission schedule that minimizes the expected media distortion for a given constraint on transmission cost, or conversely, minimizes the transmission cost for a given media distortion.

With *RaDiO*, each packet is associated with 3 attributes: size B, deadline T and distortion increment Δd. If a packet is delivered and decoded by its deadline T, the total distortion will be reduced by Δd. A packet delivered after its deadline is discarded or therefore is equivalent to loss. The distortion model is assumed to be additive across the lost packets, i.e., the total distortion caused by multiple lost packets is simply the sum of their distortion increments. When the lost packets are sufficiently far apart, the additive model is quite accurate. However, in the case of bursty loss, the additive model is no longer accurate, as detailed in [80]. In the rest of

this thesis, we assume an additive distortion model because of its simplicity and lower complexity.

The transmission policy for a single packet P_n is represented by π_n, which varies with the underlying network. For instance, in a network with DiffServ services [81], the policy π_n is simply the forwarding service (Guaranteed Service, Assured Forwarding, or Best-Effort) chosen to transmit packet P_n; when FEC is used to protect packet from loss, the policy π_n refers to the amount of redundancy added to packet P_n; when ARQ is used to re-transmit lost packet, the policy π_n refers to the number of (re-)transmissions for packet P_n. Associated with the policy π_n are a cost per byte $\rho(\pi_n)$ for transmitting P_n and an error probability $\varepsilon(\pi_n)$ of not delivering the packet by its deadline T_n. To transmit a single packet P_n, the optimal performance can be achieved by selecting the transmission policy π_n to minimize the Lagrangian in Eq(2.1) where λ_n is positive value.

$$\varepsilon(\pi_n) + \lambda_n \rho(\pi_n) \tag{2.1}$$

The transmission policy for N packets is represented by a policy vector $\pi = [\pi_0, \pi_1, ...\pi_n, ...\pi_{N-1}]$, where π_n is the transmission policy of packet P_n. The total expected transmission cost is the sum of the transmission cost of all N packets, as in Eq(2.2). In turn, the transmission cost for a packet P_n is the cost per byte $\rho(\pi_n)$ times the packet size B_n. The term $R(\pi)$ represents the total expected cost for transmitting all N packets using policy π.

$$R(\pi) = \sum_{n=0}^{N-1} B_n \rho(\pi_n) \tag{2.2}$$

The total expected distortion for N packets is more complicated to express. It is assumed that a packet P_n may depend on some other packets for decoding. For example in Figure 2-7, P_6 depends on I_0 and P_3; B_5 depends on I_0, P_3 and P_6. We use $\phi(P_n)$ to represent the set of packets needed by packet P_n for decoding. For easy explanation, P_n is also included in $\phi(P_n)$, i.e., $P_n \in \phi(P_n)$. Therefore, the probability that a packet is decodable can be expressed with $\prod_{P_{n'} \in \phi(P_n)} (1-\varepsilon(\pi_{n'}))$. At the receiver side, if a packet is delivered and decoded before its deadline, the total distortion is reduced by Δd_n; otherwise, the distortion is not reduced. Assuming that the total distortion when no packet is received is D_0 and distortion is additive, the expected distortion of the reconstructed media at the receiver can be expressed with Eq(2.3).

$$D(\pi) = D_0 - \sum_{n=0}^{N-1} \Delta d_n \prod_{P_{n'} \in \phi(P_n)} (1-\varepsilon(\pi_{n'})) \tag{2.3}$$

With Eq(2.2) and Eq(2.3) for the expected transmission cost and expected distortion, the policy vector π can be optimized to minimize the expected distortion subject to a constraint on the expected transmission cost. The problem can be solved by finding the policy vector π that minimizes the expected Lagrangian in Eq(2.4), for a given $\lambda > 0$.

$$\begin{aligned} J(\pi) &= D(\pi) + \lambda R(\pi) \\ &= D_0 + \sum_{n=0}^{N-1} \left[\Delta d_n \left(- \prod_{P_{n'} \in \phi(P_n)} (1-\varepsilon(\pi_{n'})) \right) + \lambda B_n \rho(\pi_n) \right] \end{aligned} \tag{2.4}$$

However, it is not an easy problem to minimize the expected Lagrangian in Eq(2.4). The probability that a packet P_n is decodable depends on not only its own policy π_n but also the policy of other packets in $\phi(P_n)$. As such, the expected

distortion cannot be split into a sum of terms, each depending on only a single policy π_n, and the minimization problem in Eq(2.4) is very complicated.

An iterative descent algorithm is proposed to solve the minimization problem. The objective function $J(\pi)$ minimizes one policy variable at a time, keeping the other variables constant. This is repeated until the objective function converges. At each step, the minimization problem is actually transformed into an error-cost minimization problem for a single packet similar to Eq(2.1). For instance, for packet P_n, its optimized policy π_n^* can be searched using Eq(2.5), where S_n is the sensitivity factor of packet P_n, which is the amount by which the total expected distortion will increase if the packet P_n cannot be received by its deadline, given the current transmission policies for all other packets.

$$\pi_n^* = \arg\min_{\pi_n} \varepsilon(\pi_n) + \frac{\lambda B_n}{S_n} \rho(\pi_n) \tag{2.5}$$

Note that *RaDiO* achieves very good performance at the price of increased computational complexity. Searching for an optimized transmission policy is computationally expensive, due to the coding dependency among the packets. The iterative descent algorithm has to run for many iterations before the objective function converges. Therefore, *RaDiO* is inappropriate for on-line optimized video streaming.

Chakareski, *et al* [39][40] propose a low-complexity rate-distortion optimized streaming method. Specifically, an R-D hint track (RDHT) is generated when media is encoded, which contains side information to facilitate high quality streaming with low complexity.

The *RaDiO* framework is also extended in more advanced streaming scenarios, such as streaming over multiple paths [41][42], distributed streaming from multiple servers [43][44], streaming via network edge [45][46], streaming with

multiple deadlines [47], streaming with rich acknowledges [48] and with improved rate and distortion model[49][50].

CHAPTER 3 - STREAM AUTHENTICATION BASED ON BUTTERFLY GRAPH

Graph-based stream authentication amortizes a digital signature among a group of *N* packets, by connecting them as a DAG. Since only one signature is generated for all *N* packets, a graph-based method has much lower complexity and overhead. However, it also introduces dependency among the packets, i.e., loss of one packet may cause some other packets not to be verifiable, and thereby equivalent to a loss. At the receiver, lost packets are removed from the graph and a received packet is verifiable only if it has at least a path to the signature packet. That is to say, each received packet is verifiable with a probability. For instance, with Simple Hash Chain [31], if a packet is lost, all subsequent packets have no path to the signature packet and therefore are not verifiable. In order to increase the verification probability, more redundant paths have to be added such that a received packet has a path to the signature packet with higher probability. EMSS [33], Augmented Chain [34] and Piggyback [35] follow this direction of adding redundant paths.

While adding redundant paths increases the verification probability, it also increases the overhead, because each edge in the graph corresponds to the overhead of one hash value. Therefore, a trade-off exists between the overhead and verification probability. It is desirable to have an optimal balance point such that the verification probability is maximized for a given overhead, or conversely the overhead is

minimized for a given verification probability. Therefore, the key problem is how to construct an authentication graph that is able to achieve the optimal balance.

Butterfly graph is commonly used in the design of communication networks for parallel and distributed systems, due to the following properties: 1) Good fault-tolerance: A node has a high probability of being connected in the presence of faulty nodes in the graph; 2) The maximum fan-in and fan-out degree is 2, i.e., a node has up to 2 incoming edges and up to 2 outgoing edges. For graph-based stream authentication, the fault-tolerance of the graph is equivalent to verification probability, and the fan-in and fan-out degrees are equivalent to overhead. This naturally motivates us to apply butterfly graph for stream authentication, known as *Butterfly Authentication*. We also experimentally demonstrate that the Butterfly Authentication method has higher verification probability than existing methods when the overhead is fixed at around 2 hashes per packet. More details of Butterfly Authentication can be found in Section 3.1.

Nevertheless, Butterfly Authentication has its own limitations: (1) the total number of packets is not flexible; (2) fixed overhead due to its fixed topology; (3) the signature packet size grows with the total number of packets. If the number of packets is large, the signature packet may be bigger than the network *MTU* and, therefore, is fragmented for transmission, which increases its loss probability and negatively impacts verification probability of all other packets in the graph. To overcome the abovementioned limitations, we propose the *Generalized Butterfly Graph* (*GBG*) in Section 3.2. The GBG graph supports an arbitrary number of packets and arbitrary overhead. Further, with the GBG graph the signature packet can be confined within one *MTU*, which implies that it does not have to be fragmented for transmission. The GBG graph supports a wide range of possible authentication graphs, and the problem

of finding the best authentication graph for a given situation corresponds to a graph design problem. Simulation results demonstrate that the GBG authentication method outperform the other graph-based methods.

3.1 BUTTERFLY AUTHENTICATION

Stream packets are divided into groups of N packets, and only one signature is generated for each group. The N packets and the extra signature packet are connected as a butterfly graph, assuming $N=N_R(\log_2 N_R+1)$ and N_R is the number of rows in the butterfly graph. The Butterfly authentication graph is defined as follows:

Definition: A butterfly authentication graph is a DAG containing one signature packet P_{SIGN} and $N= N_R(\log_2 N_R+1)$ packets. The N data packets are arranged into a matrix with N_R rows and N_C columns, where $N= N_R N_C$ and $N_C=\log_2 N_R+1$. In the original sequence, the n-th packet is represented by P_n, where $0\le n<N$. In the butterfly graph, the packet P_n is also represented with $P_{c,r}$, where $n=cN_R+r$, $0\le c<N_C$ and $0\le r<N_R$. The first subscript c is the column index while the second subscript r is the row index. In the butterfly graph, there is an edge from packet $P_{c,r}$ to packet $P_{c',r'}$ if either of the following conditions is satisfied:

1. $c = c' + 1 \quad and \quad r = r'$
2. $c = c' + 1$ and r' is different from r only at c'-th most significant bit.

In addition, each packet in the first column has an edge to the signature packet, i.e., the signature packet P_{SIGN} contains the hashes of all packets in the first column and a digital signature generated from the hashes. Recall that an edge from packet A to packet B is realized by appending A's hash to B.

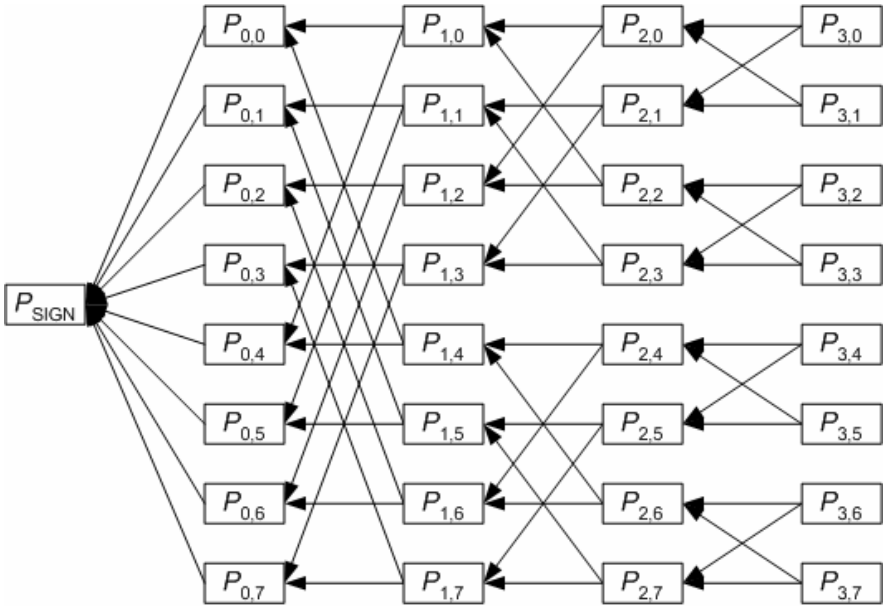

Figure 3-1 – An example butterfly authentication graph

Figure 3-1 gives an example of the butterfly authentication graph, with 4 columns and 8 rows. The signature packet P_{SIGN} contains the signature and hashes of all packets in the first column. All packets in column 0 to N_C-1 containing two hashes, corresponding to two incoming edges, and the packets in the last column do not have any hash. Assuming that the hash size is h bytes and the signature size is g bytes, the average overhead per packet O_{avg} can be expressed with Eq (3.1).

$$2h + \frac{g}{N} - \frac{h}{N_C}, where \quad N_C = \log_2 N_R + 1 \tag{3.1}$$

The computational complexity is also very low, including 1 signature operation and N hashing operations at the sender. At the receiver, the complexity is 1 verification operation and N hashing operations.

Since the signature packet P_{SIGN} carries the digital signature, loss of P_{SIGN} will cause all other packets not verifiable. Thus, the signature packet has to be delivered using mechanisms like multiple repeated transmissions, ARQ or FEC. For the moment, we assume that P_{SIGN} is always received. The packet loss probability ε

follows an identical and independent distribution, i.e., random packet loss. Since a packet $P_{c,r}$ is connected to $P_{c-1,r}$ and $P_{c-1,r'}$ (where r and r' differ at (c-1)-th most significant bit), $P_{c,r}$ is verifiable if either $P_{c-1,r}$ or $P_{c-1,r'}$ is verifiable and received. Further, the butterfly graph has a nice "*independency property*", i.e., a packet is connected to two packets that are independent of each other for verification. This property maximizes the packet's verification probability. Therefore, the verification probability $V_{c,r}$ can be expressed with Eq (3.2).

$$V_{c,r} = \begin{cases} 1 & when \quad c = 0 \\ V_{c-1,r}\varepsilon + V_{c-1,r'}\varepsilon - V_{c-1,r}V_{c-1,r'}\varepsilon^2 & when \quad c > 0 \end{cases} \tag{3.2}$$

From Eq(3.2), we can see that the verification probability is initially 1 for packets in the first column and is decreasing as we go from the first column to the last column. This is because the packets in a later column are further away from the signature packet and they have more dependency than the packets in earlier column. Nevertheless, the decreasing trend is slowed down by the factor that a packet in later column has more redundant paths to the signature packet. For example, a packet in the second column has 2 alternative paths to the signature packets P_{SIGN} and each path has a distance of 2 edges, while a packet in the third column has 4 alternative paths to P_{SIGN} and each path has a distance of 3 edges. As a result, the verification probability has a noticeable drop-off for the first few columns only, and thereafter remains constant, as illustrated in Figure 3-2. Note that all packets in the same column have the same verification probability.

Regarding the burst packet loss, the butterfly authentication method is able to resist up to $N_R/2^{c+1}$ consecutive packet losses at column c, where $0 \le c < N_C$. In the example graph depicted in Figure 3-1, it can resist a burst loss of up to 4 packets and 2 packets in column 0 and column 1, respectively.

Note that the Butterfly Authentication method also has a sender delay equivalent to N packets, because the sender has to compute the hashes and the signature before sending the first packet. For live-encoded media, the long sender delay may not be acceptable. However, for pre-encoded media, sender delay is not important as the authentication can be done off-line. Since the signature packet is located at the beginning and all edges point backward, the Butterfly Authentication method does not have any receiver delay, i.e., a packet can be verified immediately after it is received. Therefore, the receiver does not have to maintain a buffer to store the packets that have been received but not yet verified. The above analysis of receiver delay is based on the assumption that packets are delivered in the same order as they are transmitted by the sender. However, if the packets are delivered out-of-order, it is possible that a received packet is not immediately verifiable, and therefore a buffer is required to temporarily store this packet until it becomes verifiable. The buffer size varies with the networks.

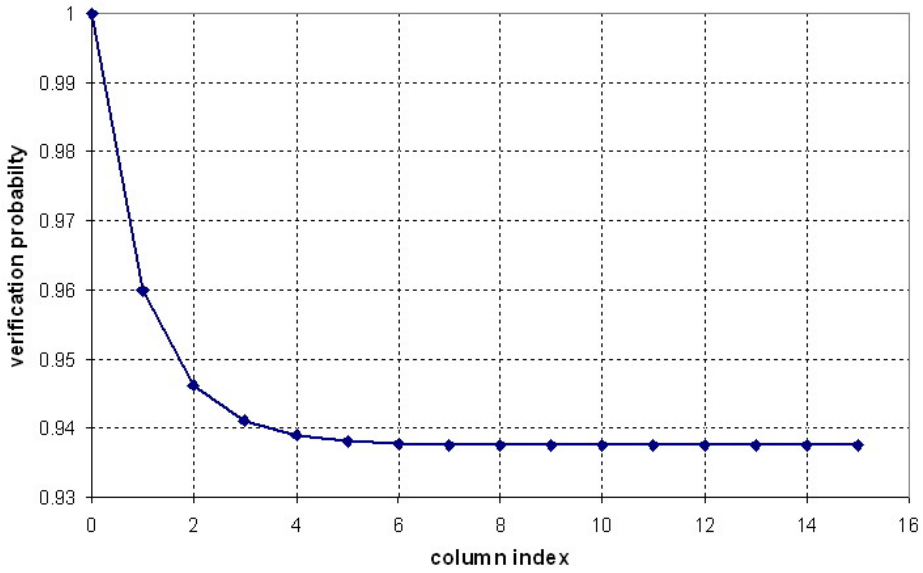

Figure 3-2 – Verification probability at different columns of a butterfly graph (ε=0.2)

3.1.1 Performance Evaluation

In this section, we compare the Butterfly Authentication method with other existing methods using the evaluation criteria defined in Section 1.2.5. The comparison is summarized in Table 3-1. The results are obtained with the following settings and assumptions:

- A digital signature is amortized by a group of N packets, hash size is h bytes and signature size is g bytes.
- The Tree-Authentication in [29] has a degree of 2, i.e., each internal node has two children nodes.
- For EMSS [33], each packet has l outgoing edges connecting to the packets that are randomly selected from previous L packets.
- For Augmented Chain [34], the term a is the maximum distance between two nodes that are connected with a global edge, and p is the size of packet buffer at the sender.
- For Butterfly Authentication, it is assumed that $N=N_R(\log_2 N_R+1)$.

Table 3-1 – Comparison of various graph-based authentication methods

Methods	**Complexity**	**Overhead (bytes/packet)**	**Sender delay**	**Receiver delay**	**Verification probability**
Tree-authentication [29]	1 Signature $2N$-1 hashing	$g+h\log_2 N$	N	1	very high
Simple Hash Chain [31]	1 Signature N hashing	$g/N+h$	N	1	Very low
EMSS [33]	1 Signature N hashing	*Variable*	1	N	High
Augmented Chain [34]	1 Signature N hashing	$g/N+2h$	p	N	High
Butterfly	1 Signature	$g/N+2h$-	N	1	High

Authentication	N hashing	$h/(\log_2 N_R+1)$			

The methods listed in Table 3-1 have roughly the same complexity, including 1 signature/verification operation and N hashing operations. The only exception is the Tree-Authentication method [29], which has to perform N-1 extra hashing operations than the other methods.

In terms of overhead, the Tree-Authentication method [29] and Simple Hash Chain [31] are two extreme cases, where the former has very high overhead (each packet carries a signature and $\log_2 N$ hashes) and the latter has very low overhead (the first packet carries a signature and all other packets carry a single hash). EMSS [33] has variable overhead, and Augmented Chain [34] has around 2 hashes per packets. The Butterfly Authentication method has overhead close to Augmented Chain.

The Tree-Authentication [29], Simple Hash Chain [31] and the proposed Butterfly Authentication have a sender delay equivalent to N packets and a receiver delay equivalent to 1 packet. Therefore, the sender has to maintain a buffer of N packets to store the packets that are generated by a media encoder but not yet ready for transmission. For pre-encoded media, the authentication process can be done offline at the sender side, and hence sender delay is not a problem. The EMSS [33] and Augmented Chain [34] have higher receiver delay of N packets, which implies that the receiver has to maintain a big buffer to store the packets that have been received but not yet verified. In addition, the high receiver delay might negatively impact the smoothness of media play out.

To evaluate the performance of the Butterfly Authentication method, we conducted simulations using the following settings: 1) N=1024; 2) h=16 bytes; 3) g=128 bytes; and 4) the signature packet is always received; (5) Packet loss rate ε ranges from 0 to 0.5 [88]. In the simulation experiment, EMSS and Augmented Chain

are implemented with their optimal setting. For instance, in the EMSS graph, the distance of two endpoints of an edge is uniformly distributed in the interval [1, 128]. For the Augmented Chain, a=15 and p=7, which are selected by the original authors in [34]. Figure 3-3 shows the verification probability at different overheads, by fixing packet loss rate at 0.3. We can see that the Butterfly Authentication method outperforms the other methods under the same overhead.

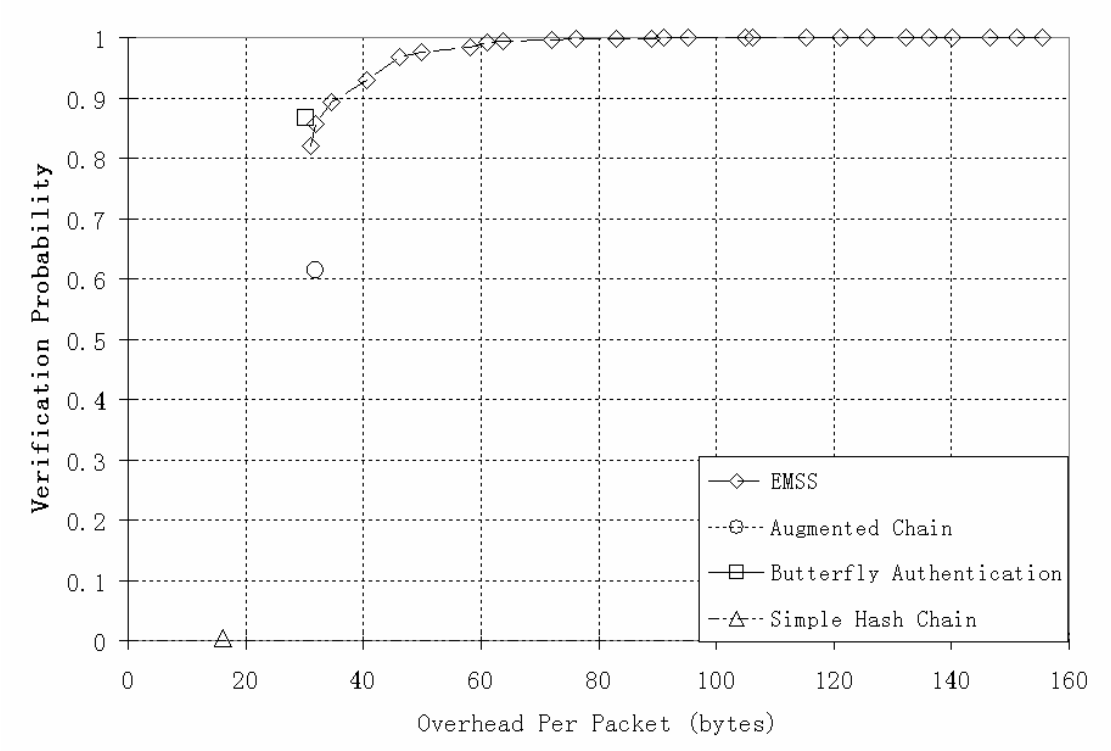

Figure 3-3 – Verification probability at various overheads (Packet loss rate = 0.3)

Figure 3-4 shows the verification probability at various packet loss rates (overhead is fixed at 32 bytes per packet), which demonstrates that the Butterfly Authentication method outperforms the other methods at all loss rates. In addition, the performance gap between Butterfly Authentication and the other methods grows with the packet loss rates.

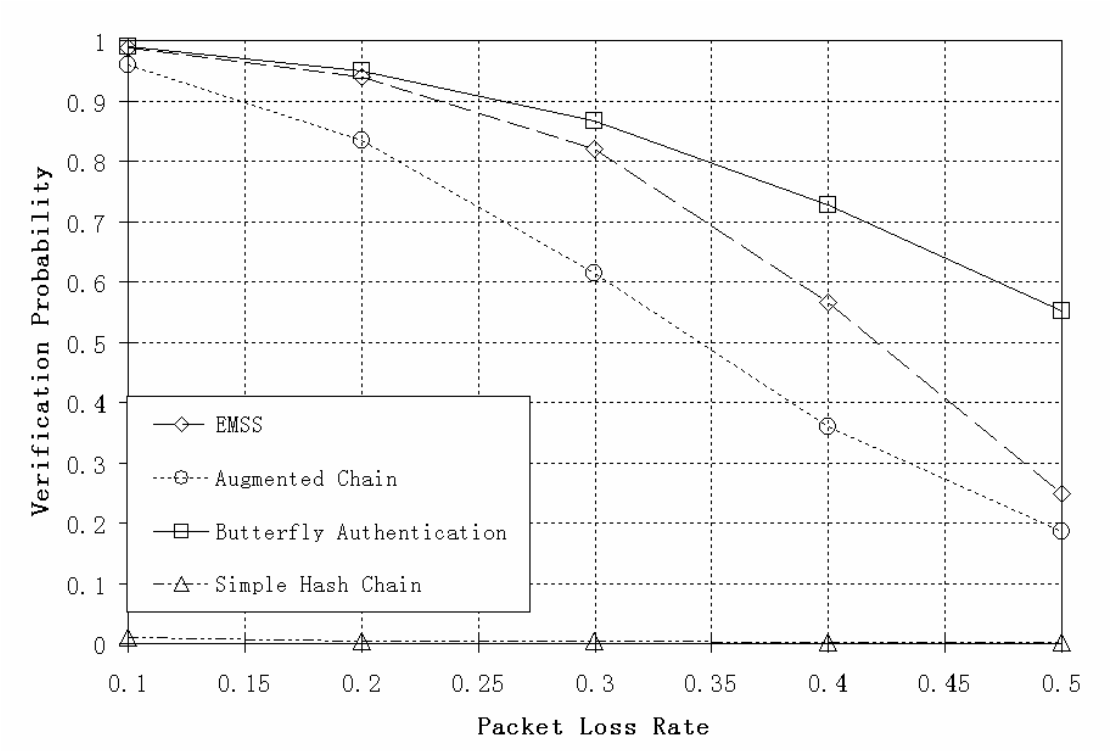

Figure 3-4 – Verification probability at various packet loss rates (Overhead is 32 bytes per packet)

3.2 GENERALIZED BUTTERFLY GRAPH AUTHENTICATION

Although the Butterfly Authentication method outperforms the existing methods, it has three limitations. First, the total number of packets is not flexible, i.e., $N=N_R(\log_2 N_R+1)$, where N_R is the number rows in the butterfly graph. Second, the overhead is fixed at around 2 hashes per packet. In certain situations, we may want to have higher verification probability at the expense of higher overhead, which cannot be supported by the Butterfly Authentication method. Thirdly, the signature packet grows with N, as it contains the hashes of all packets in the first column. For large N the signature packet may be larger than the network *MTU* and therefore has to be fragmented into several datagrams for network transmission. The signature packet is considered lost if any of its datagrams is lost, which increases its loss probability and

negatively impacts all the packets. Recall that loss of signature packet causes all packets not to be verifiable in the graph.

In this section, we try to preserve the Butterfly graph's valuable fault-tolerance property, while extending it to overcome the above 3 limitations, and refer to this framework as the *Generalized Butterfly Graph* (*GBG*). The GBG framework supports a wide range of possible authentication graphs, and the problem of finding the best authentication graph for a given situation corresponds to a graph design problem. The input parameters include the total number of packets N, the packet loss rate ε, and the overhead budget R_O. The output parameters include the number of rows and columns (N_R & N_C), edge placement, and number of transmissions M of the signature packet. The signature packet may be transmitted multiple times to reduce the probability of loss. R_O accounts for the single or multiple transmissions of the signature packet and overhead data corresponding to edges in the graph. The evaluation metric is verification probability. In addition, we also propose a new evaluation metric called the *Loss-Amplification-Factor (LAF)*, defined as the ratio of effective loss rate over packet loss rate. The effective loss rate accounts for both lost packets and packets which are received but unverifiable, i.e., $\varepsilon+(1-\varepsilon)(1-V)$, where ε is packet loss rate and V is the verification percentage. For example, a value of 1.5 means the authentication technique causes the effective loss rate to be 50% higher than that when no authentication is used. Ideally, the LAF would equal 1, i.e., all received packets are verified. The closer the LAF for an authentication technique is to 1, the greater the robustness is against network losses.

In the subsequent sections, we first analyze the butterfly authentication graph to find a cost-effective edge placement strategy given an overhead budget R_O, as detailed in Section 3.2.1. After that we examine in Section 3.2.2 how to relax the

butterfly structure to support an arbitrary number of packets and to confine the signature packet within an *MTU*. Based on this analysis, we propose in Section 3.2.3 the GBG framework and describe how to design an authentication graph for a given situation. In Section 3.2.4 we evaluate the proposed GBG authentication method against existing methods.

3.2.1 Analysis of Butterfly: Edge Placement

The butterfly graph has a fixed topology and therefore fixed overhead. This section studies the problem of how to place the edges in the graph given an arbitrary overhead budget R_O. To gain more insight into this problem, we implement two greedy algorithms to progressively add edges (one at a time) to the authentication graph, starting from the initial state depicted in Figure 3-5. Initially, the packets are arranged into the same matrix as in a butterfly and each packet is connected to only one packet in the previous column. Then, at every step, the greedy algorithm computes the overall verification probability for all possible new edges and adds the edge that provides the largest gain. When there are multiple candidate edges with the same gain, it randomly picks one such candidate. The first greedy algorithm, referred to as *unconstrained greedy*, allows an edge from one packet to any packet in the previous column. The second algorithm, referred to as *constrained greedy*, allows only the edges that appear in the original Butterfly. Figure 3-6 gives one example graph after 24 edges are added by unconstrained greedy algorithm, and Figure 3-1 gives the graph after 24 edges are added by constrained greedy algorithm.

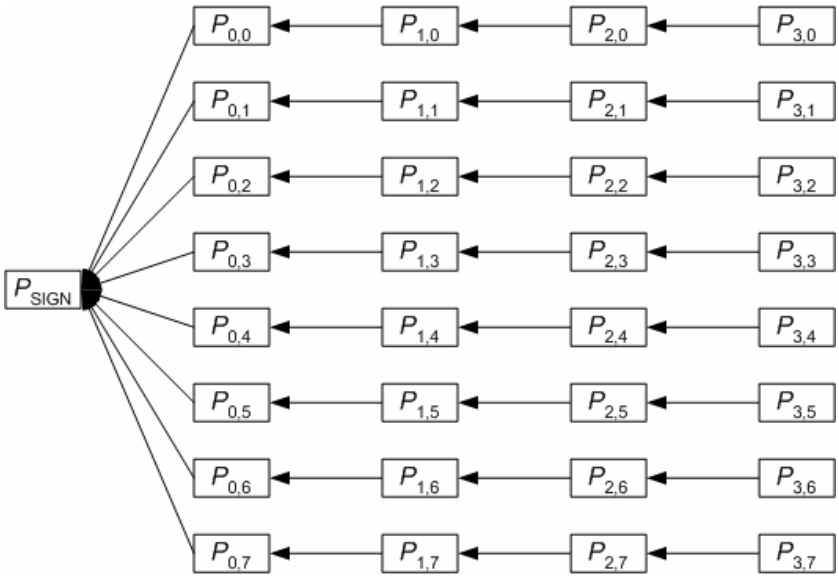

Figure 3-5 – Initial state of greedy algorithms (with 32 packets)

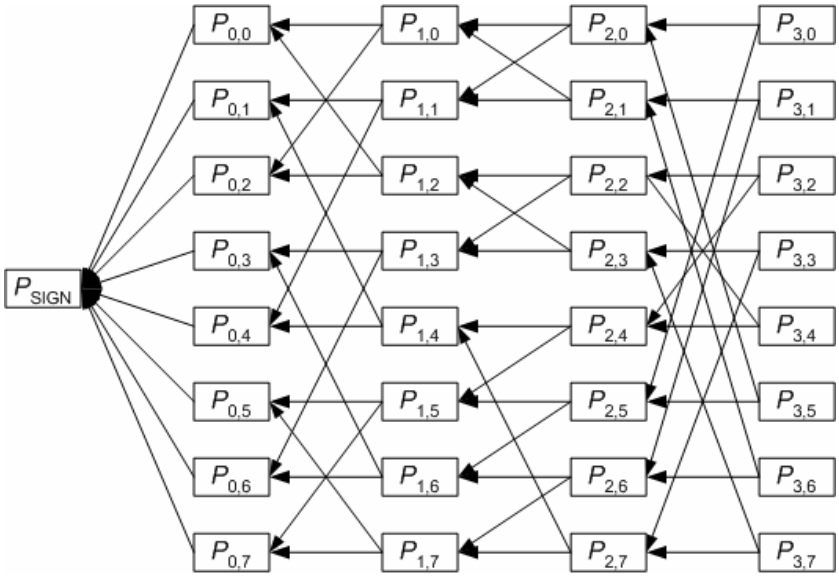

Figure 3-6 – A resulting graph after 24 edges are added by greedy algorithm (without butterfly constraint)

Figure 3-7 compares the LAF of the two greedy algorithms with 32 packets (N_R=8 and N_C=4), where 24 extra edges are progressively added, starting from the initial state shown in Figure 3-5. One observation is that after 24 edges are added, each packet (except the packet in the first column) has exactly 2 outgoing edges with both *unconstrained* and *constrained* greedy algorithms. Another observation is that a

packet's verification probability is maximized if it is connected to two packets that are independent of each other for verification (i.e., independence property, as observed from the butterfly graph). The unconstrained greedy algorithm has lower LAF when the number of added edge $\alpha \leq 15$, while the constrained greedy algorithm has slightly lower LAF when $\alpha > 15$. When α is small, the unconstrained greedy algorithm allows more than 2 packets to be connected to a packet with higher verification probability, without violating the independence property. When α is large, it is no longer possible for the unconstrained greedy algorithm to find edges that satisfy the independence property, while the constrained greedy algorithm can still find such edges. Therefore, the butterfly-constraint greedy algorithm produces slightly worse performance in the middle and slightly better performance at the end. We will apply this butterfly constraint for edge placement when overhead budget is not more than 2 hashes per packet.

We also observe that the greedy algorithm adds the edges in middle column first, and then progressively adds to the outer columns. In summary, when the given overhead budget is less than 2 edges per packet, the edges placement process should follow the butterfly constraint and the order of placement should be from middle column to outer columns.

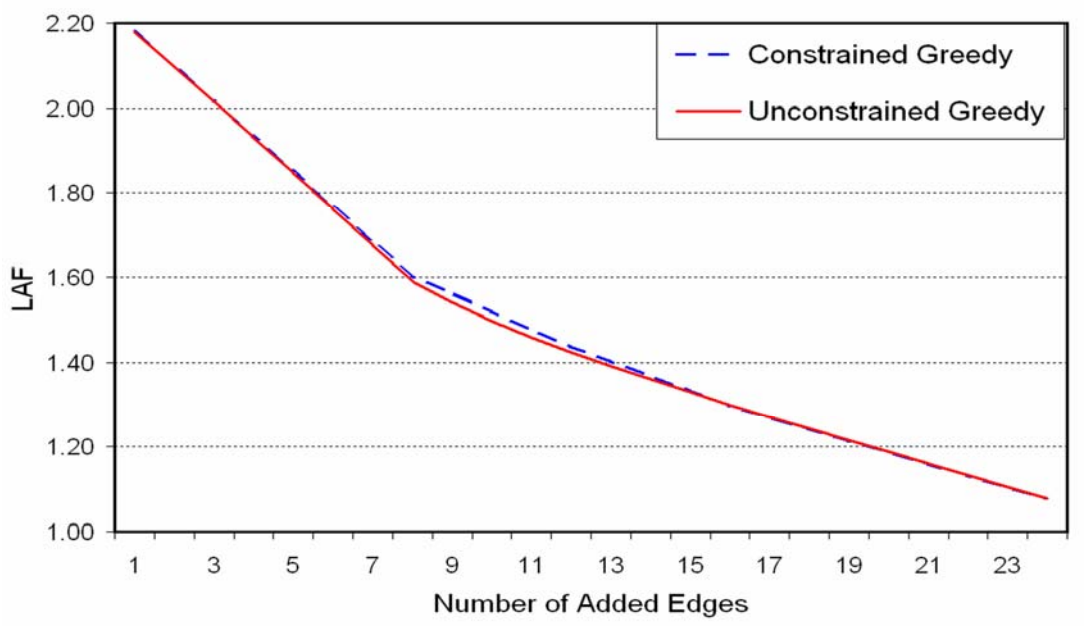

Figure 3-7 – LAF of graphs built with unconstrained and constrained greedy algorithm

The above experiment demonstrates that the butterfly graph provides the best performance when the overhead budget is exactly 2 hashes per packet (i.e., the same overhead to butterfly authentication). Next, we examine the problem of finding the best edge placement order beyond 2 edges per packet. In a butterfly graph, adding one edge originating from a packet $P_{c,r}$ has two benefits: one is the increased verification probability of $P_{c,r}$ itself, and the other is the increased verification probability of its dependent packets. For example, in Figure 3-1, if an edge is added from packet $P_{1,0}$ to $P_{0,1}$, the first benefit is the increased verification probability of $P_{1,0}$ and the second benefit is the increased verification probability of $P_{2,0}$, $P_{2,2}$, $P_{3,0}$, $P_{3,1}$, $P_{3,2}$ and $P_{3,3}$.

Figure 3-8 shows the increment of verification probability of $P_{c,r}$ if an edge originating from $P_{c,r}$ is added to the butterfly graph. We can see that the increment grows with the column index, i.e., the higher the column index c, the greater the increment is. However, the increment converges when the column index c is greater than 4. The reason is that the packets in earlier columns already have very high verification probability and adding one more edge does not help much.

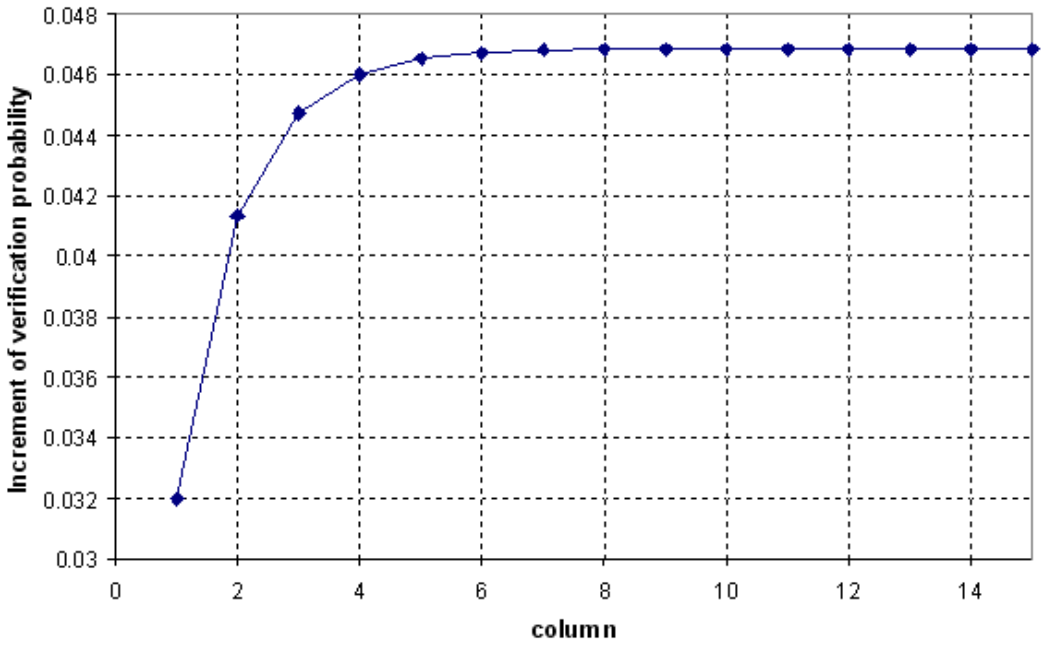

Figure 3-8 – Increment of verification probability of $P_{c,r}$ versus the column index c (adding one edge originating from $P_{c,r}$, ε=0.2)

Figure 3-9 shows the increment of verification probability for the dependent packets of $P_{1,r}$ whose verification probability is increased by 0.05. We can see that the packet has noticeable impact on its dependent packets in the next two columns only. For the dependent packets that are more than 2 columns away from $P_{1,r}$, the impact is negligible. Similar results are observed when the verification probability of packets in other columns is changed.

Combining Figure 3-8 and Figure 3-9, we can compute the overall benefit of adding an edge to a butterfly, as shown in Figure 3-10. The maximum benefit is obtained when the edge is placed in the middle column. The overall benefit is smallest for the first and last column. Packets in the last column do not have any dependent packets, so adding to the last column cannot enjoy the second benefit. Although packets in the first column have many dependent packets, adding one edge does not help a lot for their own verification probability. Therefore, the edges should be added progressively from middle column to outer columns. For example, in a butterfly with

17 columns, the best order should be column-9, 8, 10, 7, 11, 6, 12, 5, 13, 4, 14, 3, 15, 2, 1, and 16. Note that the benefits for columns from 4 to 13 are almost identical.

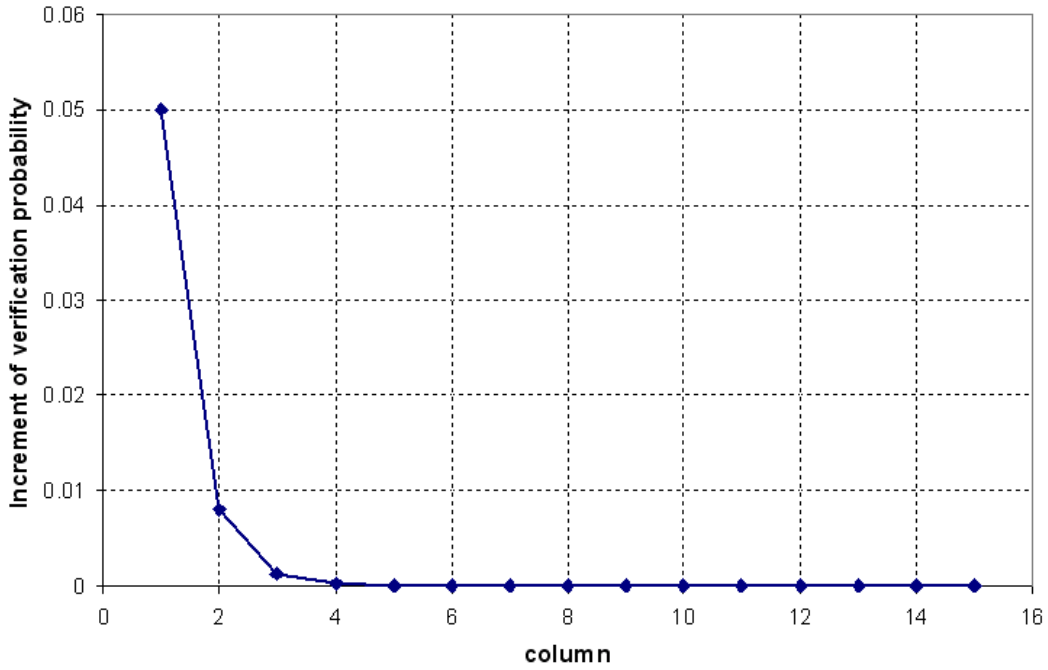

Figure 3-9 – Increment of verification probability for the dependent packets of a column-1 packet $P_{1,r}$ whose verification probability is increased by 0.05 (ε=0.2)

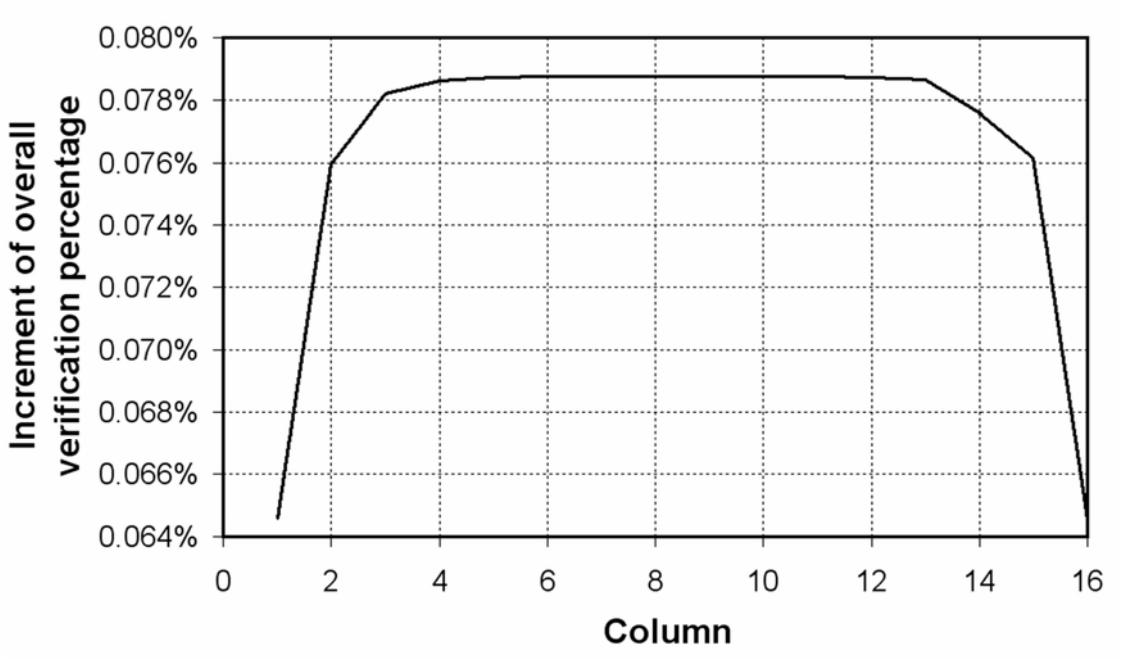

Figure 3-10 – Increment in overall verification percentage when 1 edge is added to different columns of a butterfly with 17 columns.

3.2.2 Relaxing Butterfly Structure

The goal of this section is to overcome the second and third limitations of the Butterfly Authentication graph. We examine how to relax the Butterfly structure to (a) support an arbitrary number of packets *N,* and (b) to confine the signature packet to within one *MTU*.

In the original Butterfly graph, the signature packet grows with the total number of packets *N*, as it contains the digital signature and the hashes of all packets in the first column. If the signature packet is too big to fit in one *MTU*, it has to be fragmented for transmission, which increases its loss probability and negatively impacts the verification probability of all other packets. Alternatively, when *N* is large, we can divide the *N* packets into smaller groups and form a butterfly graph within each group. However, this simple approach has high overhead and complexity, since there is one signature packet per group and signature packet is costly in terms of computation and transmission. This is exacerbated by the fact that each signature packet needs to be transmitted multiple times to avoid loss. We have found that we can overcome this problem, without incurring significant loss in performance, by limiting the number of rows (N_R) so that the signature packet is confined within one *MTU*, and the extra packets are appended at the end of the graph by adding additional columns. Any consecutive $\log_2 N_R+1$ columns taken from this relaxed graph constitute a Butterfly graph. Figure 3-11 shows an example with N_R=4 and N_C=8. Note that any 3 consecutive columns form a Butterfly graph of 12 packets. Therefore, to support an arbitrary number *N*, we can first construct a butterfly graph, and then append extra packets to the end of the butterfly graph.

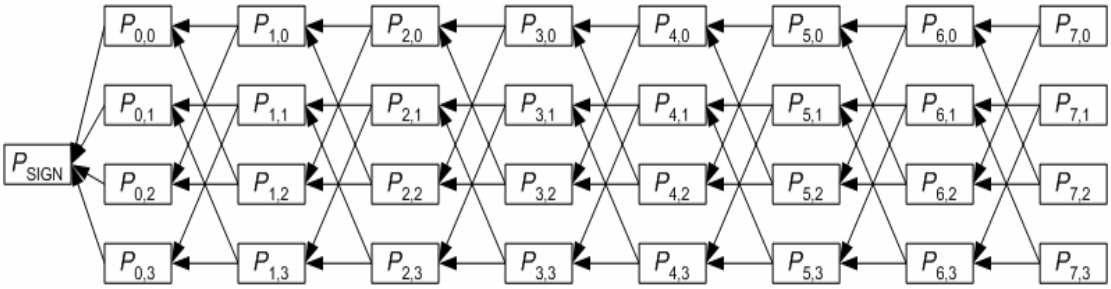

Figure 3-11 – Relaxed Butterfly graph with 4 rows and 8 columns

In the Butterfly graph, all packets within a column have the same verification probability, which drops for the first 2 columns and then becomes flat for the subsequent columns. The relaxed butterfly graph also has this property, as shown in Figure 3-12 which compares the verification probability of packets from different columns in a Butterfly (128x8) and a relaxed Butterfly graph (64x16). It shows that the extra columns in the relaxed butterfly have flat verification probability. Note that the result in Figure 3-12 is obtained by assuming the signature packet is always received. Without this assumption, the relaxed butterfly will have better performance because its signature packets size is much smaller than that of the original butterfly.

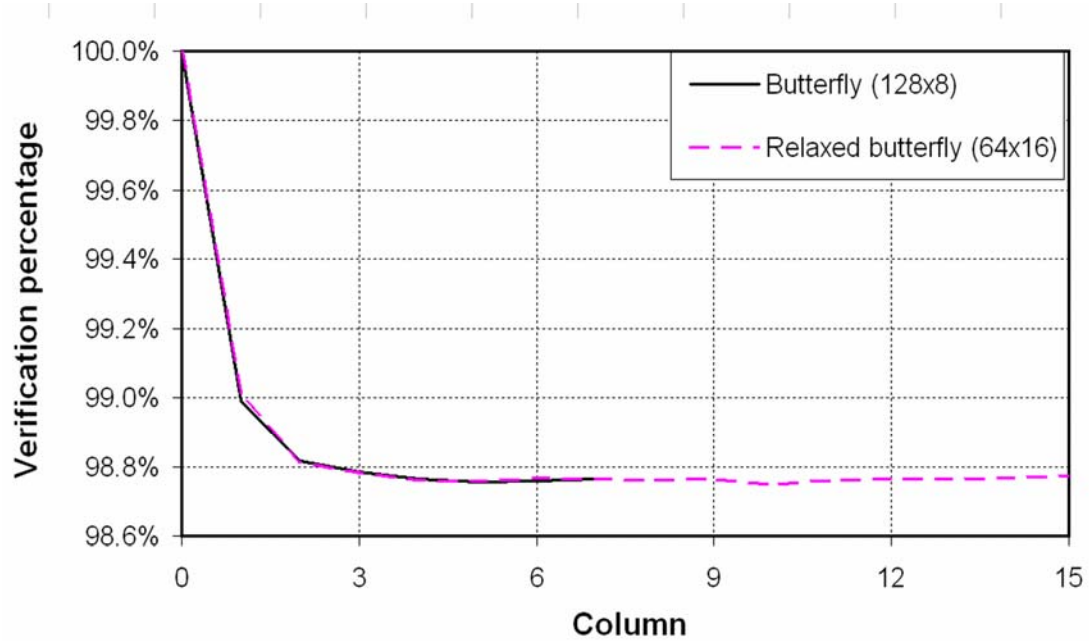

Figure 3-12 – Verification probability of packets in different columns of Butterfly and Relaxed butterfly graph (ε=0.1)

3.2.3 Generalized Butterfly Graph

This section proposes the *Generalized Butterfly Graph* (*GBG*) framework, and describes how to design an efficient authentication graph, based on the prior analysis.

Given N packets, a GBG is constructed as a matrix with N_R rows and N_C columns, where $N_R=2^K$ and $N_C=\lceil N/N_R \rceil$. A packet in r-th row and c-th column is referred to as $P_{c,r}$, where $0\le r<N_R$ and $0\le c<N_C$. The signature packet P_{SIGN} contains the signature and hashes of all packets in the first column. A packet $P_{c,r}$ is connected to $P_{c-1,r}$ and is possibly connected to other packets in column c-1. Note that the *GBG* framework covers many possible graphs. For instance, when $N_R=1$, the GBG graph is the same as the Simple Hash Chain [31]; when $N_R=N$, the *GBG* graph is equivalent to Tree-Authentication [29] with degree N. In addition, if $N=N_R(\log_2 N_R+1)$, the *GBG* graph is exactly the same as a Butterfly graph. In the proposed *GBG* framework, we examine how to design an efficient authentication graph. The input parameters include the total number of packets N, the overhead budget R_O, the packet loss rate ε, and the maximum transmission unit (*MTU*); the output parameters include N_R and N_C, the number of transmissions of signature packet M, and the graph topology (edge placement). The following subsections discuss how to choose the appropriate values for these output parameters.

3.2.3.1 Number of Rows and Columns

The value of N_R has two implications: it determines the size of the signature packet which contains signature (g bytes) and hashes (h bytes per hash) of all packets in the first column; it also affects robustness against network loss. For instance, if N_R is too big, the signature packet has to be fragmented for transmission, which negatively

affects overall verification probability. On the other hand, if N_R is too small, a burst loss of length N_R may cut the graph into two halves and the second half cannot be verified anymore. Therefore, N_R should be chosen to be the maximum number satisfying two conditions: $N_R=2^K$ and $g+N_Rh\leq MTU$, where K is a positive integer.

3.2.3.2 Number of Transmissions for Signature Packet

If the signature packet is lost, all the other packets are not verifiable, and therefore it has to be transmitted multiple times to avoid loss. However, every transmission is counted towards the overhead budget. For instance, if the signature packet is transmitted M times, the overhead budget left for the other packets is R_O-$M(g+N_Rh)$ bytes. A small value of M leads to high loss probability of the signature packet, while a large value of M leads to fewer edges in the graph for a given overhead budget. A one-dimensional search can be performed to find the optimal value for M, e.g., Figure 3-13 shows the verification percentage with different values of M. (Experiment setting: N=1024, R_O= 40960 bytes, g=128, h=20, MTU=1500 and ε =0.1).

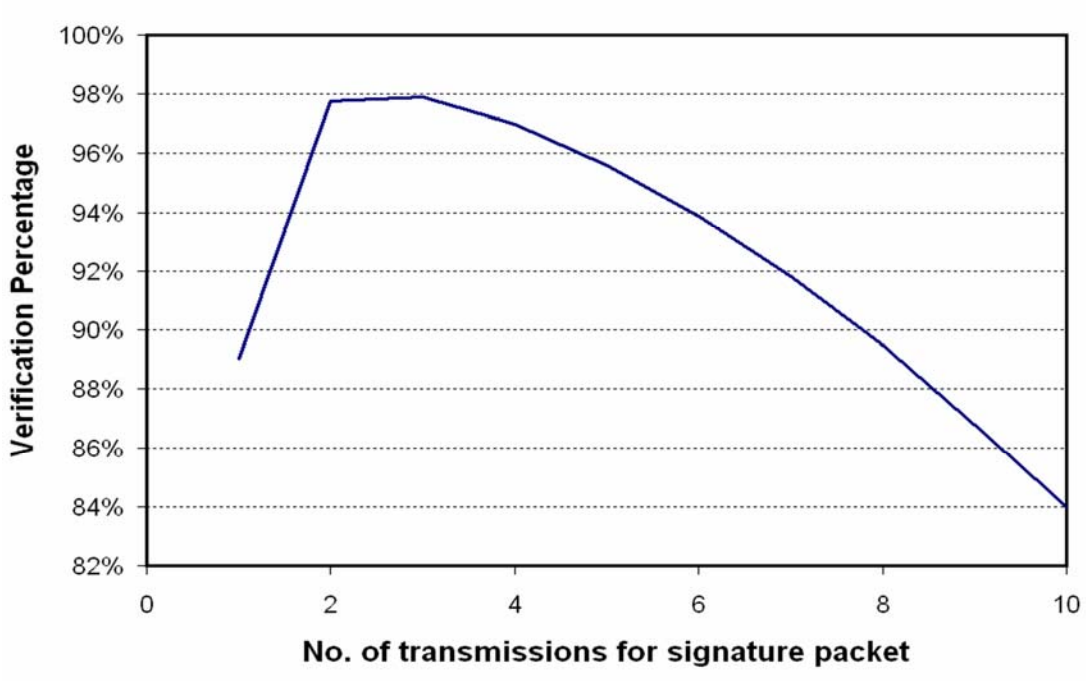

Figure 3-13 – Verification probability for various values of *M*

3.2.3.3 Edge Placement Strategy

Given a N_R by N_C *GBG*, the initial graph topology (as in Figure 3-5) and M transmissions of signature packet occupy $R_{init} = M(g+N_R h)+(N-N_R)h$ bytes. Therefore, the remaining overhead budget is R_O-R_{init}, corresponding to $e = \lfloor (R_O - R_{init})/h \rfloor$ extra edges which can be placed using the algorithm illustrated in Figure 3-14. The extra edges are placed in a round-robin manner, starting from packets in the middle column and progressively going outwards to the second column and the last column. After one round, if there is still extra edge left, the edge placement algorithm will repeat the above procedure, until there is no more extra edge left.

```
EdgePlacement(N_R, N_C, e) {
int L, R, round=0;

while(e > 0) {
  if(N_C%2==0)
    L = R = N_C/2
  else
    L = N_C/2;    R = L + 1;

  for( ; L>0 && e>0; L--, R++) {
    for(int i=0; i<N_R && e>0; i++) {
      if(round == 0)
        create an edge from P_i,L with butterfly constraint;
      else
        create an edge from P_i,L to the first node in column L-1 with least incoming
edges;
      e --;
    } //end for
    if(L == R)       continue;
    for(int i=0; i<N_R && e>0; i++) {
      if(round == 0)
        create an edge from P_i,R with butterfly constraint;
      else
        create an edge from P_i,R to the first node in column R-1 with least incoming
edges;
      e --;
    } //end for
  } //end for
  round ++;
 }//end while
} //end EdgePlacement(..)
```

Figure 3-14 – Algorithm to allocate e extra edges in (N_RxN_C) GBG graph

Note that the receiver can deduce the graph topology, given the values of N, N_R and R_O. Therefore, these three parameters are signaled in a header in the signature packet. The overhead corresponding to these three parameters is negligible, compared with hashes and signatures.

3.2.4 Performance Evaluation

To evaluate the performance of the proposed *GBG* authentication graph, we implemented 4 authentication graph methods and measure their performances in the following settings: N=1024, ε=0.01~0.2, MTU=1500, g=128 and h=20. The first method is the proposed *GBG* method with N_R=64 and N_C=16, the extra edges are placed using the algorithm in Figure 3-15, and the optimal value of M is selected by a one-dimensional search. The second method is EMSS authentication [33] where each packet is connected to a packet that is randomly selected from the preceding 64 packets and the signature packet contains the hash of the first 10 packets. The third method is Augmented Chain [34] $C_{15,7}$, where each packet is connected to a previous packet and another packet that is 8 packets away in the high-level chain, and 14 packets are iteratively inserted between two consecutive packets in high-level chain. The fourth method is the Butterfly graph with N_R=128 and N_C=8, and the signature packet contains hashes of all packets in the first column.

Figure 3-15 compares the LAF of the four approaches at various overhead levels for ε=0.1. The proposed GBG authentication method always outperforms (i.e., lower LAF) the other methods at all overheads. Further, when overhead is below 40 bytes per packet (i.e., 2 hashes/packet), it significantly outperforms *EMSS*. The other two methods, Butterfly and Augmented Chain, cannot be configured to have overhead

lower than 40 bytes per packet. However, when the overhead is greater than 40 bytes per packet, the performance gap is becoming smaller and smaller.

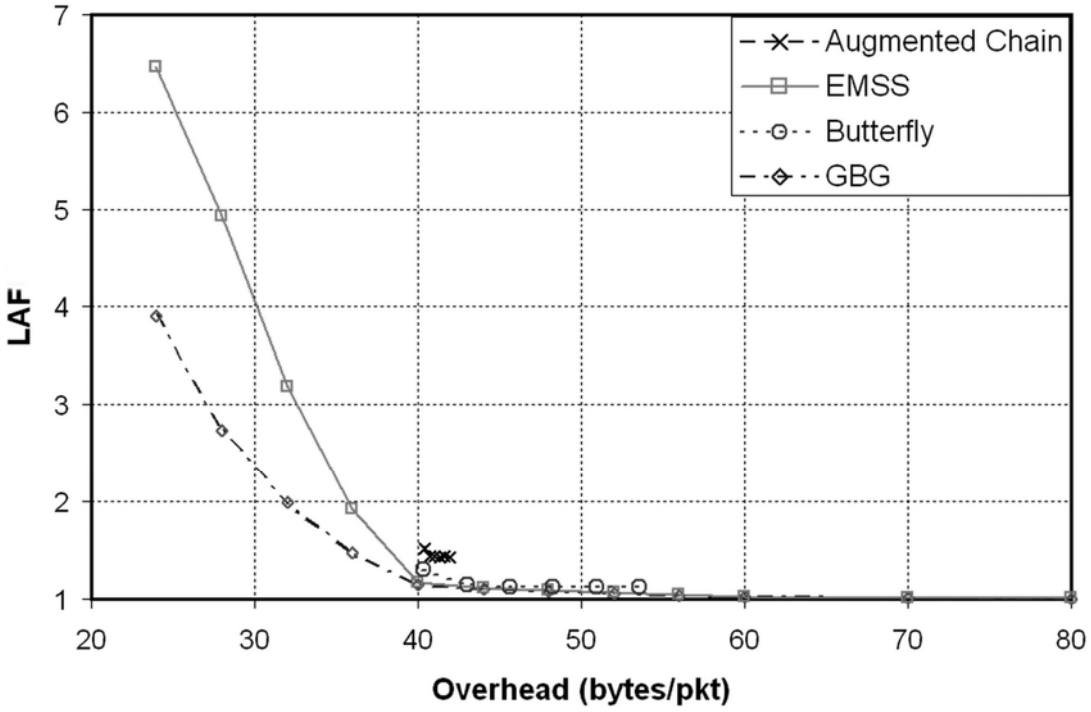

Figure 3-15 – Comparison of LAF at various overhead (ε=0.1)

Figure 3-16 compares the LAF of the four methods at various packet loss rates ranging from 0.01 to 0.2, where the overhead is fixed at 40 bytes per packet (i.e., around 2 hashes/packet). It shows that the proposed GBG method outperforms the other three methods at all loss rates. Note that although the performance gap between GBG and EMSS is small, GBG has much lower receiver delay than EMSS. For instance, a received packet can be immediately verified using GBG, because the signature is the first packet in the graph and all edges are pointing backward. However, using EMSS, a received packet cannot be verified because the signature is the last packet in the graph and all edges are pointing forward.

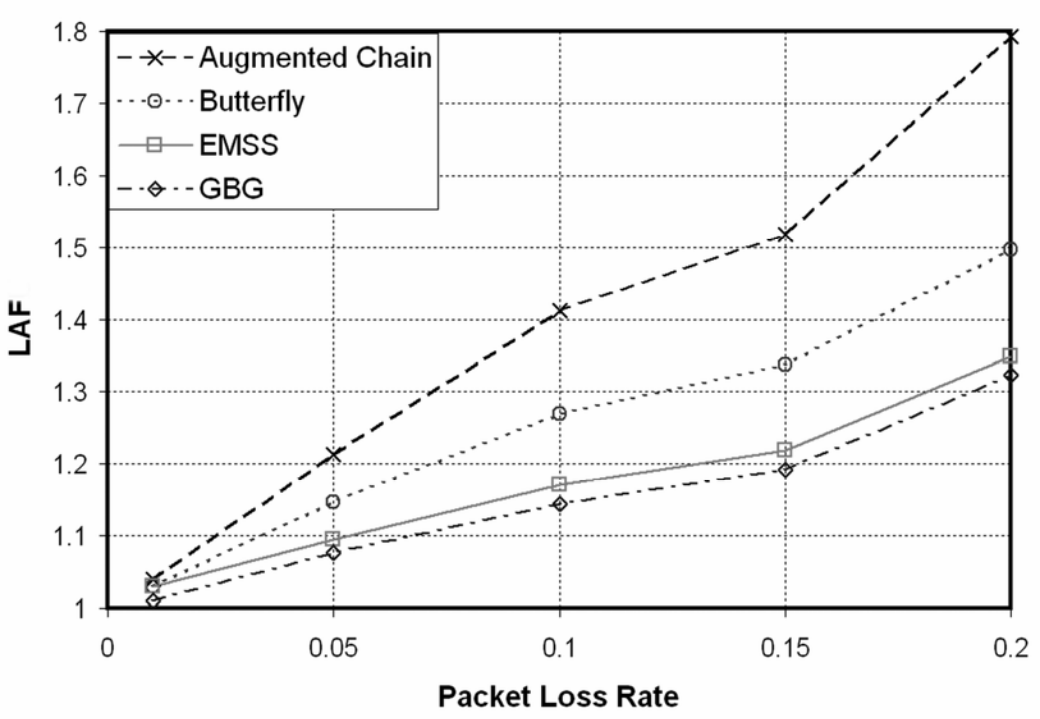

Figure 3-16 – Comparison of LAF at various loss rates (overhead = 40 bytes per packet)

3.3 CONCLUSIONS

In this chapter, we propose the Butterfly Authentication method, which exploits the fault-tolerance property of a butterfly graph. The analysis and experimental results demonstrate that the Butterfly Authentication method outperforms the existing graph-based authentication methods. However, the Butterfly Authentication has its own limitations, like fixed overhead, inflexible number of packets and potentially large signature packet. To overcome the abovementioned limitations, we propose a Generalized Butterfly Graph (GBG) framework and describe how to design a graph to maximize the verification probability given the total number of packets, overhead budget and packet loss rate. Experimental results demonstrate the performance improvement over the other methods.

CHAPTER 4 - CONTENT-AWARE STREAM AUTHENTICATION

Existing stream authentication methods [29][31][33][34][35] and the methods proposed in Chapter 3 give different ways of constructing an authentication graph with the aim to maximize the verification probability, for a given overhead. They assume all packets are equally important and the verification probability is proportional to the media quality for media stream, i.e., maximizing verification probability is equivalent to maximizing media quality.

However, this assumption is not true in many cases, especially for media stream. Figure 4-1 gives the distribution of packets' importance in a JPEG 2000 codestream for an image "Bike" (2048x2560) in Figure 4-4(c). The image is encoded with 16 layers and 160 packets per layer. The importance of a packet is measured by its distortion increment, which is the amount by which the overall distortion will decrease if the packet is decoded. From Figure 4-1, we can see that the packets' importance exhibits huge differences. Out of the 2560 packets, 2464 packets (more than 96%) have a distortion increment of less than 100 MSE units, and the rest of the packets (96 packets, less than 4%) have much greater distortion increment. In other words, a small number of packets are much more important than the rest. This characteristic is often exploited via Unequal Error Protection (UEP) to transport media data over a lossy network. For instance, more important packets are allocated with more transmission opportunities using ARQ or more redundant information using FEC.

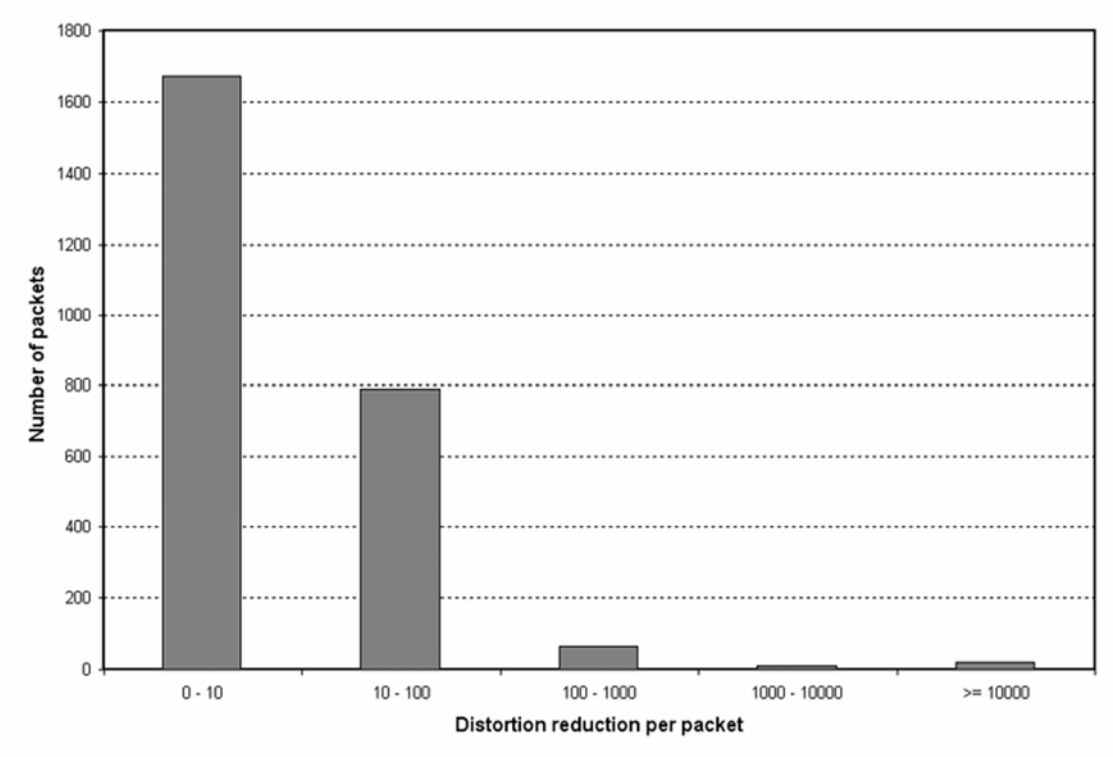

Figure 4-1 – Distribution of packets' distortion increment in a JPEG 2000 codestream (Bike 2048x2560)

Similarly, stream authentication can also exploit this characteristic of unequal importance to maximize the quality of authenticated media. Towards this goal, we differentiate the packets based on their importance, which can be obtained from the media encoding process. Some media formats provide packet importance information in their syntax. For example, JPSEC [4] and JPWL [5] have distortion field and MP4 file format [10] has hint track. More important packets are allocated with more overhead and thereby have high verification probability, while less important packets are allocated with less overhead. In other words, we formulate an optimization framework to compute an authentication graph to minimize the expected distortion of the authenticated media for a given overhead and given knowledge of packet loss rate, or conversely minimize the overhead for a given distortion. To differentiate from existing stream authentication methods which are all focusing on maximizing the

verification probability, we therefore name the proposed method as *content-aware* stream authentication.

In this chapter, section 4.1 describes the proposed distortion-overhead optimization framework for stream authentication. Section 4.2 describes how to build a content-aware optimized authentication graph under the proposed distortion-overhead optimization framework. Section 4.3 proposes a simplified authentication graph which has low complexity to construct. Finally, Section 4.4 presents analysis and performance evaluation for the proposed methods.

4.1 DISTORTION-OVERHEAD OPTIMIZATION FRAMEWORK

The problem of authenticating a media stream can be solved in the Distortion-Overhead Optimization framework, which is used to construct an authentication graph trading off two conflicting goals: minimal authentication overhead and minimal distortion (or maximal media quality). Given a specific overhead and network condition, we try to compute an authentication graph that minimizes the expected distortion of the authenticated media at the receiver. Conversely, the optimized authentication graph minimizes the authentication overhead, given a specific distortion and network condition. In other words, the distortion-overhead performance of the optimized graph lies on the lower convex hull of the set of all achievable distortion-overhead performances.

Recall that an authentication graph is a DAG, where a packet corresponds to a node. A directed edge from a packet A to another packet B is realized by appending A's hash to B, where packets A and B are referred to as the *source node* (or source packet) and *target node* (or target packet), respectively. The *redundancy degree* of a

packet is the number of edges originating from it, which also indicates the number of replicated copies of its hash that are appended to other packets. In particular, the redundancy degree is zero for a signature packet, as there is no edge originating from a signature packet.

To formulate the distortion-overhead optimization problem, we define θ_n as the set of target nodes of the edges originating from the packet P_n, and the redundancy degree of P_n is number of elements in θ_n, $|\theta_n|$. The variable θ_n is also referred to as topology policy for packet P_n. For example in Figure 1-4(a), packet P_2 has $\theta_2 = \{P_0, P_1\}$ and its redundancy degree is 2. The signature packet is a special case with no edge originating from it, and therefore its redundancy degree is 0. Given a set of N packets, the overall topology policy $\theta = [\theta_0, \theta_1, ...\theta_n ...\theta_{N-1}]$ uniquely defines the topology of the authentication graph. Denoting the total authentication overhead (including signature and hashes) as O and the overall media distortion as D, our goal is to find the optimal vector θ^* that minimizes the expected Lagrangian in Eq(4.1) for a given $\lambda > 0$. The Lagrange multiplier λ is used to control the trade-off between the overhead O and the expected distortion D. For instance, a smaller value of λ will result in an optimized vector θ^* leading to smaller expected distortion and higher overhead, and vice versa.

$$\theta^* = \arg\min_{\theta}(D + \lambda O) \tag{4.1}$$

The authentication overhead O is the extra bytes introduced for stream authentication. For instance, it includes the digital signature and the hashes appended to the packets. Therefore, the authentication overhead O is a function of the graph topology θ, as expressed in Eq(4.2), where g is the size of a digital signature and h is the size of a crypto hash.

$$O(\theta) = g + \sum_{n=0}^{N-1} h|\theta_n| \tag{4.2}$$

Recall that an authenticated media is decoded exclusively from packets that are both decodable and verifiable. A decodable but unverifiable packet should not be used for media decoding because of security concern; however, a verifiable but undecodable packet is also meaningless. Therefore, the expected distortion can be expressed as in Eq(4.3), assuming distortion is additive and D_0 is the distortion when no packet is received. For instance, if a packet P_n is both decodable (ξ_n is the probability of being decodable) and verifiable ($V(\theta_n)$ is the probability of being verifiable), its distortion increment Δd_n will be deducted from D_0. Note that ξ_n is equivalent to the probability that the packet P_n is received and all the packets it depends on for decoding are received and verified. Also note that $V(\theta_n)$ is a function of the topology policy θ_n, i.e., the verification probability is determined by how the packet is connected to the other packets in the authentication graph.

$$D(\theta) = D_0 - \sum_{n=0}^{N-1} \Delta d_n \xi_n V(\theta_n) \tag{4.3}$$

With Eq(4.1), Eq(4.2) and Eq(4.3), we define a distortion-overhead optimization framework which can be used to search for a graph topology θ to achieve minimized distortion of the authenticated media for a given overhead, or conversely to achieve minimized overhead for a given distortion. The problem is solved in the subsequent sections of this chapter.

4.2 A CONTENT-AWARE OPTIMIZED STREAM AUTHENTICATION METHOD

This section proposes methods to build a content-aware optimized authentication graph. We start by assuming a general layered media format shown in Figure 4-2 with L layers and Q packets per layer, where each packet depends on all the corresponding lower layer packets for decoding. For instance, a packet P_q^l is decodable if packet $P_q^{l'}$ is received and verifiable for all $0 \le l' < l$. In the rest of this chapter, the notation is changed for convenience of representing packets in layered media format. A packet P_q^l has a distortion increment Δd_q^l, topology policy θ_q^l, probability of being decodable ξ_q^l and verification probability $V\left(\theta_q^l\right)$.

To authenticate a layered media data with L layers and Q packets per layer, the most vital step is to find the optimized graph topology in the distortion-overhead optimization framework proposed in Section 4.1. The Lagrangian expression in Eq(4.1) can be rewritten for layered media data as follows:

$$\begin{aligned} J(\theta) &= D(\theta) + \lambda O(\theta) \\ &= D_0 + \lambda g + \sum_{l=0}^{L-1}\sum_{q=0}^{Q-1}\left(\left(-\Delta d_q^l \xi_q^l V\left(\theta_q^l\right)\right) + \lambda\left|\theta_q^l\right| h\right) \end{aligned} \tag{4.4}$$

Given a Lagrange multiplier $\lambda > 0$, we need to find the optimized topology policy θ that minimizes the Lagrangian $J(\theta)$. This can be accomplished in two steps: the first step is to determine the topology policy θ_q^l for all high layer packets θ_q^l where $0 < l < L$ and $0 \le q < Q$, as detailed in Section 4.2.1. The second step is to determine the topology policy θ_q^0 of a layer-0 packet P_q^0 where $0 \le q < Q$, as detailed in Section 4.2.2.

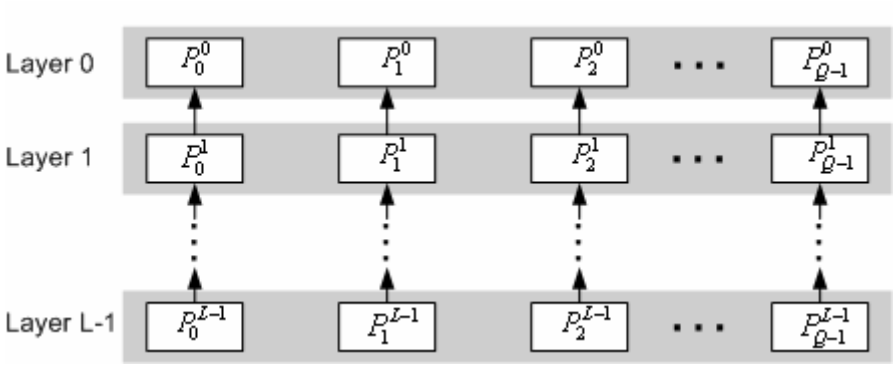

Figure 4-2 – General layered media format with *L* layers and *Q* packets per layer

4.2.1 Topology Policy for High-Layer Packets

In the layered media data depicted in Figure 4-2, a high layer packet is not decodable unless all the corresponding lower layer packets are decodable and authenticated, i.e., P_q^l depends on $P_q^{l'}$ for all l' such that $0 \le l' < l$, we say that P_q^l is a descendent of $P_q^{l'}$ and $P_q^{l'}$ is an ancestor of P_q^l. In this case, it is sufficient to have an edge originating from packet P_q^l and ending at one of its ancestors. The reason is as follow: if a packet P_q^l is decodable (i.e., all its ancestors are decodable and verifiable), it is verifiable with probability of 1 as long as it has an edge connecting to one of its ancestors; if the packet is not decodable, it is meaningless to verify it. Furthermore, the target node of the only edge originating from P_q^l is chosen to be its immediate ancestor P_q^{l-1}, i.e., $\theta_q^l = \left\{P_q^{l-1}\right\}$, because choosing any other ancestor is not optimal in the rate-distortion sense. Being a target node of an edge implies that a hash is appended to it, which will dilute its R-D importance. Since P_q^{l-1} is the least important packet among the ancestors of P_q^l, it should be chosen as the target packet of an edge originating from P_q^l.

Therefore, given the fixed topology policy for all other packets, the topology policy of P_q^l, $\theta_q^l = \left\{P_q^{l-1}\right\}$, minimizes the Lagrangian $J(\theta)$ as the resulting verification probability $V\left(\theta_q^l\right)$ is equal to 1 and the redundancy degree $\left|\theta_q^l\right|$ takes the smallest value of 1. Therefore, the topology policy for all high layer packets should be $\theta_q^l = \left\{P_q^{l-1}\right\}$, where $0 < l < L$ and $0 \le q < Q$, in order to obtain optimized distortion-overhead performance.

4.2.2 Topology Policy for Layer-0 Packets

After determining the topology policy for high layer packets, the decoding probability of a packet P_q^l can be expressed as in Eq(4.5), where ε_q^l is the probability that packet P_q^l is not received. In other words, the packet P_q^l is decodable if and only if all its ancestors (namely, $P_q^0, P_q^1, ..., P_q^{l-1}$) and P_q^l itself are received, and the layer-0 packet, P_q^0, is verifiable.

$$\xi_q^l = V\left(\theta_q^0\right)\prod_{l'=0}^{l}\left(1-\varepsilon_q^{l'}\right) \tag{4.5}$$

Substituting Eq(4.5) into Eq(4.4), the Lagrangian $J(\theta)$ can be expressed as Eq(4.6)

$$J(\theta) = D_0 + \lambda g + \sum_{l=0}^{L-1}\sum_{q=0}^{Q-1}\left(\left(-\Delta d_q^l V\left(\theta_q^0\right)\prod_{l'=0}^{l}\left(1-\varepsilon_q^{l'}\right)\right) + \lambda\left|\theta_q^l\right| h\right) \tag{4.6}$$
$$where \quad \theta_q^l = \left\{P_q^{l-1}\right\} \quad when \quad l > 0$$

To ensure the authentication graph is acyclic, we mandate that for edges with source node P_q^0, the target node must be $P_{q'}^0$, where $q' < q$. At this step, the goal is to find the optimized topology policy for each layer-0 packet that minimizes the

Lagrangian $J(\theta)$. The straightforward method is exhaustive each, but its high computational complexity is not acceptable. A more practical approach is to use an iterative descent algorithm [84], where the objective function $J(\theta)$ is minimized by searching for optimized topology policy for one packet at a time, keeping the other packets' policies fixed. These steps are repeated until the Lagrangian $J(\theta)$ converges. For instance, let $\theta(0)=\left[\theta_0^0(0),\theta_1^0(0),...,\theta_{Q-1}^0(0)\right]$ be the initial topology policy for all layer-0 packets, the policy at the *n*-th iteration, $\theta(n)=\left[\theta_0^0(n),\theta_1^0(n),...,\theta_{Q-1}^0(n)\right]$ where *n*>0, is determined as follows. At the *n*-th iteration, we select one packet, say, P_q^0, to find its θ_q^0. For any $q'\neq q$, let $\theta_{q'}^0(n)=\theta_{q'}^0(n-1)$, while for packet P_q^0, let

$$\begin{aligned}\theta_q^0(n) &= \arg\min_{\theta_q^0} J\left(\theta_0^0(n),\theta_1^0(n),...,\theta_q^0,...,\theta_{Q-1}^0(n)\right)\\ &= \arg\min_{\theta_q^0} \mu_q^0 V\left(\theta_q^0\right)+\lambda h\left|\theta_q^0\right|\end{aligned} \tag{4.7}$$

Eq(4.7) follows from Eq(4.6) with Eq(4.8).

$$\mu_q^0=\sum_{l=0}^{L-1}\left(\Delta d_q^l\prod_{l'=0}^{l}\left(1-\varepsilon_q^{l'}\right)\right) \tag{4.8}$$

At the *n*-th iteration, we need to search for a topology policy for a particular packet that minimizes the Lagrangian value. In subsequent iterations, the same process is repeated for a different packet. The utility value μ_q^0 of the packet P_q^0, as expressed in Eq(4.8), can be regarded as the amount by which the overall distortion will increase if P_q^0 is not verifiable given that it is decodable. The utility value μ_q^0 can be derived by taking the partial derivative of Eq(4.6) with respect to $V\left(\theta_q^0\right)$.

Therefore, the larger the utility value μ_q^0 is, the more attractive it is to increase its verification probability $V\left(\theta_q^0\right)$ and to increase its redundancy degree $\left|\theta_q^0\right|$.

During these optimization iterations, the Lagrangian $J(\theta)$ is non-increasing and has a lower bound of 0, so convergence is guaranteed. However, we cannot guarantee that it can reach a global minimal. To increase the chances of reading the global minimal, one alternative solution is to invoke the optimization process with multiple initial topology policy vectors, and choose the resulting vector that incurs the smallest Lagrangian values.

4.3 A SIMPLIFIED AUTHENTICATION GRAPH

Building the distortion-overhead optimized graph is computationally intensive because many iterations are required before convergence, and each iteration needs to search for the optimal topology policy for the selected packet. In this section, we empirically build a simplified authentication graph which requires much lower computation complexity.

For the high layer packets P_q^l (where $0 < l < L$), their topology policies are exactly the same as the distortion-overhead optimized graph, i.e. P_q^l has only one edge connecting to its immediate ancestor P_q^{l-1}. For each packet P_q^0 in layer 0, we first compute the utility value μ_m^0 using Eq(4.8). After that, the packets are sorted into a list where the utility values are in descending order. The sorted list, denoted by *SL*, is then divided into *S* segments, namely, Seg_0, Seg_1..., Seg_{S-1}. Each packet in Seg_i has γ_i outgoing edges whose target packets are randomly selected from the preceding

packets in *SL*. The redundancy degree in consecutive segments is in decreasing order, i.e. $\gamma_0 > \gamma_1 > ... > \gamma_{S-1}$. There are a number of possible segmentation methods. One is based on equal segment size. Another method is based on utility value, e.g., Seg_i contains all packets with utility value falling in $\left[\mu_{\min} + i \times \eta, \mu_{\min} + (i+1) \times \eta\right)$, where $\mu_{\min}$ and $\mu_{\max}$ are the lowest and highest utility values, respectively, and $\eta = \left(\mu_{\max} - \mu_{\min}\right)/S$. The first method makes it easier to control the average redundancy degree, while the second method has slightly better performance, because its overhead allocation is more related to the packets' utility value. The signature packet contains the hashes of the first *Z* packets in *SL*. The parameter *Z* is typically sufficient to be 3 and our experiment uses *Z*=3. Table 4-1 summarizes the above parameters used for constructing the authentication graph. Through experiments, we found that it is typically sufficient to have equal-sized segmentation, *S*=3 and $\gamma_0 = \gamma_1 + 1 = \gamma_2 + 2$, where the redundancy degree is γ_1 on average.

Table 4-1 – Parameters and semantics of the proposed simplified authentication scheme

Parameters	Semantics
S	The number of segments. If *S* is 1, all lowest layer packets have the same redundancy degree
γ_i	The redundancy degree of packet in Seg_i.
Z	The number of packets whose hashes are included in the signature packet

The algorithm for constructing a simplified authentication graph is summarized in Figure 4-3(a), which is illustrated in Figure 4-3(b) using a layered media stream with 3 layers and 8 packets per layer, assuming that all packets are non-

empty. This example uses the following parameters: S=2, $\gamma_0 = 2$, $\gamma_1 = 1$, and Z=2. First, for each high layer packet P_m^i (i>0), an edge is created from P_m^i to its immediate ancestor P_m^{i-1}. Second, for packets in layer 0, we compute their utility values, sort them and divide them into S=2 segments of equal size. After that, for each packet in segment 0, γ_0=2 outgoing edges are created with the target packet randomly chosen from the proceeding packets in the sorted list, and similarly for packets in segment 1. Finally, a signature packet is created containing the hash of the first 2 packets in the sorted list.

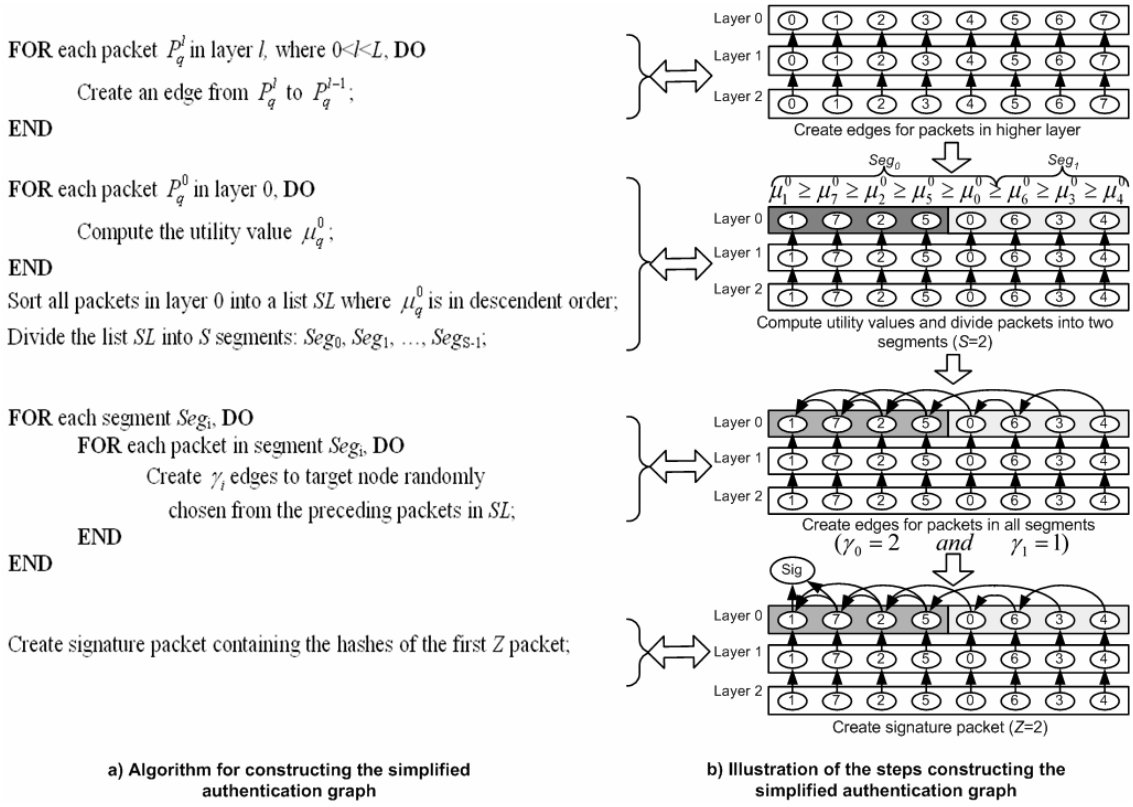

Figure 4-3 – Algorithm and example of constructing a simplified authentication graph

4.4 ANALYSIS AND EXPERIMENTAL RESULTS

This section analyzes the optimized content-aware authentication method and compares it with existing methods, in various aspects like complexity, overhead, sender delay, receiver delay, verification probability and quality of authenticated media.

4.4.1 Comparison with Existing Methods

Since the proposed content-aware optimized stream authentication methods are based on DAG, we choose graph-based authentication methods for comparison, including Simple Hash Chain [31], Tree Authentication [29], EMSS [33], Augmented Chain [34] and Butterfly Authentication. The performance criteria are complexity, overhead, verification probability, sender delay and receiver delay, which have been explained in Section 1.2.5.

Table 4-2 – Comparison of the content-aware authentication method against the existing methods

	Simple Hash Chain	Tree-Authentication	EMSS	Augmented Chain	Butterfly	Content-aware
Complexity	N,1	$2N$-1,1	N+1, 1	N+1, 1	N+1, 1	N+1, 1
Overhead	h+g/N	$g+h\log_2 N$	*Variable*	$2h+g/N$	$2h+g/N$	Variable
Verification probability	*Low*	*Very High* (1)	*High*	*High*	*High*	*High*
Sender delay	N	N	1	p	N	N
Receiver delay	1	1	N	N	1	1

Table 4-2 summarizes the performance of these six authentication schemes. The values are obtained based on the following assumptions: 1) A digital signature is

amortized to a group of N packets; 2) h is the size of a hash and g is the size of a digital signature; and 3) The tree-authentication method uses a binary tree (i.e., degree of two); (4) The augmented chain is parameterized by variables a and p, which are named in [34] as the *chain length* and *packet buffer size* on the sender side, respectively.

In terms of computation overhead, the Tree-authentication scheme has to perform about N more hash operations than the other schemes. In terms of sender delay and receiver delay, the Augmented Chain has the worst performance, while the other schemes are the same.

The Simple Hash Chain has the smallest overhead (only one hash per packet), but its verification probability is also the lowest. On the other hand, Tree-authentication has the highest overhead but also the highest verification probability. These two methods are two extreme cases (one favors lower overhead while the other favors higher verification probability), while all other methods try to achieve a balance in-between. In particular, both EMSS and the proposed content-aware method can be configured with different overheads, resulting in different verification probabilities. The Augmented Chain and Butterfly Authentication have fixed overhead. In this regard, EMSS and the proposed content-aware method are more flexible and generically applicable since their overhead level is tunable.

The above comparisons are for streaming general data packets. However, for media streams, we should use different benchmark criteria. The most important performance measure should be the PSNR of the authenticated image rather than the verification probability of the delivered packets. In this regard, our content-aware scheme is more efficient than existing schemes due to two reasons:

1. It eliminates all the unnecessary edges (therefore reduce the overhead) that can help to increase the verification probability only, but do not help to increase the PSNR of the authenticated image. For example, if packet P_i depends on P_j, it is sufficient to have an edge from P_i to P_j, and any edge from P_i to any other packet does not help to improve the PSNR of the authenticated image.
2. The overhead is used in a more efficient manner. The more important packets have more outgoing edges (i.e., higher redundancy degree), which help to increase the PSNR, while the less important packets (which account for a large proportion of the total packets) have less outgoing edges (i.e., smaller redundancy degree), resulting in smaller overhead.

4.4.2 Security Analysis

Similar to the existing graph-based authentication methods, the content-aware authentication method relies on the hash chaining and digital signature. Therefore, the security strength of this scheme is the same as the underlying cryptographic algorithms. For example, SHA-1 can be used for one-way hashing and RSA can be used for signature generation and verification. For more details on security strength, refer to Merkle's tree authentication [30].

4.4.3 Discussion of Utility Values

As every network has a *MTU* size, a large packet has to be segmented into smaller datagrams for transmission. Therefore, the probability of losing the packet P_q^l,

ε_q^l, can be expressed in Eq(4.9), where ε is the loss probability of each network datagram (here we assume i.i.d random loss) and R_m^l is the size of P_q^l in bytes.

$$\varepsilon_q^l = 1-(1-\varepsilon)^{\lceil R_q^l/MTU \rceil} \tag{4.9}$$

By substituting Eq(4.9) into Eq(4.8), we can see that the utility value, μ_q^l, is determined by the distortion increment values of its descendent packets, packet sizes of its descendent packets, the loss probability ε, and the *MTU*. If the packet size R_q^l is big, it needs more datagrams for transmission, thereby decreasing its probability of being received. So, given a fixed value for ε and *MTU*, the packet P_q^l will have a greater utility value when its descendent packets have larger distortion value and smaller size.

However, the value of the *MTU* varies with physical network links, e.g., Ethernet has a *MTU* of about 1500 bytes and the ATM network has a *MTU* of 53 bytes. When a communication path between two end hosts consists of different physical links, the smallest *MTU* can be discovered by the end host using the path MTU discovery protocol [85].

To derive the utility values, the sender also needs to know the network loss probability ε. The sender could presume a reasonable value for it, or could estimate it based on past communications. For instance, the Real-Time Transport Protocol (RTP) [86] specifies the window-based measurement techniques for estimating the loss probability. However, for the simplified authentication graph, it is not important to have an accurate value of ε, because it does not change the relative order of the utility values in the sorted list *SL*.

4.4.4 Experimental Results

This section experimentally compares our content-aware authentication method against EMSS, Augmented Chain and Butterfly authentication methods to further demonstrate the validity of our proposed scheme. We choose JPEG 2000 coded images in our experiment.

We implement five schemes, namely, *WITHOUT_AUTH*, *EMSS_AUTH, C_A_AUTH, AC_AUTH* and *BUTTERFLY_AUTH*. The first scheme, *WITHOUT_AUTH*, simply sends the packets in the order they appear in the JPEG-2000 codestream, and no authentication is applied. This scheme provides a reference for the achievable distortion performance if verification is not required. Note that this scheme provides an upper bound on the performance of all authentication schemes. The second scheme, *EMSS_AUTH*, implements the EMSS authentication [33], where every packet has the same redundancy degree. The third scheme, *C_A_AUTH*, implements the proposed content-aware authentication using the simplified authentication graph as proposed in Section 4.3. Through simulation, we find that the content-aware scheme yields good performance when the parameter S is 3. Further increasing S does not produce substantial performance improvement, because its performance is already quite close to the upper bound when S is set to 3. The fourth scheme, *AC_AUTH*, implements the Augmented Chain [34], and the fifth scheme, *BUTTERFLY_AUTH*, implements the butterfly authentication. For all schemes, the packets are sent in the order they appear in JPEG-2000 codestreams, while the signature packet is sent multiple times to minimize its loss probability.

The network is modeled by an i.i.d distribution, where the average loss probability ranges from 0 to 0.15. In addition, the *MTU* is set to 1500 bytes, as used in

Ethernet. Our experiment uses 8 JPEG-2000 testing images (each 2560x2048 pixels). Each image has 4 resolution levels and 260 packets per layer, and we vary the number of layers in the experiments. The test images used in our experiments are shown in Figure 4-4.

Figure 4-4 – The testing images used in the experiments

The first experiment is to demonstrate the effectiveness of the authentication redundancy adapted to the distortion. The JPEG 2000 images are encoded with only 1 layer, so *C_A_AUTH* can take advantage of the distortion information but not the layer structure. For *C_A_AUTH*, the parameters are set as follows: S=3, $\gamma_0 = 3$, $\gamma_1 = 2$, $\gamma_2 = 1$, and Z=6, so the redundancy degree is 2 on average. The other schemes use a similar level of redundancy. Figure 4-5 gives the PSNR of the five schemes. *C_A_AUTH* consistently outperforms the other schemes at all network loss rates. In fact, the PSNR curve of *C_A_AUTH* is very close to that of *WITHOUT_AUTH*, which achieves our original design goal, because the authentication overhead is added in an optimized manner.

Figure 4-6 shows the verification probabilities of the four authentication schemes. When the loss rate is less than 0.1, *C_A_AUTH* has a slightly lower verification probability, because one third of the packets have redundancy degree of 1. When the loss rate is larger than 0.1, a flat redundancy degree of 2 for all packets is not enough, which causes a dramatic decrease for *EMSS_AUTH, AC_AUTH* and *BUTTERFLY_AUTH*. For *C_A_AUTH*, such decrease is relatively small because one third of the packets still have redundancy degree of 3.

From Figure 4-5 and Figure 4-6, we can see that although *C_A_AUTH* sometimes has lower verification probability than the other authentication schemes, it produces higher PSNR. The *C_A_AUTH* is able to achieve near distortion-overhead optimized performance, as the authentication overhead is added in a more cost-effective manner. The packets are differentiated according to their importance, more important packets have higher verification probability and less important packets have lower authentication overhead.

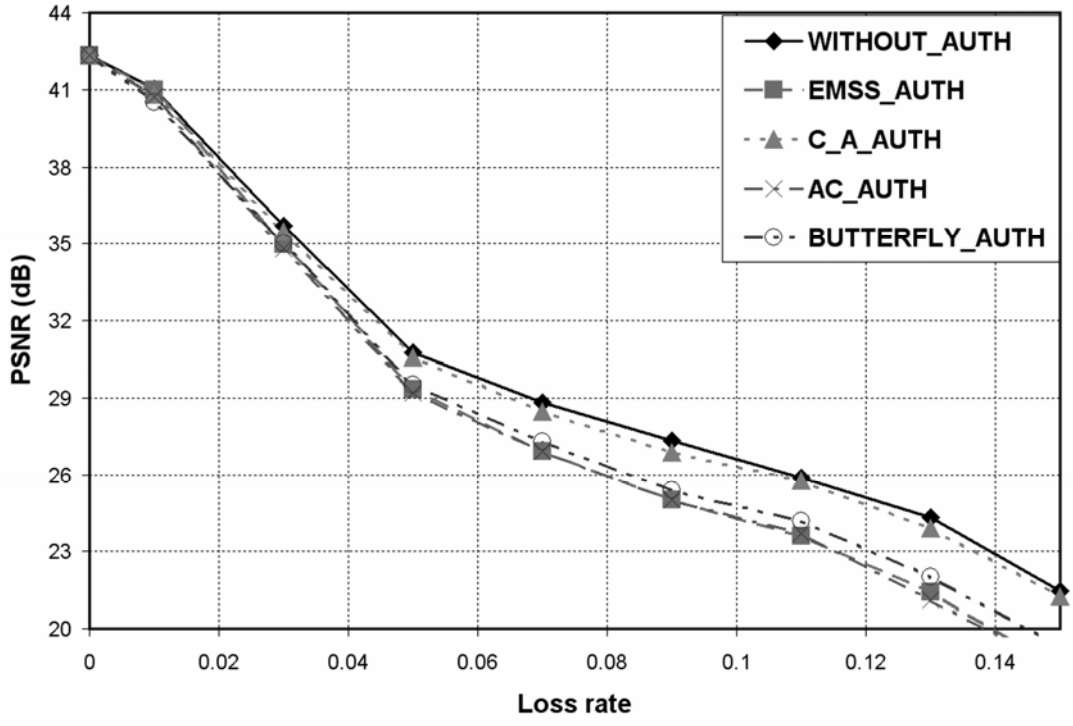

Figure 4-5 – PSNR at various loss rates (2 hashes / Packet on average, with 1 layer)

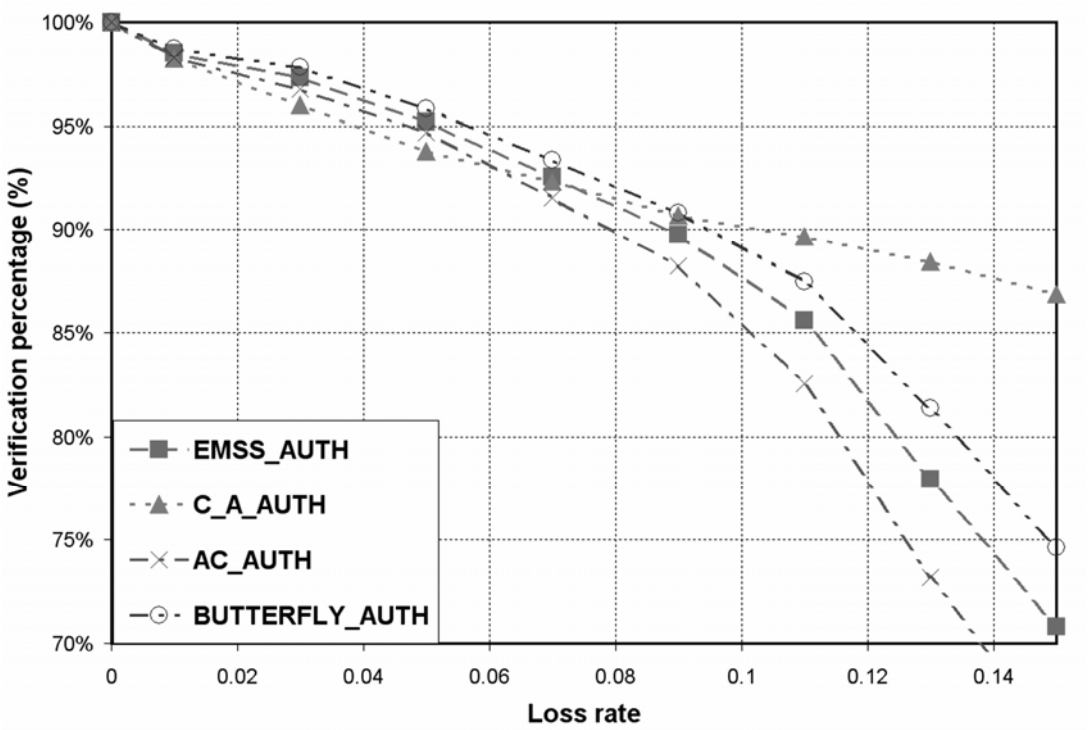

Figure 4-6 – Verification probability at various loss rates (2 hashes/packet on average with 1 layer)

The second experiment is to evaluate the proposed system when both distortion and layer structure are utilized. Accordingly, the JPEG 2000 codestreams are encoded with 6 layers. Thus, *C_A_AUTH* is able to take advantage of both the layer structure and the distortion information, but the other schemes are agnostic to both. The parameters for this experiment are the same as that for the previous experiment. Figure 4-7 shows the PSNR curves of the three systems, which are similar to those in Figure 4-5.

Figure 4-8 compares the verification probabilities of the four authentication schemes. The *C_A_AUTH* has significantly lower verification probability than the other schemes. This is because higher layer packets have redundancy degree of only one in *C_A_AUTH*, resulting in lower verification probability.

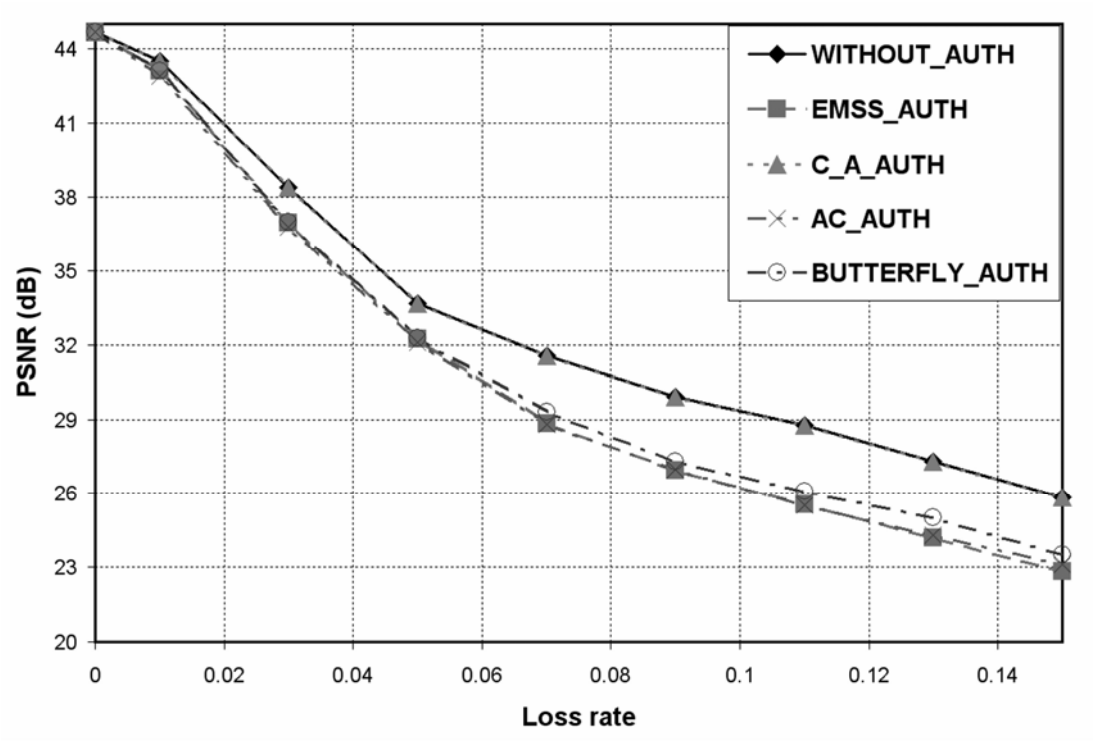

Figure 4-7 – PSNR at various loss rates (2 hashes / packet on average, with 6 layers)

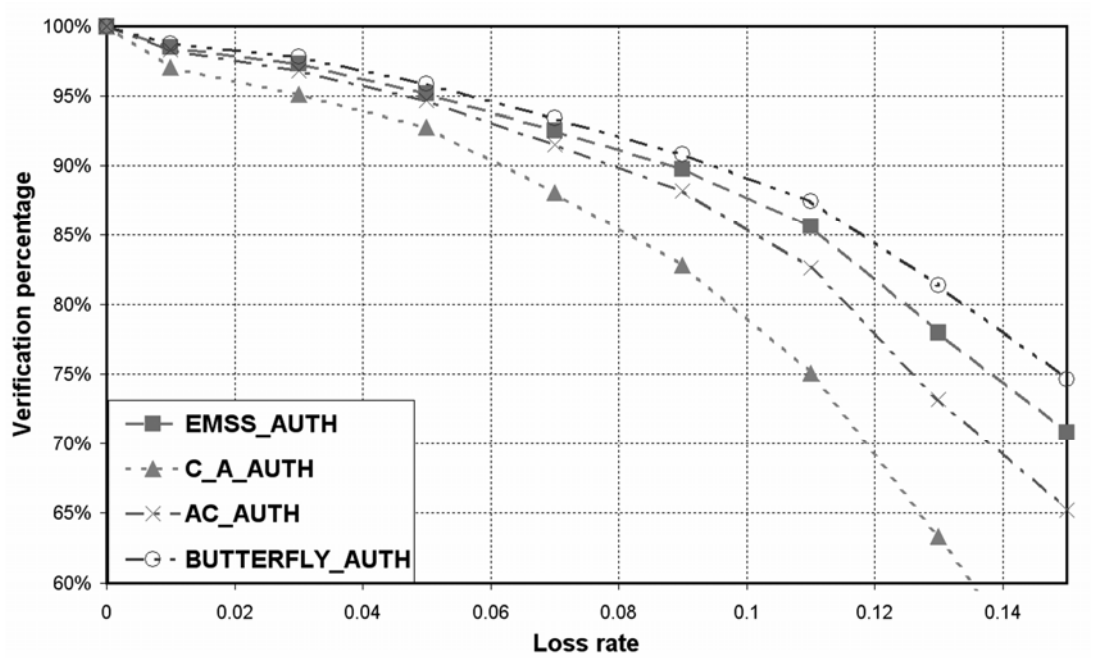

Figure 4-8 – Verification probability at various loss rates (2 hashes / packet on average, with 6 layers)

From the results of the previous two experiments (Figure 4-5, Figure 4-6, Figure 4-7 and Figure 4-8), we can conclude that *C_A_AUTH* has PSNR very close

to *WITHOUT_AUTH*. In addition, it consistently outperforms the other scheme in terms of PSNR, even though it may have lower verification probability. This is because the proposed *C_A_AUTH* method tries to optimize the quality of authenticated media, while the other methods try to optimize the verification probability.

The third experiment evaluates the PSNR versus bit-rate curve for transmitting the authenticated image over a lossy network. The JPEG 2000 codestreams are also encoded with 6 layers. All authentication schemes have the same average redundancy degree of 2. The network loss probability is set to 0.05, because higher loss probability will flatten the rate-PSNR curve. All other parameters are the same as in the second experiment. Figure 4-9 shows the PSNR curves. Again, at all image bit-rates, *C_A_AUTH* achieves a PSNR close to *WITHOUT_AUTH*. It consistently outperforms the other schemes in terms of the media quality, due to the same reason as explained in previous experiment.

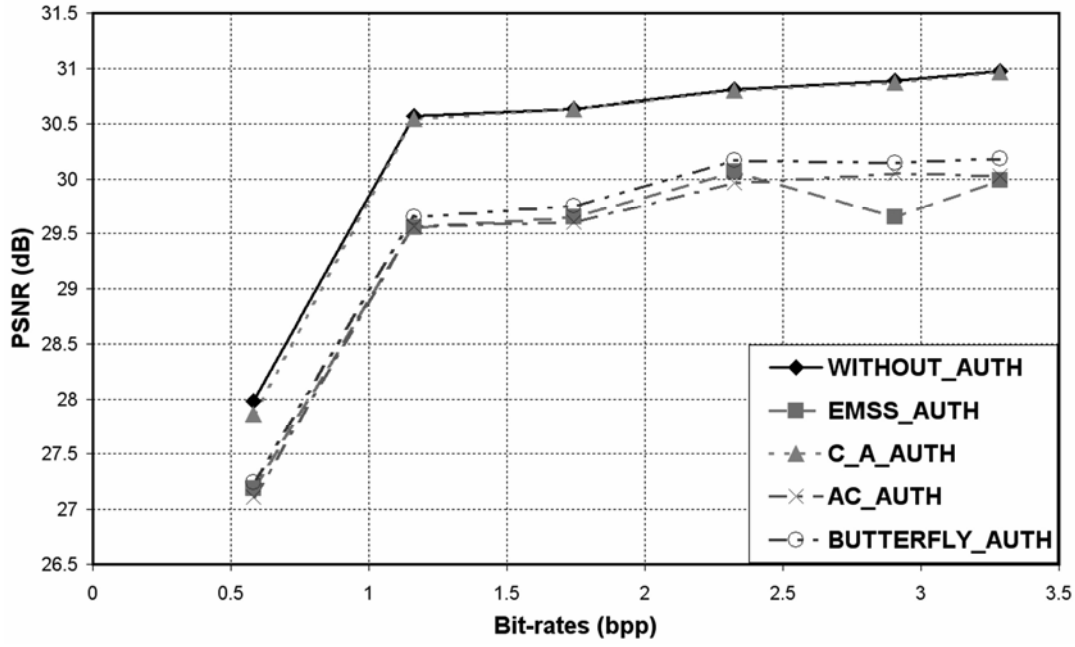

Figure 4-9 – PSNR at various bit-rates (loss rate=0.05, 2 hashes / packet on average, with 6 layers)

The fourth experiment is to compare the performance of the four authentication schemes at various overhead levels. Again, we set the loss probability to 0.05. The JPEG-2000 images are encoded with 1 layer, because we want *C_A_AUTH* to utilize the distortion information but not the layer structure. We measure the PSNR at various overhead levels, ranging from 1 to 6 hashes per packet. Figure 4-10 shows that at a loss rate 0.05 the proposed scheme outperforms the other schemes when the redundancy degree is less then 3. When the loss rate is higher, the improvement of the proposed scheme over the other schemes will be further increased.

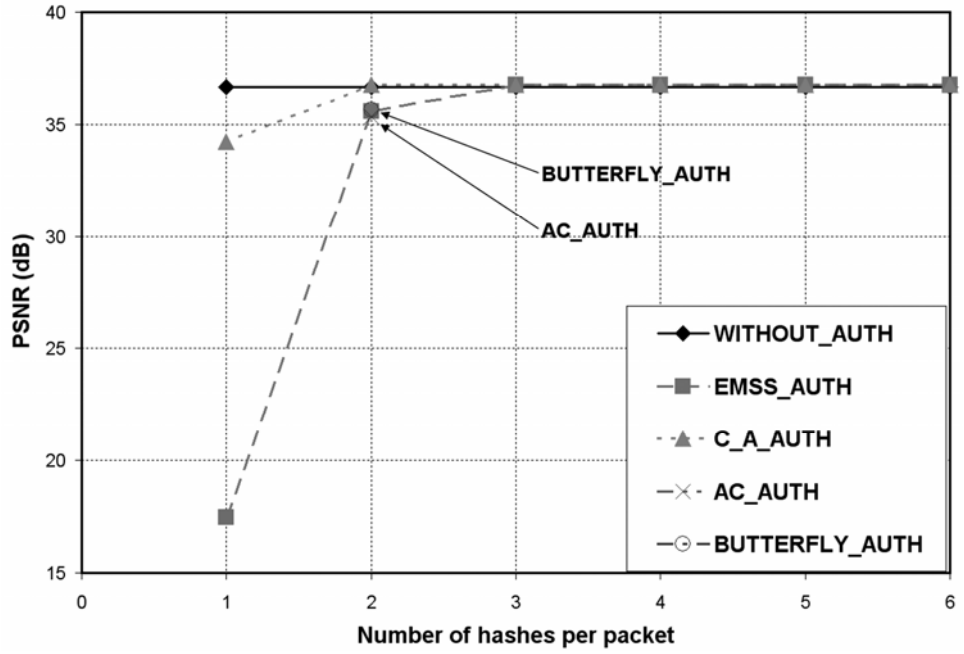

Figure 4-10 – PSNR at various redundancy degrees (loss rate = 0.05, with 6 layers)

The fifth experiment is to measure the minimum overhead required to achieve a PSNR that is 99% of *WITHOUT_AUTH*. The JPEG-2000 codestreams have one layer, for the same reason as the previous experiment. As shown in Figure 4-11, *C_A_AUTH* requires the overhead to be 2 hashes per packet when the loss rate is not greater than 0.1, and requires the overhead to be 3 hashes per packet otherwise. However, *EMSS_AUTH* requires much more overhead in order to maintain the same

PSNR level. The reason for Figure 4-11 is that *C_A_AUTH* is aware of the importance of different packets and optimally allocate the authentication overhead to achieve maximized media quality, while *EMSS_AUTH* simply ignores the unequal packet importance and thereby treats all packets equally.

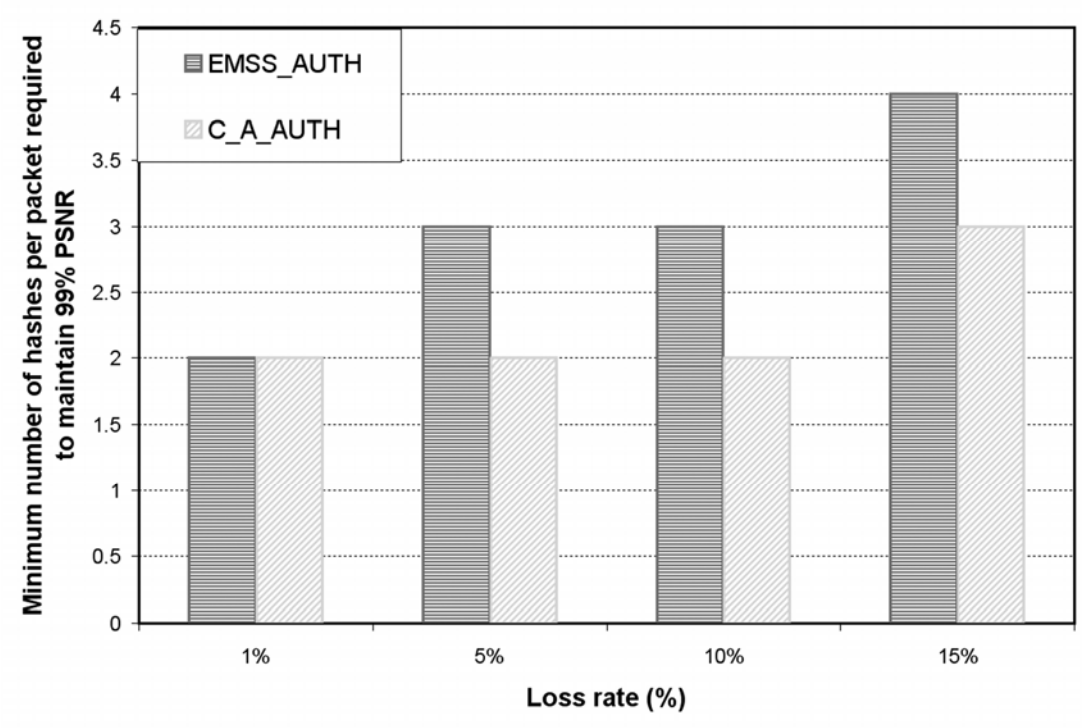

Figure 4-11 – Minimum overhead required to achieve 99% PSNR at various loss rates (with 1 layer)

4.5 CONCLUSIONS

In this chapter, we have proposed an optimized content-aware authentication method, which is able to achieve distortion-overhead optimization by utilizing information about the coding structure and media content. The main contributions are summarized as follows.

- For authenticating media stream delivered over lossy network, instead of optimizing the verification probability, we focus on optimizing authenticated media quality.
- We proposed a distortion-overhead optimization framework for media authentication. Given a specific authentication overhead, the proposed method computes an authentication graph that minimizes the distortion of the authenticated image at the receiver.
- In view that the optimization process has high computational complexity, we also proposed a simplified authentication graph construction that requires much lower complexity to compute.
- Through system analysis and simulation, we demonstrated that the proposed method achieves an R-D curve of the authenticated image which is very close to the R-D curve when no authentication is required, and it substantially outperforms existing schemes at all bit-rates and loss probabilities.

CHAPTER 5 - RATE-DISTORTION-AUTHENTICATION OPTIMIZED MEDIA STREAMING

Recent advances in media streaming include the Rate-Distortion Optimized (*RaDiO*) streaming techniques [35-47], which compute a transmission policy (or transmission schedule) that minimizes the expected end-to-end distortion, subject to a constraint on the average transmission rate. The transmission policy is computed based on the *distortion increment*, *packet size* and *playout deadline* of individual packets. For instance, a more important packet may be given more opportunities for transmission, and a packet with earlier playout deadline may be given higher priority for transmission. As a result, the performance gain of *RaDiO* techniques over heuristic streaming techniques like [51][52][53] is significant.

Stream authentication provides a means for the receiver to verify the authenticity of received media stream, which is achieved by connecting the packets into a DAG. In other words, stream authentication creates dependency between packets, as packet loss may cause some other packets to be unverifiable.

Ideally, we would like to have both R-D optimized media quality, as produced by *RaDiO*, and assurance of media authenticity. However, when the *RaDiO* technique is applied to a media stream protected using stream authentication, it will produce highly sub-optimal performance for authenticated media, because they optimize rate and distortion only. Here, the "rate" includes the data rate only and the "distortion" is measured between the original media and the decoded media (not the authenticated

media). For a media protected with stream authentication, a packet is associated with two additional parameters: *authentication importance* and *overhead size*, besides the *distortion increment*, *packet size* and *playout deadline* as used by *RaDiO*. The *authentication importance* of a packet is the amount of additional distortion due to the unverifiable packets caused by the loss this packet, and the *overhead size* is the size (in bytes) of authentication information appended to this packet (including crypto signature, hashes, or both). Conventional *RaDiO* techniques do not consider authentication and, therefore, are referred to as *authentication-unaware* streaming techniques.

In this chapter, we propose a *Rate-Distortion-Authentication* (*R-D-A*) Optimized streaming technique to achieve optimized quality for authenticated media at the receiver. The *R-D-A* optimization is defined as a rate-distortion optimization for authenticated media, where the "rate" includes the data rate for coded media data and the authentication overhead, and the "distortion" is measured by the difference between the original media and the authenticated media. The *R-D-A* optimized technique is able to achieve optimized performance for authenticated media, as it accounts for the *authentication importance* and *overhead size*, besides the original R-D dependency and parameters.

Given a coded media with an authentication method applied, first we need to compute the quantities associated with each packet. The *distortion increment*, *packet size* and *playout deadline* are the same as that in conventional *RaDiO* techniques. The *overhead size* can be computed from the topology of the authentication graph. The *authentication importance* of a packet depends on the following factors: 1) the packet(s) whose verification is affected by this packet; 2) the *distortion increment* of the affected packets; 3) the loss probability of the affected packets; 4) how much

influence the packet has to the individual affected packets. Second, at every transmission opportunity, the *R-D-A* optimization process selects the best set of packets for transmission based on these parameters. For example, packets with higher importance (*distortion increment* + *authentication importance*) and smaller size (*packet size* + *overhead size*) will be assigned with more transmission opportunities. In summary, we formulate an *R-D-A* optimization problem to minimize the expected end-to-end distortion of the authenticated media at the receiver, subject to the constraint on average transmission rate.

In stream authentication, a packet is useful, if and only if, it is received and verified before its playout deadline, i.e., a packet received or verified after its playout deadline is equivalent to loss. Nevertheless, even if a packet missed its playout deadline, it is still useful for verification of other packets that depends on this packet for verification. Therefore, each packet is actually associated with multiple deadlines: the first one is its own playout deadline, while the others are the playout deadlines of those packets that depend on this packet for verification. First, we examine the simple case in Section 5.1, where each packet is considered having a single deadline, namely, its own playout deadline. A packet received and verified after its playout deadline is discarded and therefore equivalent to loss. Considering that the proposed optimization algorithm has high complexity, we also propose a low-complexity algorithm. In Section 5.2, we examine the case where each packet has multiple deadlines as explained above. Section 5.3 illustrates how to realize the proposed *R-D-A* Optimization framework with various authentication methods. Section 5.4 validates the proposed technique with simulation results.

5.1 R-D-A OPTIMIZATION WITH SINGLE DEADLINE

In single-deadline case, each packet is associated with four parameters: *packet size B*, *distortion increment* Δd, *playout deadline T* and *overhead size O*. The distortion of the authenticated video is reduced by Δd, if and only if, the packet is received and verified before its deadline *T*.

Given that the signature packet P_{SIGN} is received, let φ_n be the set of packets that affects the verification probability of a packet P_n and φ_n is referred to as a *determining set* of the packet P_n. Similarly, let Φ_n be the set of packets whose verification probability is affected by the packet P_n, and Φ_n is referred to as a *dependent set* of the packet P_n. For example in Figure 1-4(a), the *determining set* of P_2 is $\varphi_2 = \{P_0, P_1\}$ and the *dependent set* of P_1 is $\Phi_1 = \{P_2, P_3\}$. Note that for simplicity, P_{SIGN} is not included in the *determining set* of any packet, but of course, all packets depend on P_{SIGN} for verification and this is accounted for separately. We assume that for any packet $P_{n'} \in \Phi_n$, its verification probability is a linear function of the loss probability of packet P_n, as shown in Eq(5.1), where $\alpha_n^{n'}$ and $\beta_n^{n'}$ are positive numbers. The term $\alpha_n^{n'}$ represents P_n's influence on the verification probability of packet $P_{n'}$, i.e. if P_n is lost, the verification probability of $P_{n'}$ will be reduced by $\alpha_n^{n'}$. Note that P_n has no influence on the verification of any packet $P_{n'} \notin \Phi_n$. In Figure 1-4(a), suppose P_{SIGN} is received, if P_1 is received, P_2 will be verified with probability 1; Otherwise, P_2 depends on P_0 for verification, i.e., P_2's verification probability is reduced to $(1-\varepsilon_0)$, where ε_0 is the loss probability of packet P_0. Thus, the influence of

P_1 on P_2's verification probability is the difference, i.e. $\alpha_1^2 = 1-(1-\varepsilon_0) = \varepsilon_0$. Similarly, we can compute P_1's influence on packet P_3, i.e., $\alpha_1^3 = 1$.

$$V_{n'} = -\alpha_n^{n'}\varepsilon(\pi_n) + \beta_n^{n'} \tag{5.1}$$

The overhead size can be easily computed from the topology of the authentication graph. Assuming the hash size is *h* and the signature size is *g*, the overhead size of packet P_n can be expressed as:

$$O_n = e_n h + m_n g \tag{5.2}$$

The term e_n denotes the number of incoming edges to packet P_n, and m_n is 1 if packet P_n is the signature packet and 0 otherwise.

To transmit the given *N* packets (including the signature packet) with transmission policy $\pi = [\pi_{SIGN}, \pi_0, ..., \pi_{N-2}]$, the expected transmission cost is computed by summing up the transmission costs of individual packets, as shown in Eq(5.3). Recall that $\rho(\pi_n)$ is the per byte transmission cost for packet P_n using transmission policy π_n . This is similar to Eq(2.2) with the only exception of the overhead size being added to each packet.

$$R(\pi) = (B_{SIGN} + O_{SIGN})\rho(\pi_{SIGN}) + \sum_{n=0}^{N-2}(B_n + O_n)\rho(\pi_n) \tag{5.3}$$

With transmission policy π , the expected distortion of the authenticated video is expressed in Eq(5.4), where D_0 is the total distortion when no packet is received or verified and V_n denotes the verification probability of packet P_n . Recall that $\varepsilon(\pi_n)$ denotes the loss probability of packet P_n using transmission policy π_n . The distortion of the authenticated video is computed by subtracting every packet's *distortion increment* weighted by its probability of being received and being verifiable before its *playout deadline*.

$$D(\pi) = D_0 - \left(1-\varepsilon\left(\pi_{SIGN}\right)\right)\left(\Delta d_{SIGN} + \sum_{n=0}^{N-2}\Delta d_n\left(1-\varepsilon\left(\pi_n\right)\right)V_n\right) \tag{5.4}$$

Substituting Eq(5.3) and Eq(5.4) into Eq(2.4), we get the Lagrangian cost function

$$\begin{aligned} J\left(\pi\right) = D_0 &+ \left(-\left(1-\varepsilon\left(\pi_{SIGN}\right)\right)\Delta d_{SIGN} + \lambda\left(B_{SIGN}+O_{SIGN}\right)\rho\left(\pi_{SIGN}\right)\right) \\ &+ \sum_{n=0}^{N-2}\left(-\left(1-\varepsilon\left(\pi_{SIGN}\right)\right)\left(1-\varepsilon\left(\pi_n\right)\right)V_n\Delta d_n + \lambda\left(B_n+O_n\right)\rho\left(\pi_n\right)\right) \end{aligned}. \tag{5.5}$$

This R-D-A optimization problem can be solved using an iterative descent algorithm, i.e. optimizing the policy for one packet at a time while keeping the other packets' policy fixed, until the Lagrangian cost function $J\left(\pi\right)$ converges. For instance, the policy can be decided by Eq(5.6) for P_{SIGN} and by (5.7) for P_n.

$$\pi_{SIGN}^{*} = \arg\min_{\pi_{SIGN}} \varepsilon\left(\pi_{SIGN}\right) + \lambda'_{SIGN}\rho\left(\pi_{SIGN}\right) \tag{5.6}$$

$$\pi_n^{*} = \arg\min_{\pi_n} \varepsilon\left(\pi_n\right) + \lambda'_n\rho\left(\pi_n\right) \tag{5.7}$$

where $\lambda'_{SIGN} = \lambda\left(B_{SIGN}+O_{SIGN}\right)/S_{SIGN}$ and $\lambda'_n = \lambda\left(B_n+O_n\right)/S_n$, and where S_{SIGN} and S_n are the sensitivity factors described next. The sensitivity factors, S_{SIGN} and S_n, can be obtained by taking partial derivatives of $D\left(\pi\right)$ with respect to $\varepsilon\left(\pi_{SIGN}\right)$ and $\varepsilon\left(\pi_n\right)$, respectively, as shown in Eq(5.8) and Eq(5.9).

$$S_{SIGN} = \Delta d_{SIGN} + \sum_{n=0}^{N-2}\Delta d_n\left(1-\varepsilon\left(\pi_n\right)\right)V_n \tag{5.8}$$

$$S_n = \left(1-\varepsilon\left(\pi_{SIGN}\right)\right)V_n\Delta d_n + \left(1-\varepsilon\left(\pi_{SIGN}\right)\right)\sum_{P_{n'}\in\Phi_n}\alpha_n^{n'}\left(1-\varepsilon\left(\pi_{n'}\right)\right)\Delta d_{n'} \tag{5.9}$$

The sensitivity factor S_n represents the total expected distortion increment caused by the loss of the packet P_n, which comprises two parts: 1) the expected distortion increment due to the unsuccessful decoding of P_n itself (referred to as *decoding importance*), and; 2) the expected distortion increment due to the reduced

verification probability of all packets in its dependent set Φ_n (referred to as *authentication importance*). This is reflected in Eq(5.9), where the first term is the *decoding importance* SD_n in Eq(5.10) and the second term is the authentication importance SA_n in Eq(5.11).

$$SD_n = \left(1-\varepsilon\left(\pi_{SIGN}\right)\right)V_n \Delta d_n \tag{5.10}$$

$$SA_n = \left(1-\varepsilon\left(\pi_{SIGN}\right)\right)\sum_{P_{n'}\in\Phi_n} \alpha_n^{n'}\left(1-\varepsilon\left(\pi_{n'}\right)\right)\Delta d_{n'} \tag{5.11}$$

In Eq(5.10), the *decoding importance* of P_n is simply the distortion increment multiplied by its verification probability and the receiving probability of the signature packet. Thus, a packet P_n has higher decoding importance if the following criteria are met: 1) P_n has higher distortion increment Δd_n ; 2) P_n has higher verification probability, and 3) P_{SIGN} has lower loss probability.

In Eq(5.11), the *authentication importance* of P_n is the sum of the distortion increments of all packet $P_{n'} \in \Phi_n$ weighted with their respective receiving probability, P_n's influence on $P_{n'}$'s verification and the receiving probability of the signature packet. Thus, a packet P_n has higher authentication importance if the following criteria are met: 1) there are more packets in Φ_n ; 2) the packets in Φ_n have higher distortion increment; 3) the packets in Φ_n have lower loss probability; 4) P_n has higher influence on the packets in Φ_n ; 5) P_{SIGN} has lower loss probability.

The signature packet P_{SIGN} is a special case in that it does not depend on any other packet for verification, and thereby its decoding importance is simply $SD_{SIGN} = \Delta d_{SIGN}$. On the other hand, all other packets depend on P_{SIGN} for verification, i.e. $\Phi_{SIGN} = \{P_0, P_2, ..., P_{N-2}\}$. If P_{SIGN} is lost, no packet will be verified,

thereby its authentication importance is $SA_{SIGN} = \sum_{n=0}^{N-2} \Delta d_n \left(1-\varepsilon\left(\pi_n\right)\right) V_n$. Therefore, the signature packet is the most important packet with the highest sensitivity factor $S_{SIGN} = SD_{SIGN} + SA_{SIGN}$.

From Eq(5.6) and Eq(5.7), the sensitivity factor, together with the total size (including *packet size* and *overhead size*), determines the transmission policy which corresponds to the bandwidth allocation among the packets. In particular, the optimization process accounts for the *distortion increment*, *packet size*, *deadline*, *authentication importance* and *overhead size* for each packet to generate the optimal policy that minimizes the distortion of authenticated video at the receiver. In the resulting policy, a packet will have more transmission opportunities if its Lagrange multiplier is smaller, i.e. smaller size and greater sensitivity factor.

Searching for the optimal transmission policy is computationally expensive, due to the dependency imposed by graph-based authentication. The iterative process has to run for multiple iterations before the Lagrangian value converges. For instance, the complexity of the proposed method is in the order of N_i*N*(C+2^M), where N_i is the number of iterations that the optimization algorithm performs before convergence (typically in the order of 2-5), and N is the number of packets considered for transmission. The term C is the complexity of computing the sensitivity factor for a packet, and it depends on the size of the determining set and dependent set. The term M is the number of transmission opportunities in the Markov decision process. The computational complexity is exacerbated by the fact that the optimization algorithm has to run at every transmission opportunity.

5.1.1 Low-Complexity Optimization Algorithm

We propose a low-complexity algorithm for selecting the packets for transmission. At each transmission opportunity, packets are selected in the following four steps.

1. In the first step, for the signature packets, their current action in the transmission policy is set to "SEND" if they have never been sent before, or their last transmission is more than one round-trip time ago. Note that setting the current action to "SEND" does not necessary mean that it will be transmitted in this transmission opportunity. The steps below may change the action. The purpose of this step is to avoid the deadlock scenario where a packet *A* with higher distortion increment (like I-frame packet in a video stream) depends on another packet *B* with lower distortion increment and higher authentication importance (like P-frame packet serving as signature packet) for verification. At the very beginning when neither packet has been transmitted and their transmission policies are initialized to "NO SEND", the sensitivity factor (to be computed in next step) of packet *B* is very small, because *A* is not yet transmitted and its importance is not reflected in *B*'s sensitivity factor. On the other hand, the sensitivity factor of packet *A* is zero because *A* would not be verified as *B* has not yet been transmitted. Thus, this forms a deadlock scenario, where both packets have small sensitivity factor and, therefore, neither packet has much chance to get transmitted. Note that this is not a problem in the original proposed *R-D-A* optimization algorithm, because it will run for many iterations starting from the initial all-"SEND" policy.
2. In step 2, based on the transmission history of the other packets and the actions set in Step 1, we can compute the sensitivity factor S_n using Eq(5.8) and Eq(5.9), and

hence the sensitivity per unit is $S_n/(B_n+O_n)$, which is the ratio of the packet's sensitivity factor over its total size. It accounts for the authentication importance, overhead size, distortion increment and packet size.

3. In step 3, the sensitivity factor per unit is adjusted based on the transmission history and the remaining time before its playout deadline. If the deadline is less than one forward-trip time away from current time, the adjusted sensitivity per unit cost is set to 0, because it makes no difference to send the packet. Otherwise, the sensitivity per unit cost is multiplied by a factor $\tau=(1-\varepsilon_f)(\varepsilon_f/\varepsilon_r)^x(\varepsilon_f)^y$, where ε_f and ε_r are forward and round-trip loss probabilities, x is the number of previous transmissions that are more than one round-trip time ago and y is the number of previous transmissions that are less than one round-trip time ago. The sensitivity factor is discounted by $\varepsilon_f/\varepsilon_r$ for each transmission that is more than one round-trip time ago and by ε_f for each transmission that is less than one round-trip time ago.
4. Finally, in step 4 the packets are sorted in decreasing order of adjusted sensitivity per unit. Based on this rank ordering, we choose to send the first k packets whose size in total does not exceed the transmission budget.

This low-complexity optimization algorithm requires no iteration and it only needs to compute the sensitivity per unit for every packet. At each transmission opportunity, the complexity is $N*C$, where N is the number of considered packets and C is the complexity to compute a sensitivity factor. Recall that the R-D-A Optimization algorithm has a complexity of $N_i*N*(C+2^M)$, where N_i is the number of iterations, and M is the number of transmission opportunities in the Markov decision

process. Therefore, the proposed low-complexity algorithm has significantly reduced complexity.

5.2 R-D-A OPTIMIZATION WITH MULTIPLE DEADLINES

Using graph-based stream authentication, a packet P_n may be used to verify a number of other packets, which are referred to as the *dependent set* Φ_n of P_n. Therefore, assuming all packets in Φ_n have a later playout deadline, the packet P_n is associated with $|\Phi_n|+1$ deadlines, the first one is its own playout deadline and the others are playout deadlines of the packets in Φ_n. Accordingly, the packet P_n will have $|\Phi_n|+1$ error probabilities, for instance, $\varepsilon(\pi_n, n')$ is used to denote the probability that P_n does not arrive by the playout deadline of packet $P_{n'}$.

As in the *R-D-A* optimization with a single deadline, we still use an iterative descent algorithm to search for the optimized transmission policy. For each packet, the optimized transmission policy is determined by Eq(5.12).

$$\pi_n^* = \arg\min_{\pi_n} \rho(\pi_n) + \sum_{P_{n'}=P_n \ or \ P_{n'}\in\Phi_n} v_{n,n'}\varepsilon(\pi_n, n') \tag{5.12}$$

The term $v_{n,n'}$ can be computed by $v_{n,n'} = \left(SA_{n,n'} + SD_{n,n'}\right)\big/\lambda\left(B_n + O_n\right)$, where $SA_{n,n'}$ and $SD_{n,n'}$ are the *authentication importance* and *decoding importance* of packet P_n with respect to the playout deadline of $P_{n'}$. Note that the decoding importance is zero for all deadlines except the first one, as we assume the error concealment and coding dependency are implicitly accounted for by the distortion increment Δd_n as in [41]. The decoding importance is computed using Eq(5.13), where V_n is the

verification probability of P_n and $\varepsilon(\pi_{SIGN}, n')$ is the error probability of the signature packet with respect to the playout deadline of $P_{n'}$.

$$SD_{n,n'} = \begin{cases} \left(1-\varepsilon(\pi_{SIGN}, n')\right) V_n \Delta d_n & n = n' \\ 0 & otherwise \end{cases} \tag{5.13}$$

The arrival of packet P_n will benefit the verification of the packets in Φ_n with playout deadline later than or equal to the arrival time of P_n. Thus, the authentication importance $SA_{n,n'}$ can be computed using Eq(5.14), where $\alpha_n^{n''}$ is the influence packet P_n has on the verification probability of packet $P_{n''}$.

$$SA_{n,n'} = \left(1-\varepsilon(\pi_{SIGN}, n')\right) \sum_{P_{n''}: T_{n''} = T_{n'}} \alpha_n^{n''} \left(1-\varepsilon(\pi_{n''}, n')\right) \Delta d_{n'} \tag{5.14}$$

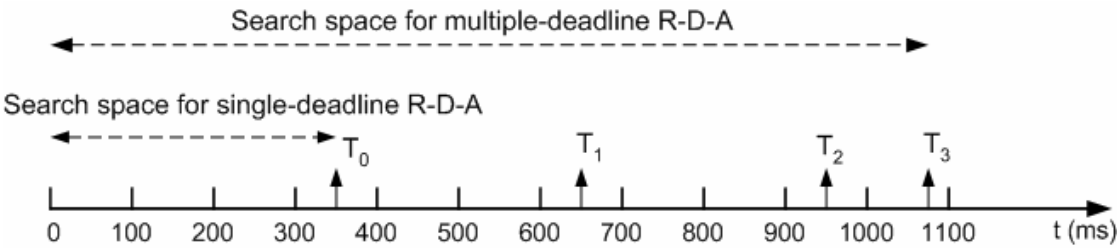

Figure 5-1 – Search space in single-deadline and multiple-deadline R-D-A optimization (transmission interval = 100ms)

Suppose a packet has 4 deadlines, out of which the first deadline is its playout deadline T_0, as shown in Figure 5-1. In single-deadline *R-D-A* optimization, the packet is only considered for transmission in the first 4 transmission opportunities, while in multiple-deadline R-D-A optimization, it is still considered for transmission even after T_0. This will increase the verification probability of other packets with dependence on the given packet, and therefore improve the quality of the authenticated video.

On the other hand, the consideration of multiple deadlines drastically increases the complexity, as the search space grows exponentially with the number of transmission opportunities. This is illustrated in Figure 5-1, where in this example the multiple-deadline R-D-A optimization has to search 2^{10} possible transmission policies, compared with 2^4 possibilities for the single-deadline case.

5.2.1 Low-complexity Optimization Algorithm

We next examine two methods to reduce the complexity associated with multiple-deadline R-D-A optimization. The first method (referred to as *window-split* method) is to split the transmission window into two segments: the first segment lasts until T_0-k and the second segment starts from T_0-k and ends at the last deadline, where k is the minimum forward propagation delay. In each time segment, the R-D-A optimization searches transmission possibilities within the segment only, although it still accounts for multiple error probabilities with respect to all deadlines. This greatly reduces the search space, e.g. the search space is reduced from 2^{10} to 2^4+2^6 in Figure 5-1.

The single-deadline *R-D-A* optimization method seldom transmits a packet in the time interval [T_0-t_{avg}, T_0], as transmission in this interval does not increase the probability of being received before T_0, where t_{avg} is the average forward transmission time. However, the window-split method still transmits the packet in this interval due to two reasons: 1) The consideration of multiple deadlines increases the authentication importance of those packets with a large number of dependent packets and, therefore, increases its chance of being transmitted; 2) In the first segment, the window-split method assumes no transmission after deadline T_0 (while, in fact, there can still be transmissions after T_0) and, therefore, it forces the transmission of a packet even as it

approaches T_0, adversely effecting other packets whose deadlines have not yet been reached.

We also find that the gain from transmitting a packet after its first deadline T_0 decays very fast. Therefore, we propose the second method (referred to as *extended-window* method) to simply extend the transmission window to time T_W, where $T_0 \leq T_W \leq T_M$, where T_M is the last deadline. A packet is not considered for transmission after T_W. The length of the extended window is chosen so that it has acceptable complexity while still maintaining most of the gain from the use of multiple deadlines.

5.3 R-D-A OPTIMIZATION WITH SPECIFIC AUTHENTICATION METHODS

This section describes how to realize the proposed *R-D-A* Optimization with various graph-based authentication methods. In particular, we illustrate the computation of *decoding importance*, *authentication importance* and *overhead size*, which can be substituted into Eq(5.6) and Eq(5.7) to compute the optimized transmission policy.

In the previous sections, we assume that for any packet $P_{n'} \in \Phi_n$, its verification probability is a linear function of the loss probability of packet P_n, as in Eq(5.1). This is true for Simple Hash Chain, Tree-authentication and Butterfly authentication, which can be directly incorporated into the proposed R-D-A Optimization framework. However, for authentication methods like EMSS and Augmented Chain, it is more difficult to compute the verification dependency and authentication importance in closed form. Nevertheless, one can still take a

simulation-based empirical approach to estimate the verification probability and authentication importance.

5.3.1 R-D-A Optimization with Tree-Authentication

With Tree-authentication, each packet carries the signature and the hashes of its sibling nodes along its paths to the root. Thus, each packet is individually verifiable, i.e. there is no authentication dependency among the packets. As such, for a packet P_n, its *authentication importance* is 0 (SA_n=0), its *decoding importance* is simply the associated distortion increment ($SD_n = \Delta d_n$), and its overhead size is $g+h*\log_2 N$, where N is the total number of packets. Therefore, in this case, the *R-D-A* Optimization is very similar to conventional *RaDiO* with the only exception of authentication overhead.

Note that Tree-authentication does not impose any authentication dependency, resulting in very low complexity. In this case, the complexity is the same as lower-complexity *RaDiO* using DC^0 [39]. At each transmission opportunity, we simply sort the packets in decreasing order of $\Delta d_n / (B_n + O_n)$ and choose to send the first k packets whose size in total does not exceed the transmission budget.

5.3.2 R-D-A Optimization with Simple Hash Chain

Given N packets connected as a hash chain as shown in Figure 2-3, a packet P_n can be verified, if and only if, all preceding packets are received. The *determining set* of P_n is $\varphi_n = \{P_{n'} \mid 0 \le n' < n\}$, the *dependent set* of P_n is $\Phi_n = \{P_{n'} \mid n < n' \le N-2\}$, and the

verification probability of P_n is $V_n = \sum_{n'=0}^{n-1}\left(1-\varepsilon\left(\pi_{n'}\right)\right)$. Therefore, the decoding importance of P_n can be computed using Eq(5.10). Furthermore, for any packet $P_{n'} \in \Phi_n$, its verification probability $V_{n'}$ is a linear function of $\varepsilon\left(\pi_n\right)$, as shown in Eq(5.15).

$$V_{n'} = -\alpha_n^{n'}\varepsilon\left(\pi_n\right) + \beta_n^{n'} \qquad where \quad \alpha_n^{n'} = \beta_n^{n'} = \sum_{\substack{0\le n''<n' \\ n''\neq n}}\left(1-\varepsilon\left(\pi_{n''}\right)\right) \tag{5.15}$$

Substituting Eq(5.15) into Eq(5.11), we can obtain the authentication importance. The overhead size is g+h for the signature packet P_{sig}, 0 for the last packet $P_{N\text{-}2}$, and h for the rest.

Therefore, the decoding importance, authentication importance and overhead size can be substituted into Eq(5.6) and Eq(5.7) to compute the optimized transmission policy.

5.3.3 R-D-A Optimization with Butterfly Authentication

Given N packets (assuming $N = N_R N_C + 1$ and $N_C = \log_2 N_R + 1$) connected as a butterfly authentication graph, the first packet is the signature packet, the rest of the packets are divided into N_C columns and each column has N_R packets. Let us denote packet P_n as $P_{\text{c,r}}$, where c is the column number, r is the packet index within a column, and $n = cN_R + r$. The signature packet P_{SIGN} contains the hashes of all packet in column 0, packet $P_{c,r}$ has its hash appended to $P_{c\text{-}1,r}$ and $P_{c-1,r'}$, where r and r' are $\left(N_C - 1\right)$-bit numbers differing only at the $(c\text{-}1)^{th}$ most significant bit. $P_{c,r}$ is verifiable if either $P_{c\text{-}1,r}$ or $P_{c-1,r'}$ is received and verified. Thus, its verification probability $V_{c,r}$

can be expressed using the loss probability and verification probability of connected packets in the previous column, as in Eq(5.16).

$$V_{c,r}=\begin{cases}\begin{pmatrix}\left(1-\varepsilon\left(\pi_{c-1,r}\right)\right)V_{c-1,r}+\left(1-\varepsilon\left(\pi_{c-1,r'}\right)\right)V_{c-1,r'}\\ -\left(1-\varepsilon\left(\pi_{c-1,r}\right)\right)V_{c-1,r}\left(1-\varepsilon\left(\pi_{c-1,r'}\right)\right)V_{c-1,r'}\end{pmatrix} & when\quad 0<c<N_C\\ 1 & when\quad c=0\end{cases} \tag{5.16}$$

For packet $P_{c,r}$, its *determining set* $\varphi_{c,r}$ includes all packets to which $P_{c,r}$ has a directed path in the butterfly graph, i.e., $\varphi_{c,r}=\left\{P_{c',r'}\mid 0\le c'<c, P_{c,r}\xrightarrow{path}P_{c',r'}\right\}$; its *dependent set* $\Phi_{c,r}$ includes all packets that has a path to $P_{c,r}$, i.e. $\Phi_{c,r}=\left\{P_{c',r'}\middle| c<c'\le \log_2 K, P_{c',r'}\xrightarrow{path}P_{c,r}\right\}$. For example, in Figure 3-1, $\varphi_{1,1}=\left\{P_{0,1},P_{0,5}\right\}$ and $\Phi_{1,1}=\left\{P_{2,1},P_{2,3},P_{3,0},P_{3,1},P_{3,2},P_{3,3}\right\}$.

By iteratively applying Eq(5.16), $V_{c,r}$ can be expressed using loss probabilities of all packets in its *determining set* $\varphi_{c,r}$, which can be substituted into Eq(5.10) to compute the decoding importance of packet $P_{c,r}$.

Lemma 1: In the butterfly authentication graph, if packet $P_{c',r'}$ depends on packet $P_{c,r}$ for verification, i.e. $P_{c',r'}\in\Phi_{c,r}$, the dependent set of $P_{c',r'}$ is a subset of the dependent set of $P_{c,r}$, i.e. $\Phi_{c',r'}\subseteq\Phi_{c,r}$. Similarly, if the packet $P_{c,r}$ depends on $P_{c',r'}$ for verification, i.e. $P_{c',r'}\in\varphi_{c,r}$, the determining set of $P_{c',r'}$ is a subset of the determining set of $P_{c,r}$, i.e. $\varphi_{c',r'}\subseteq\varphi_{c,r}$.

Proof: The first statement can be proved as follows: Let $P_{c'',r''}$ be any packet in $\Phi_{c',r'}$. As $P_{c'',r''}$ has a path to $P_{c',r'}$ which in turn has a path to $P_{c,r}$, $P_{c'',r''}$ also has a path to $P_{c,r}$, i.e. $P_{c'',r''}\in\Phi_{c,r}$. Therefore, $\Phi_{c',r'}\subseteq\Phi_{c,r}$. The second statement can be proved

as follows: Let $P_{c'',r''}$ be any packet in $\varphi_{c',r'}$. As $P_{c,r}$ has path to $P_{c',r'}$ which in turn has path to $P_{c'',r''}$, there is path from $P_{c,r}$ to $P_{c'',r''}$, i.e. $P_{c'',r''} \in \varphi_{c,r}$. Therefore, $\varphi_{c',r'} \subseteq \varphi_{c,r}$.

The dependency relationship in Butterfly authentication graph has the transition property. In short, if packet *A* depends on packet *B* which in turn depends on packet *C*, then *A* also depends on *C*.

Lemma 2: In a Butterfly authentication graph, for any packet $P_{c',r'}$ in the dependent set of $P_{c,r}$, i.e., $P_{c',r'} \in \Phi_{c,r}$, its verification probability $V_{c',r'}$ can be expressed as a linear function of the loss probability of $P_{c,r}$, i.e., $V_{c',r'} = -\alpha_{c,r}^{c',r'} \varepsilon\left(\pi_{c,r}\right) + \beta_{c,r}^{c',r'}$, where $\alpha_{c,r}^{c',r'}$ and $\beta_{c,r}^{c',r'}$ are positive numbers.

Proof: This lemma can be proved using the induction method in two steps: First, when $c' = c+1$, for any packet $P_{c+1,r'} \in \Phi_{c,r}$, it is obvious from Eq(5.16) that $V_{c+1,r'} = -\alpha_{c,r}^{c+1,r'} \varepsilon\left(\pi_{c,r}\right) + \beta_{c,r}^{c+1,r'}$ with $\alpha_{c,r}^{c+1,r'} = V_{c,r}\left(1 - V_{c,k}\left(1-\varepsilon\left(\pi_{c,k}\right)\right)\right)$ and $\beta_{c,r}^{c+1,r'} = V_{c,r} + V_{c,k}\left(1-\varepsilon\left(\pi_{c,k}\right)\right) - V_{c,r}V_{c,k}\left(1-\varepsilon\left(\pi_{c,k}\right)\right)$, where k and r are $\left(N_C - 1\right)$-bit numbers that differ only at the c^{th} most significant bit. Thus, the statement is true when $c' = c+1$. Second, suppose the statement is true for $P_{c',r'} \in \Phi_{c,r}$, i.e. $V_{c',r'} = -\alpha_{c,r}^{c',r'} \varepsilon\left(\pi_{c,r}\right) + \beta_{c,r}^{c',r'}$. For any packet $P_{c'+1,r''} \in \Phi_{c',r'}$, it is obvious from Eq(5.16) that $V_{c'+1,r''} = aV_{c',r'} + b$, where a and b are positive numbers. Thus, the verification probability of $P_{c'+1,r''}$ can be written as $V_{c'+1,r''} = -\alpha_{c,r}^{c'+1,r''} \varepsilon\left(\pi_{c,r}\right) + \beta_{c,r}^{c'+1,r''}$, where $\alpha_{c,r}^{c'+1,r''} = a\alpha_{c,r}^{c',r'}$ and $\beta_{c,r}^{c'+1,r''} = a\beta_{c,r}^{c',r'} + b$. From *Lemma 1* we know that $P_{c'+1,r''} \in \Phi_{c,r}$. Therefore, this Lemma is proved.

Lemma 2 says that the authentication probability of a packet in the dependent set of packet $P_{c,r}$ can be expressed as a linear function of the loss probability of packet $P_{c,r}$. Therefore, we can compute packet $P_{c,r}$'s influence $\alpha_{c,r}^{c',r'}$ on any packet $P_{c',r'} \in \Phi_{c,r}$, which can be substituted into Eq(5.11) to compute the authentication importance of $P_{c,r}$.

In a Butterfly authentication graph with $N = N_R N_C + 1$ packets, the signature packet contains the digital signature and hashes of all packets in column-0, thereby the *Overhead Size* of P_{SIGN} is $O_{SIGN} = g + N_R h$, where g and h denote signature size and hash size, respectively. The packets in the last column do not contain any hash, and the rest of the packet contains two hashes each.

$$O_{c,r} = \begin{cases} 2h & 0 \le c < \log_2 K \\ 0 & c = \log_2 K \end{cases} \tag{5.17}$$

Therefore, we can substitute the decoding importance, authentication importance and overhead size into Eq(5.6) and Eq(5.7) to compute the optimal transmission policy.

5.4 ANALYSIS AND EXPERIMENTAL RESULTS

In this section, the proposed R-D-A Optimization technique is benchmarked against 1) *authentication-unaware RaDiO* technique coupled with graph-based authentication, and; 2) straightforward streaming of video data protected by a graph-based authentication method.

5.4.1 Experiment Setup

We consider a video streaming scenario where every received packet is acknowledged by the receiver and re-transmission is driven by the sender. A packet is discarded if it is not delivered or verified before its deadline. Our experiments use the same settings to [38][41][39] for media streaming. Specifically, the network is assumed to be a packet-erasure channel, where a packet is either correctly received or lost. The packet loss and delay are random and independent in the forward and backward channel. Packet loss follows a uniform distribution (the packet loss rate is denoted by ε) while packet delay follows a shifted Gamma distribution with parameter k (constant delay in the network path), n (number of routers in the network path), $1/\alpha$ (mean queuing delay per router), and $1/\alpha^2$ (variance of the queuing delay per router). In our experiments, the forward and backward channels are modeled as having the same loss rate and delay distribution: the loss rate ε is set to 0.03, 0.05, 0.1 and 0.2, as recommended by JVT for error resilience testing [87], the delay parameters are set to k = 50ms, n = 2 and $1/\alpha$ = 25ms. The interval between two consecutive transmission opportunities is 100ms and playout delay is δ = 600ms. At any time t, only those packets whose deadline is in window $[t+k, t+k+\delta]$ are eligible for transmission. NS-2 [89] is used for the simulation. For *RaDiO* streaming, the Lagrange multiplier λ is used to control the transmission rate (recall that smaller λ results in higher transmitted bit rate) and is fixed for one streaming session.

Two QCIF video sequences, *Foreman* (400 frames) and *Container* (300 frames), are encoded using H.264/ACV reference software JM 10.2 [90] at around 150Kbps and 70Kbps respectively. We select these two sequences in our experiment, namely *Foreman* with fast motion, and *Container* with slow motion. The frame rate is

30 frames per second and each GOP comprises one I-frame followed by 14 P-frames. For the convenience of network transmission, each frame is divided into slices (or slice NAL units) based on the coding length. As such, an I-frame may be divided into more than one slice while a P-frame may comprise one slice only. A slice is wrapped by one RTP packet, similar to the single NAL unit mode in [91]. The parameter sets (including sequence parameter set and picture parameter set) are transmitted out-of-band. Other NAL unit types, like SEI and EOS [6] are not used here. In addition, no slice NAL unit shall exceed 1200 bytes and the network MTU is set to 1500 bytes. The space of 300 bytes is reserved for authentication overhead (signatures and hashes appended) and RTP/UDP/IP headers (around 40 bytes). Therefore, no packet segmentation is required in the network.

For authentication, we use SHA-1 Hash (128-bit) and RSA Signature (1024-bit) [18] to construct authentication graphs like EMSS, Augmented Chain and Butterfly, which are all configured with their respective optimal parameters. A signature is amortized among a group of 33 consecutive packets, corresponding to around one-second of video data. As such, for an overhead size of 2 hashes per packet and frame rate of 30 frames per second, the authentication overhead constitutes around 8Kbps of extra data rate, on top of the data rate of the original video.

In total, we implemented six systems described as follows:

1. The first system, *Dumb-AC*, implements a straightforward transmission of video packets protected with Augmented Chain which is claimed optimal for generic data stream [34]. If there is sufficient bandwidth, the sender will re-transmit the packets that has been sent but not yet acknowledged. *Dumb-AC* is the baseline system in our experiments.

2. The second system, *RaDiO*, implements authentication-unaware *RaDiO* for un-authenticated video, whose performance is used as the upper bound for all other systems. We measure the R-D performance for un-authenticated video with 1) no loss and no delay; 2) loss but no delay, and; 3) loss and delay, which can demonstrate the impact of network loss and delay on the R-D performance.
3. The third system, *R-D-A-Opt-Butterfly*, implements our proposed R-D-A optimized streaming and Butterfly Authentication. We choose Butterfly Authentication because Simple Hash Chain is not robust against packet loss while Tree-authentication has too high authentication overhead. The performance of *R-D-A-Opt-Butterfly* is used to validate our proposed R-D-A Optimization technique.
4. The fourth system, *RaDiO-Butterfly*, implements authentication-unaware *RaDiO* and Butterfly Authentication. This system is used to benchmark the R-D-A Optimization technique.
5. The fifth system, *RaDiO-EMSS*, implements authentication-unaware *RaDiO* and EMSS.
6. The sixth system, *RaDiO-AC*, implements authentication-unaware *RaDiO* and Augmented Chain.

These last two systems are used to demonstrate that the proposed R-D-A Optimization outperforms authentication-unaware *RaDiO* not only for Butterfly Authentication but also for other authentication methods, out of which Augmented Chain is claimed to be optimal for generic data authentication. In our experiments, EMSS and Augmented Chain were slightly modified. The modified graphs have exactly the same structure as the original ones, except that the signature packet is the first packet (instead of the last one) and the original forward edges are pointing

backward. This minor modification has two advantages: 1) the authentication dependency is to align with the frame prediction dependency, because the edges point backward; 2) This modification helps to reduce receiver delay, although it also increases sender delay. Recall that for media streams that are pre-stored at the sender, receiver delay is more critical than sender delay.

Note that Augmented Chain and Butterfly Authentication have fixed overhead size (i.e. 2 hashes per packet) while EMSS has tunable overhead size. As such, in the *RaDiO-EMSS* system, we empirically determine the optimal overhead size that produces the best R-D performance. Figure 5-2 and Figure 5-3 give the R-D curves of *RaDiO-EMSS* at different overhead sizes (2-5 hashes per packet) and different loss rates (0.03, 0.1 and 0.2) for *Foreman* and *Container*, respectively. Note that due to space limitation we omit the set of R-D curves at loss rate 0.05, which has a similar R-D performance gap. With higher overhead size, the packets have higher verification probability. On the other hand, the extra overhead also shifts its R-D curve to the right, because overhead is counted toward the total bit rate. The benefit gained from the higher verification probability does not compensate for the bit rate increment due to extra overhead. Therefore, increasing the overhead size does not improve its R-D performance for *RaDiO-EMSS*. In subsequent experiments, we use the optimal overhead size of 2 hashes per packet for *RaDiO-EMSS*.

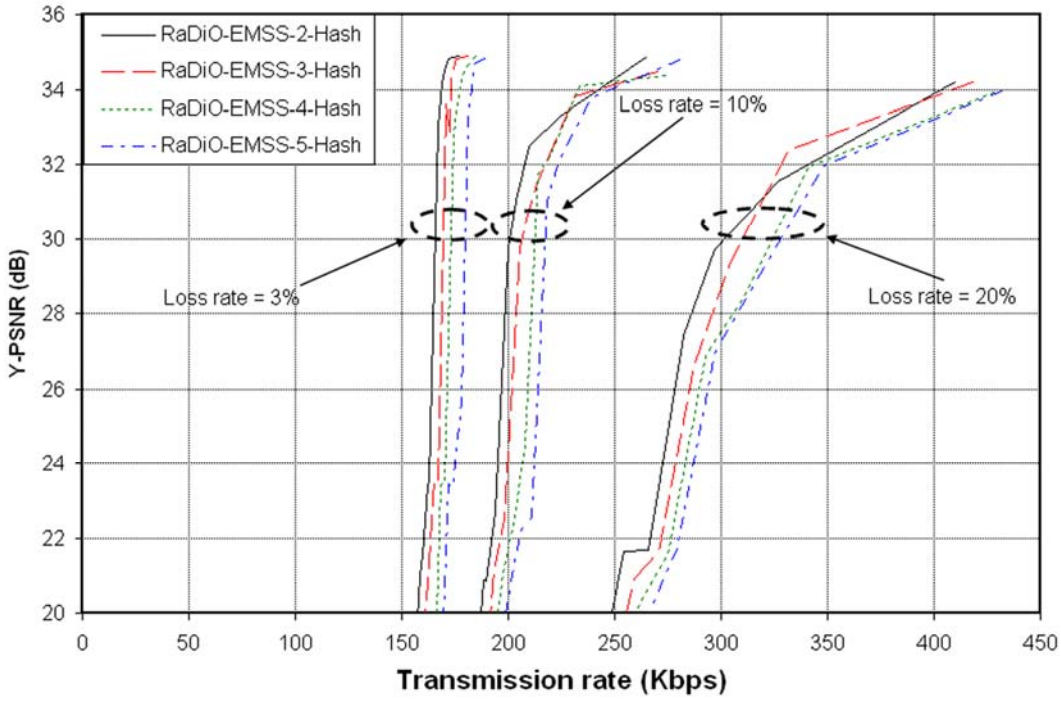

Figure 5-2 – Authentication-unaware RaDiO and EMSS authentication at different overhead sizes and different packet loss rates (0.03, 0,1 and 0.2), Foreman QCIF

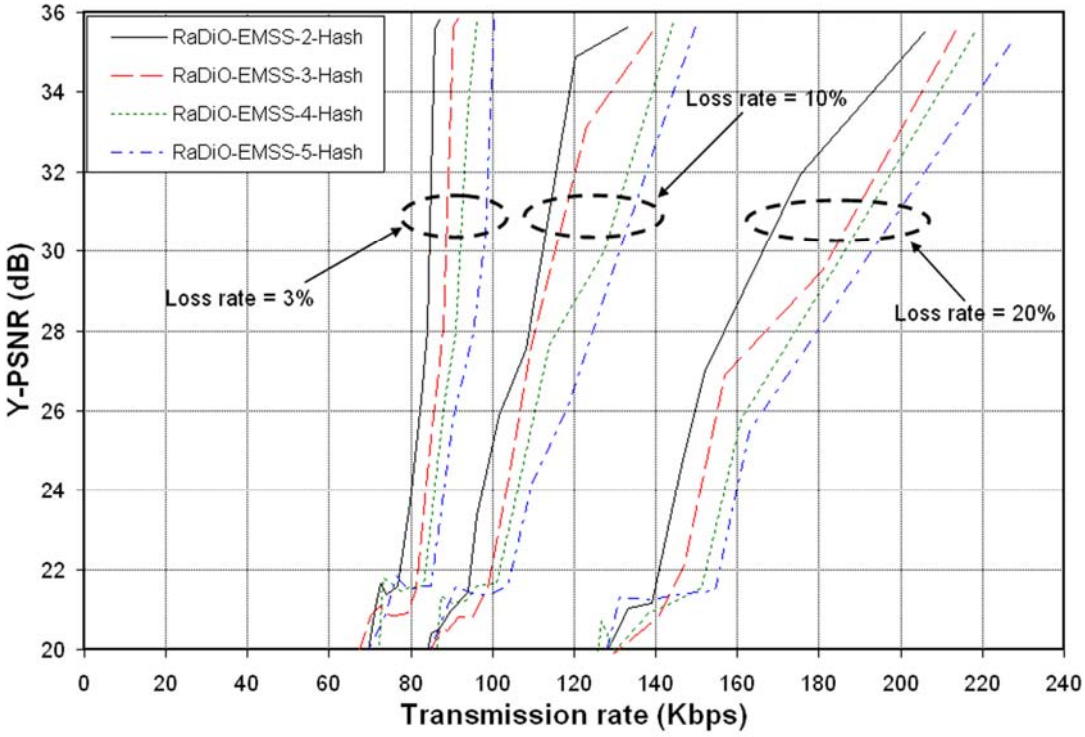

Figure 5-3 – Authentication-unaware RaDiO and EMSS authentication at different overhead sizes and different packet loss rates (0.03, 0,1 and 0.2), Container QCIF

5.4.2 R-D-A Optimization with Single Deadline

The R-D performance of the six systems are given in Figure 5-4 (ε=0.03, *Foreman*), Figure 5-5 (ε=0.03, *Container*), Figure 5-6 (ε=0.05, *Foreman*), Figure 5-7 (ε=0.05, *Container*), Figure 5-8 (ε=0.1, *Foreman*), Figure 5-9 (ε=0.1, *Container*), Figure 5-10 (ε=0.2, *Foreman*) and Figure 5-11(ε=0.2, *Container*). The authentication-unaware techniques like *RaDiO-EMSS*, *RaDiO-AC* and *RaDiO-Butterfly* do not perform well at low rates, as the Y-PSNRs drop quickly to unacceptable levels due to the lack of awareness of authentication. When bandwidth is scarce, packets with smaller distortion increments will have less transmission opportunities, and thereby lead to low probability of reception. However, these packets might be very important for verifying other packets and their loss will greatly degrade the video quality. The steep slope and quick dropoff in performance for the authentication-unaware *RaDiO* techniques may be reduced by increasing the packets' verification probability, but this would require significant additional authentication overhead which would negatively impact the overall R-D performance. This is also demonstrated in Figure 5-2 and Figure 5-3 which shows the R-D curves of *RaDiO-EMSS* with different overhead sizes. At higher overhead size, although the performance dropoff is slightly slower, the R-D curve is shifted towards the right (lower performance).

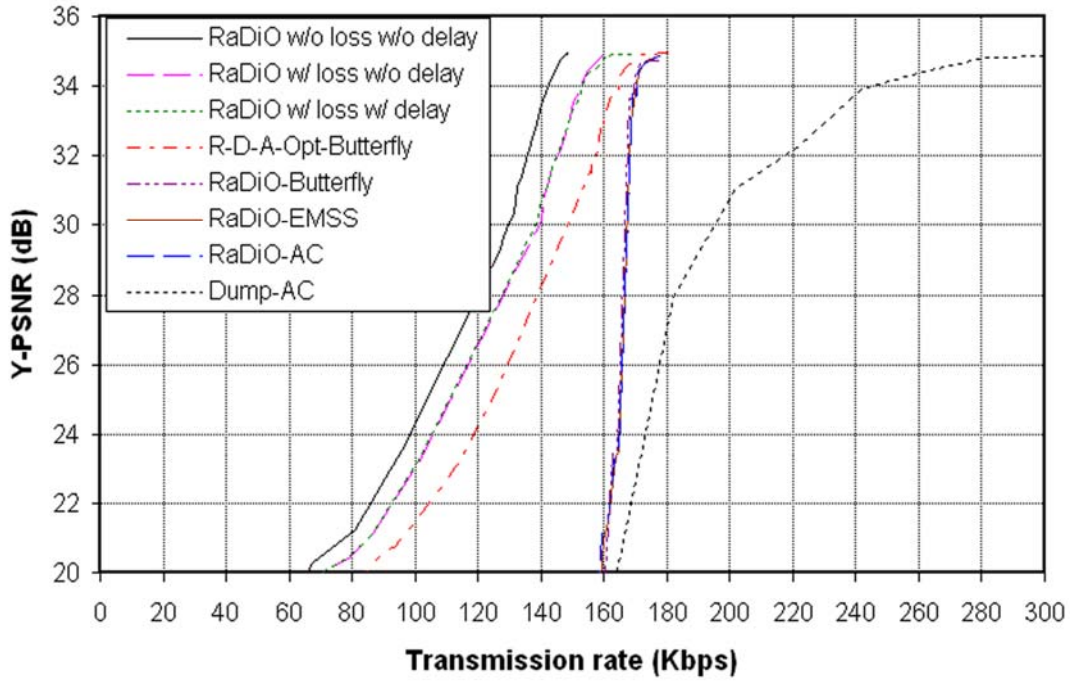

Figure 5-4 – R-D curves for various systems (packet loss rate = 0.03), Foreman QCIF

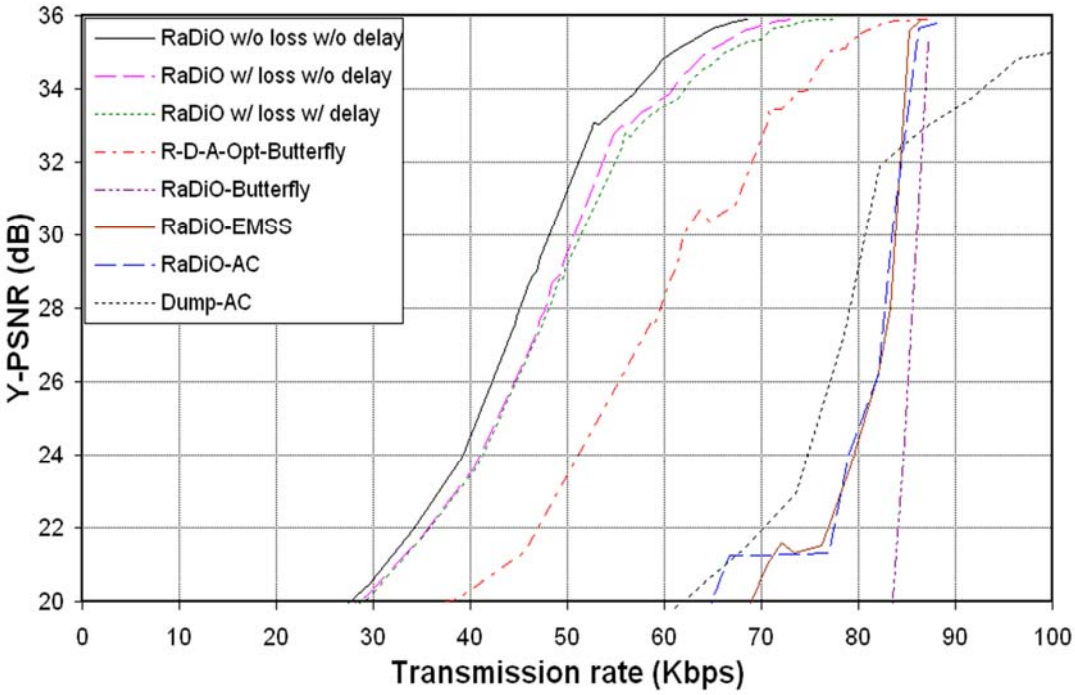

Figure 5-5 – R-D curves for various systems (packet loss rate = 0.03), Container QCIF

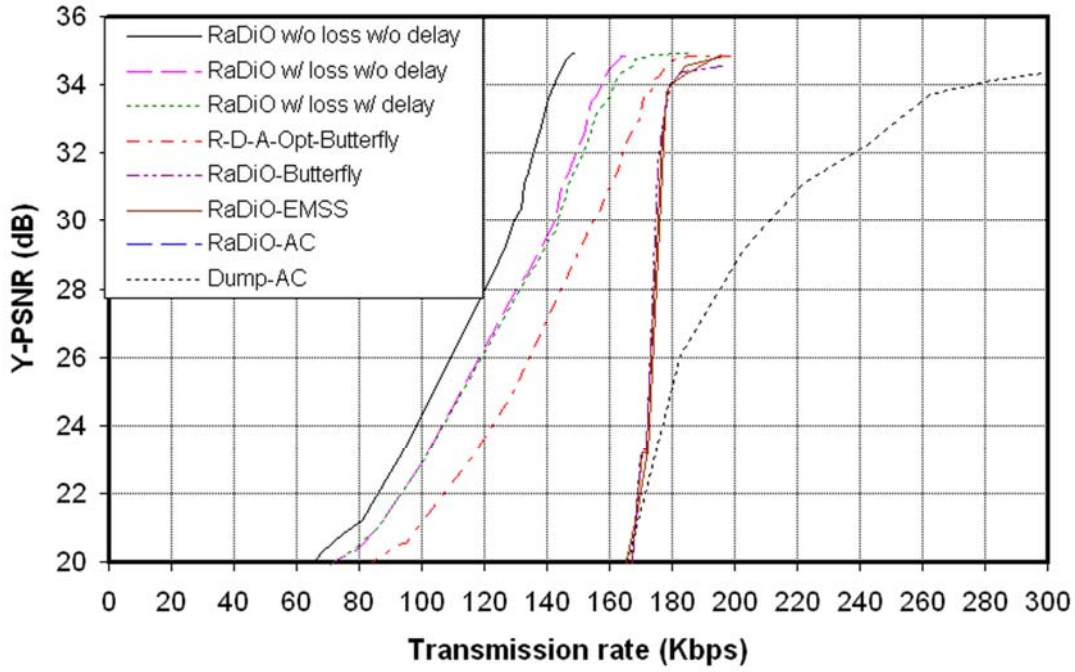

Figure 5-6 – R-D curves for various system (packet loss rate = 0.05), Foreman QCIF

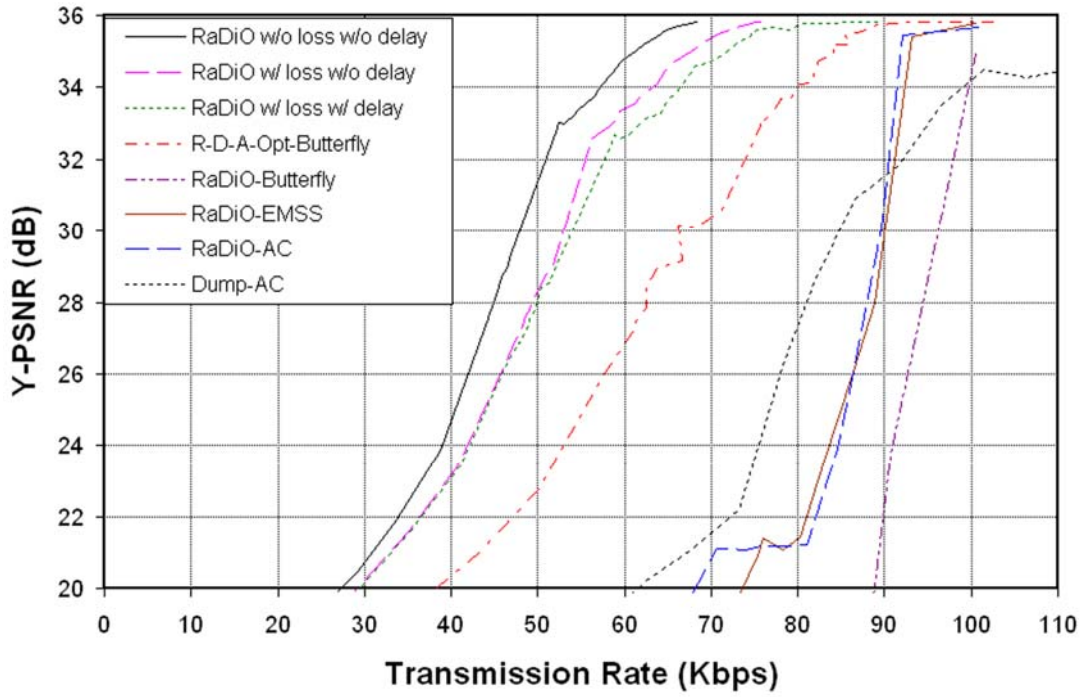

Figure 5-7 – R-D curves for various system (packet loss rate = 0.05), Container QCIF

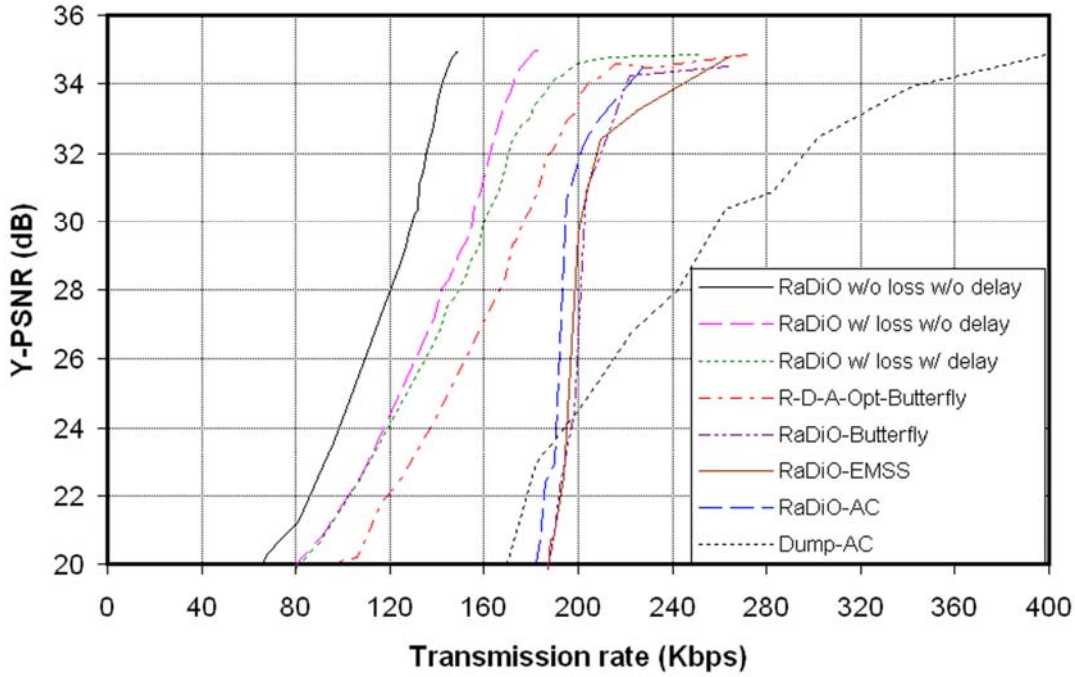

Figure 5-8 – R-D curves for various system (packet loss rate = 0.1), Foreman QCIF

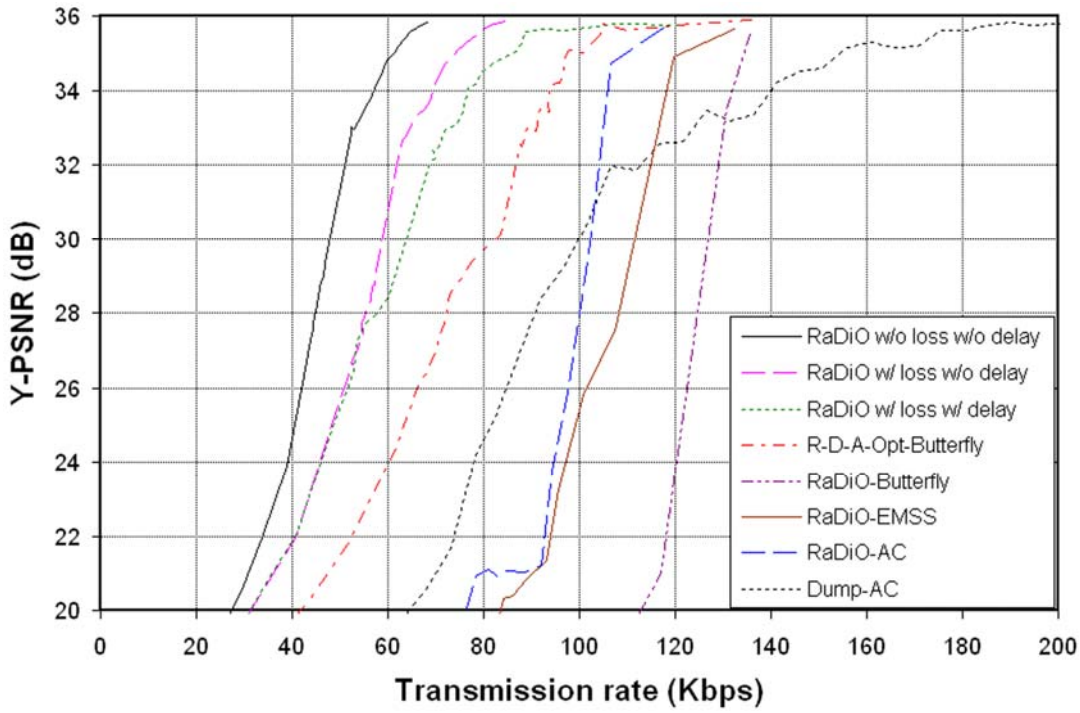

Figure 5-9 – R-D curves for various system (packet loss rate = 0.1), Container QCIF

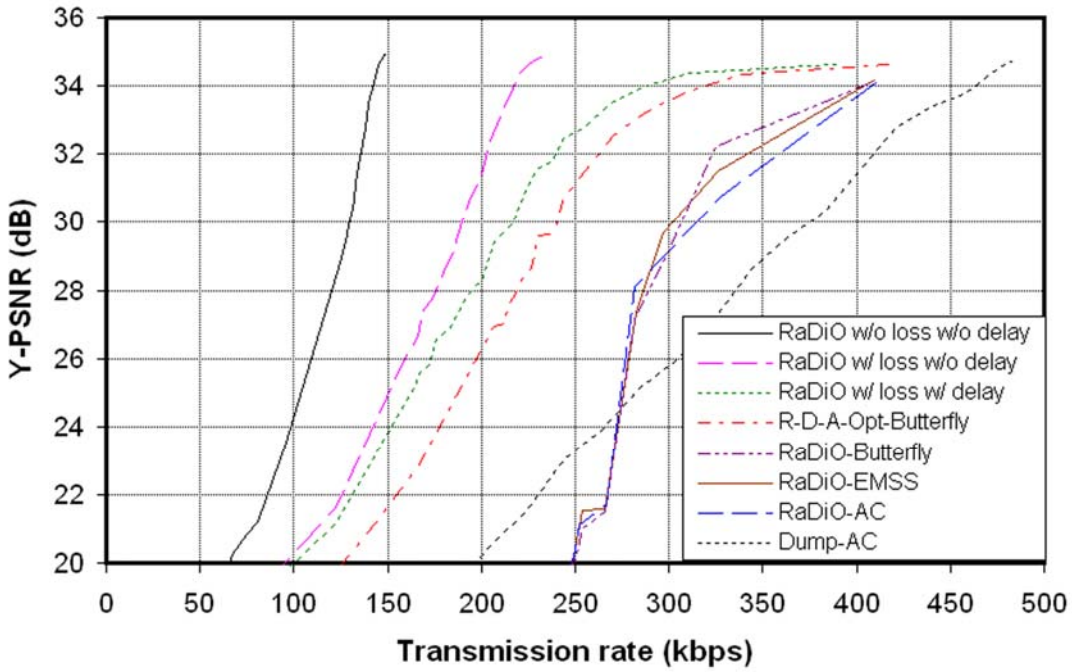

Figure 5-10 – R-D curves for various system (packet loss rate = 0.2), Foreman QCIF

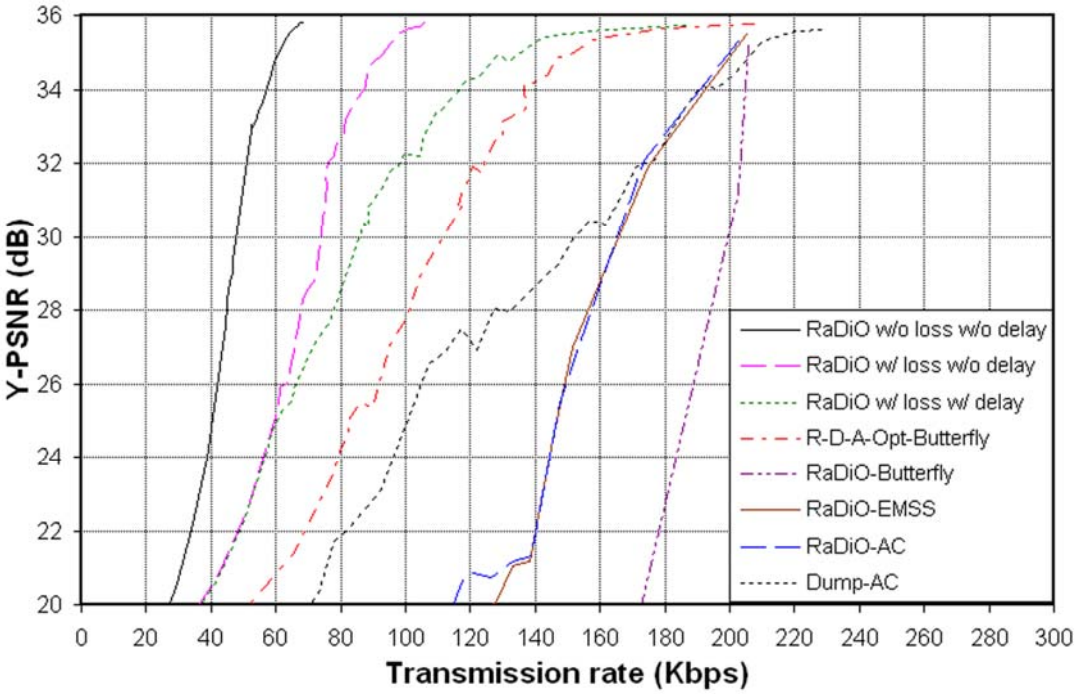

Figure 5-11 – R-D curves for various system (packet loss rate = 0.2), Container QCIF

An interesting observation is the performance difference between *Dumb-AC* and authentication-unaware *RaDiO* techniques for the different sequences *Container*

and *Foreman*. For the *Foreman* video, the packet distortion increments vary drastically among packets and, therefore, the authentication-unaware *RaDiO* techniques exploit this to provide better performance than *Dumb-AC*. In the *Container* video, the ship is moving slowly at constant velocity and hence the packet distortion increments have small variance, i.e. most packets have approximately similar distortion increments. Therefore, the authentication-unaware *RaDiO* and *Dumb-AC* techniques have similar performances, and in some cases the *Dumb-AC* has slightly better performance, which is perhaps due to the randomness of the pattern of delivered packets.

As shown in Figure 5-4, Figure 5-5, Figure 5-6, Figure 5-7, Figure 5-8, Figure 5-9, Figure 5-10 and Figure 5-11, *R-D-A-Opt-Butterfly* outperforms all systems, because it computes the transmission policy based on both packet distortion increments and authentication importance. At low bandwidth, the authentication-unaware *RaDiO* does not work anymore as its R-D curves drop quickly to unacceptable levels. Nevertheless, at the same low bandwidth the proposed R-D-A Optimization technique is still a workable solution whose R-D curves drop gracefully in parallel with the upper bound, *RaDiO* for unauthenticated video. However, we still notice that there is a performance gap between *RaDiO* and *R-D-A-Opt-Butterfly*. The possible reasons could be: 1) *R-D-A-Opt-Butterfly* has extra data rate due to the overhead size (the overhead constitutes 8Kbps data rate on top of the data rate of the coded video); 2) the packet authentication importance is not fully aligned with the distortion increments. If we could design an authentication graph such that packet authentication importance and distortion increments are fully aligned, the horizontal gap between *RaDiO* and *RaDiO_Butterfly_Aware* should be reduced to the data rate of authentication overhead (i.e. 8Kbps). However, this task constitutes future work, as

there are many constraints on the graph topology (due to factors like frame prediction, playout sequence, etc) and we cannot arbitrarily rearrange the packets in the authentication graph.

As a further observation to understand the plots, from the sender's point of view, the channel capacity is $(1-\varepsilon)^2 R_C$, where R_C is the channel bandwidth, because the sender considers a packet as successfully delivered only after the packet is acknowledged by the receiver. Therefore, to transmit all packets at source rate R_S, the required bandwidth is $R_S/(1-\varepsilon)^2$. More sophisticated acknowledgement schemes can reduce this required bandwidth to close to $R_S/(1-\varepsilon)$(depending on the playout delay), however, we keep the current approach for conceptual simplicity. When channel bandwidth drops below $R_S/(1-\varepsilon)^2$, the Y-PSNR of authenticated video starts to drop, which is validated by all R-D curves provided. For example, in Figure 5-4, the source rate is 158Kbps including 150Kbps for video data and 8Kbps for authentication overhead, so the knee of the R-D curve of *R-D-A-Opt-Butterfly* is located at $158/(1-0.03)^2$=168Kbps when the loss rate is 0.03. Similarly, when the loss rate is 0.05, 0.1 and 0.2, the knee of the R-D curve is 175Kbps in Figure 5-6, 195Kbps in Figure 5-8 and 246Kbps in Figure 5-10, respectively. Similar observations exist for the *Container* sequence in Figure 5-5, Figure 5-7, Figure 5-9 and Figure 5-11.

5.4.2.1 Low-complexity R-D-A Optimization Algorithm

We also implement the low-complexity streaming of authentication video, as described in Section 5.1.1. Figure 5-12 and Figure 5-13 compares the performance of optimized streaming (*R-D-A-Opt-Butterfly*) and low-complexity (*R-D-A-Opt-Butterfly-LC*) streaming, both using butterfly authentication. Note that we omit the R-

D curve at loss rate 0.05 due to space limitation, however, it has similar trends with other loss rates. At lower bandwidth, the low-complexity algorithm has an R-D curve close to the optimized algorithm, because it is able to identify those packets with substantially higher importance. However, the performance gap increases with bandwidth, because the low-complexity algorithm has limited capability to differentiate among the low-importance packets which do not vary much in their associated importance. In addition, the performance gap also increases with the packet loss rates. At lower loss rate, there is little uncertainty of packet delivery and hence the low-complexity algorithm achieves R-D performance close to the optimized algorithm. At higher loss rates, there is higher uncertainty and the low-complexity algorithm is, therefore, unable to handle the situation, leading to poorer performance.

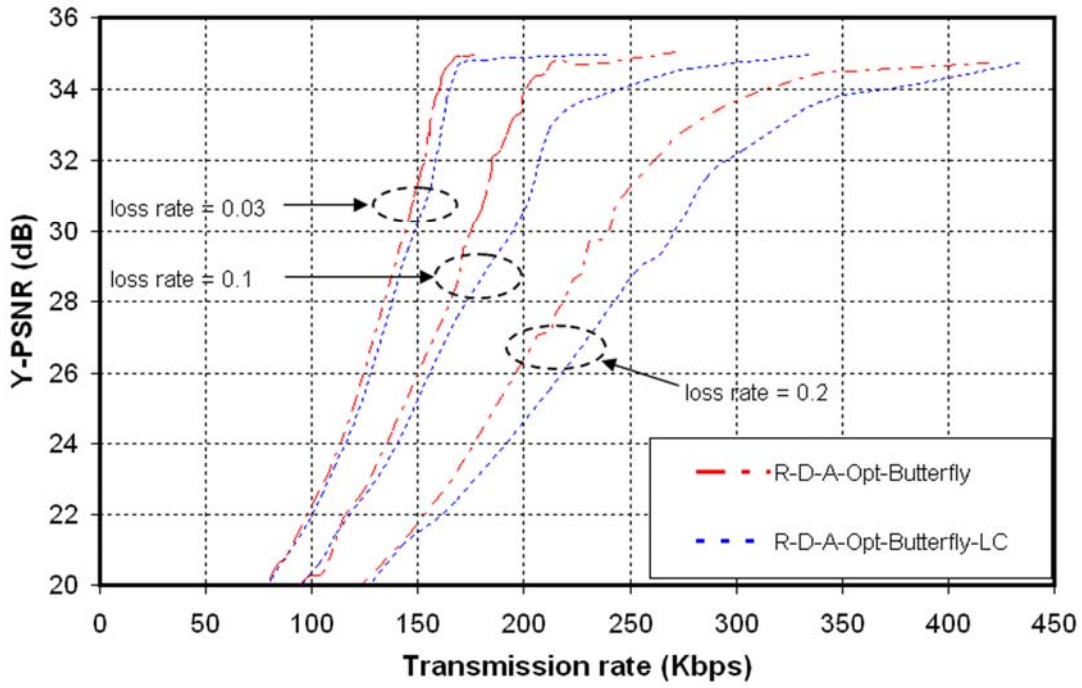

Figure 5-12 – R-D curves of R-D-A-Opt-Butterfly and R-D-A-Opt-Butterfly-LC (Packet loss rate = 0.03, 0.1 and 0.2), Foreman QCIF

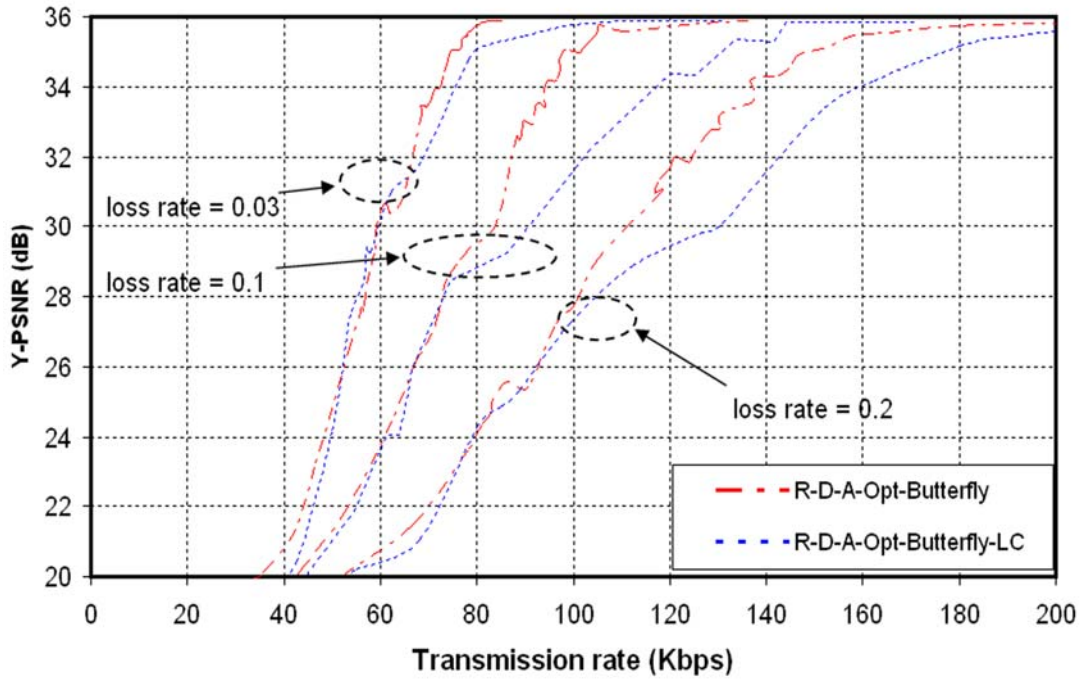

Figure 5-13 – R-D curves of R-D-A-Opt-Butterfly and R-D-A-Opt-Butterfly-LC (Packet loss rate = 0.03, 0.1 and 0.2), Container QCIF

5.4.3 R-D-A Optimization with Multiple Deadlines

To evaluate the performance of the proposed multiple-deadline R-D-A optimization algorithms, we compare three methods: 1) single-deadline (SD); 2) multiple-deadline with window-split; 3) multiple-deadline with extended-window. The experiment settings are exactly the same as those described in Section 5.4.1, with the only exception that playout delay is reduced to 400ms to show the effect of considering multiple deadlines.

Figure 5-14 and Figure 5-15 compares the PSNR performances of the three methods using test video sequences *Foreman* and *Container*, respectively. The two multiple-deadline methods outperform the single-deadline method by up to 4dB. The reason can be found from the statistics in

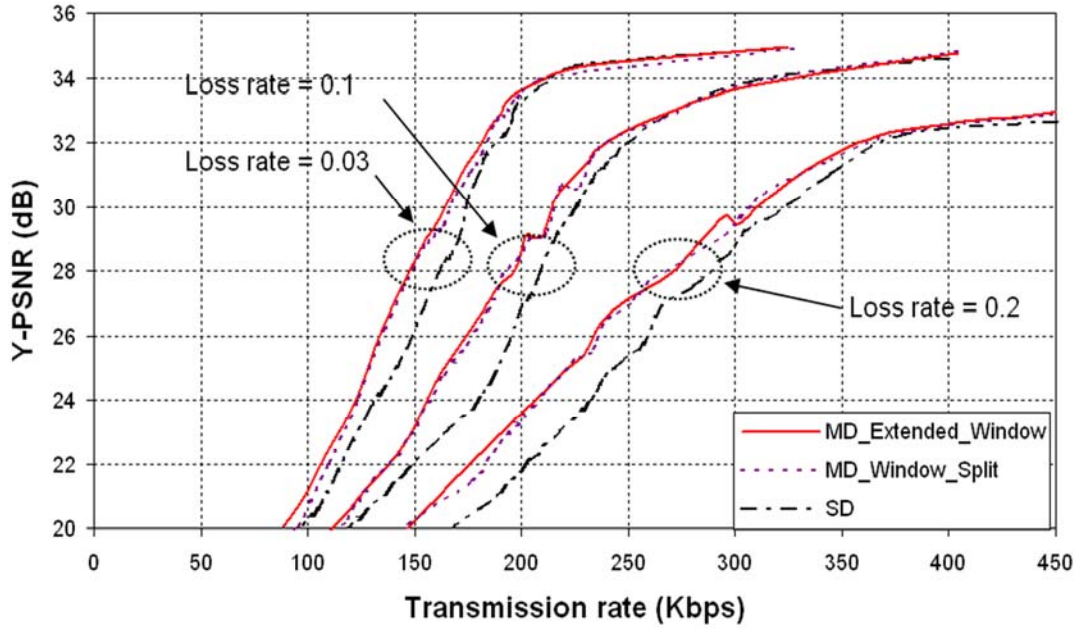

Figure 5-14 – R-D curves of *SD*, *MD_Extended_Window* and *MD_Window_Split*, Foreman QCIF

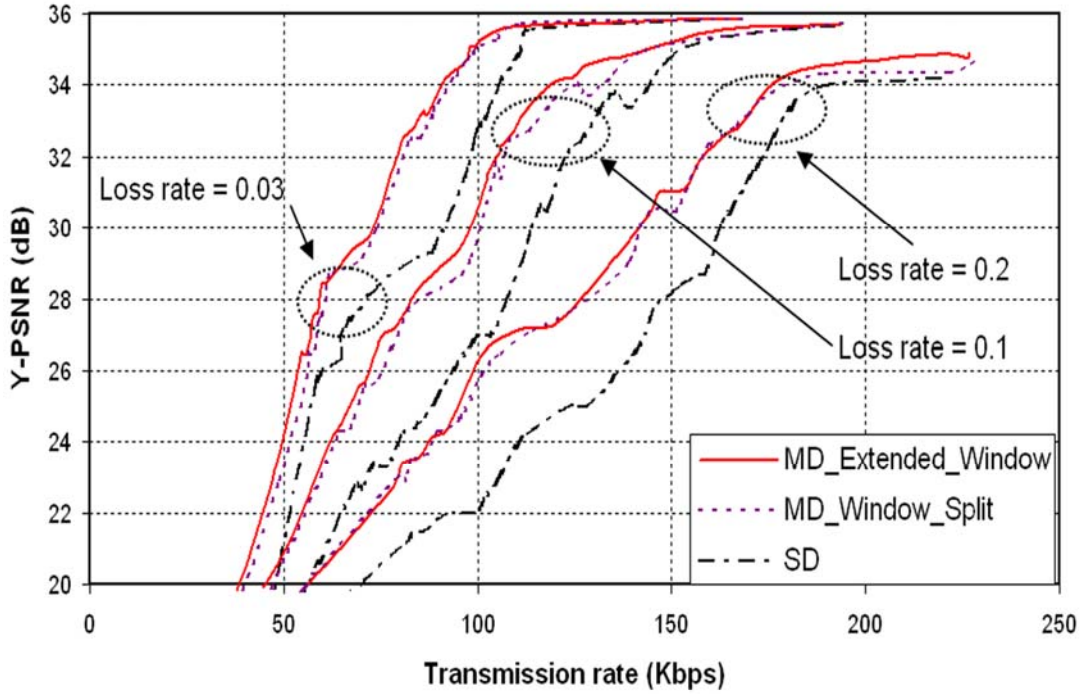

Figure 5-15 – R-D curves of *SD*, *MD_Extended_Window* and *MD_Window_Split*, Container QCIF

Table 5-1 – Statistics of packet transmission, delivery and verification (Forman, packet loss rate = 0.1)

Algorithm Tested	BW (Kbps)	No. of tx / pkt	Before T_0 (%) After T_0 (%) In [T_0-k, T_0] (%)	No. of rx / pkt	Before T_0 (%) After T_0 (%)	Rx prob. before T_0 (%)	Veri. Prob. (%)	% of pkt sent after T_0
Single Deadline R-D-A	199	1.278	100 0 0.37	1.015	98.53 1.47	79.34	98.5	0
Multiple Deadline _Window_Split	198	1.113	99.1 0.9 1.34	0.9997	96.33 3.67	80.4	99.3	4.5
Multiple Deadline _Extended_Window	196	1.11	99.01 0.99 1.28	0.9967	96.36 3.64	80.36	99.5	4.9

Table 5-1 The window-split method and extended window method have higher verification probability than the single-deadline method, due to re-transmission of expired but still important packets. Note that while the single-deadline method has a higher fraction of its received packets delivered before the playout deadline, its receiving probability before playout deadline is lower than the other two. This results from an unbalanced bandwidth distribution, i.e. certain packets are given too much bandwidth, while other packets are starved. This occurs because some packets never get a chance to be transmitted, as the packet(s) they depend on for verification is either lost or not transmitted, and then the single deadline R-D-A algorithm realizes that the packet in question should not be transmitted, given the single-deadline constraint. The multiple-deadline R-D-A algorithm provides valuable transmission flexibility which overcomes the above inefficiencies. It is also interesting to note that the two multiple deadline algorithms transmit a sizable percentage of the packets after their playout deadline (4.5% and 4.9%).

5.5 CONCLUSIONS

The main contributions of this chapter are summarized as follows:

- We propose an R-D-A Optimized streaming technique that computes the transmission policy to minimize the expected distortion of the authenticated video at the receiver. This is achieved by accounting for the *authentication importance* and *overhead size*, in addition to the original *distortion increment* and *packet size* used in conventional authentication-unaware *RaDiO*. In addition, we also show how to realize the proposed R-D-A Optimization using various authentication methods. Simulation results demonstrate that the R-D-A Optimization has the best R-D performance among all systems. Indeed, the authentication-unaware *RaDiO* systems do not work at low bandwidths, as the video quality drops quickly to unacceptable levels.
- Considering that R-D-A optimization has high complexity, we propose a low-complexity algorithm. Experimental results show that the low-complexity algorithm performs well at low bandwidth and low loss rates, compared with the optimized algorithm.
- We also show how to account for the multiple deadlines provided by the authentication graph, and evaluate the performance improvement of multiple-deadline versus single-deadline optimization for the proposed R-D-A Optimized streaming technique.

CHAPTER 6 - CONCLUSIONS AND FUTURE WORK

Media delivery and streaming over public networks are becoming practically more and more important, enabled by rapidly increasing network bandwidth, huge number of Internet users, advanced media compression standards and advanced network delivery technologies. It has a wide range of applications, including VoD, VoIP, IPTV, video conferencing, P2PTV and so on. However, security issues like authentication are serious concern for many users. Both the sender and the receiver in a streaming session would like to be assured that the received media is not modified by any unauthorized attacker and that malicious modification, if any, should be detected.

Traditional crypto-based authentication methods, like DSS, do not work well for streaming media due to the following reasons:

- Crypto-based DSS is not tolerant of network loss and even a single-bit difference may cause the received media not to pass the verification. However, the streaming media is usually encoded with error-resilient techniques and is tolerant to certain level of network loss, which is unavoidable when delivered over an unreliable channel like an UDP connection.
- Crypto-based DSS has high complexity and overhead, which imposes extra burden for the network as well as the sender and the receiver. This is exacerbated by the fact that the streaming media is huge in size and already takes a lot of bandwidth for transmission and computation resource for encoding and decoding.

- Crypto-based DSS does not support the paradigm of continuous verification as the streaming packets are being delivered. This is very important for a media streaming system, which is essentially a "play-as-being-received" technique.

In this thesis, we first propose a *Butterfly Authentication* method, which amortizes a digital signature among a group of packets, by connecting them as a butterfly graph. The Butterfly Authentication method has lower complexity, because it requires only one signature operation and around 2*N* hashing operations for a group of *N* packets. (Note that the complexity of a hashing operation is 3 orders of magnitude lower than that of a signature operation). It also has lower overhead, corresponding to 1 digital signature and 2*N* hash values for *N* packets. Note a hash (in the order of ten bytes) is much smaller than a digital signature (in the order of hundred bytes). In addition, the Butterfly Authentication method has very high verification probability because each packet in a butterfly graph is connected to two other packets that are independent of each other for verification, which maximize its verification probability. We experimentally show that the Butterfly Authentication method achieves near-optimal performance in terms of verification probability when overhead is fixed at around 2 hashes per packet.

Nevertheless, the Butterfly Authentication method has some limitations: 1) the total number of packets in a butterfly graph is not flexible; 2) the overhead is not flexible; 3) the signature packet grows with the total number of packets in the graph. To overcome the above 3 limitations, we also propose the *Generalized Butterfly Graph (GBG)* for authentication. The GBG graph supports arbitrary overhead and an arbitrary number of packets, and the signature packet does not have to grow with the total number of packets. The GBG framework includes a wide range of possible authentication graphs, and the problem of finding the best graph for a given situation

corresponds to a design problem. Given the total number of packets, packet loss rate and overhead budget, we need to find the optimized graph configurations like the number of rows/columns, edge placement and the number of transmission for signature packet. We also propose algorithms to solve the design problem based on the analysis and observation of GBG graph. Experimental results demonstrate that the GBG authentication method has significant performance improvement over existing graph-based methods.

We also proposed a Content-*aware Optimized Stream Authentication* method by recognizing the fact that packets have unequal importance in a media stream and that the average verification probability does not accurately reflect the quality of authenticated media. Therefore, the quality of authenticated media, besides the verification probability, should be used to measure the performance of the authentication methods. To make use of packets' unequal importance, we formulate a distortion-overhead optimization framework to compute an authentication graph that minimizes the expected distortion of the authenticated media, for given overhead and packet loss rate. This optimized performance is achieved by systematically allocating more authentication information (or overhead) to those more important packets, and vice versa. In other words, the authentication overhead is used in a more efficient manner. In addition, the proposed method aligns the coding dependency and authentication dependency for generic layered media data, which eliminate the situation where a packet is either not verifiable or not decodable. The system analysis and simulation results demonstrate that the proposed Content-Aware Optimized Stream Authentication method achieve a R-D curve of the authenticated media which is very close to the upper bound, i.e. the R-D curve when no authentication is

required. In addition, it substantially outperforms existing stream authentication methods in terms of media quality at all packet loss rates.

We also propose *Rate-Distortion-Authentication (R-D-A) Optimized* streaming method, which computes the optimized packet transmission policy that accounts for packet coding importance as well as authentication dependency. Here, the R-D-A optimization is defined as a rate-distortion optimization for authenticated media where the "rate" includes data rate for coded media data and authentication overhead, and the "distortion" is measured by the difference between the original media and the authenticated media. Considering that R-D-A optimization has high complexity, we also propose a low-complexity algorithm with performance close to the fully-blown algorithm. In addition, we also propose a R-D-A optimization algorithm that accounts for multiple deadlines associated with each packet. Compared with the straightforward concatenation of *RaDiO* and existing stream authentication methods, the proposed R-D-A Optimized streaming method has significantly better R-D performance. Indeed, it is the only method that works at low bandwidth.

6.1 FUTURE RELATED RESEARCH ISSUES

There are many more research issues to be solved in the field of authentication for streaming media over loss network. They include:

- Joint streaming and authentication: Chapter 4 describes the Content-aware Optimized Stream Authentication method, which is used to find the optimal topology policy for a given packet transmission policy (assuming each packet is transmitted once and packet loss rate is known). On the other hand, in Chapter 5, the Rate-Distortion-Authentication Optimized streaming method is used to find

the optimal transmission policy for a given topology policy (i.e. authentication graph is fixed). The future work could be joint streaming and authentication, where the topology policy and transmission policy are jointly determined for a given bandwidth constraints. At the higher level, we need to decide how much bandwidth should be allocated to authentication and how much bandwidth should be allocated to channel coding (like packet re-transmission or FEC parity data). At the lower level, we need to allocate the authentication overhead and channel coding redundancy to individual packets. To solve this problem, the most vital step is to formulate the problem and also find ways to solve the problem with low complexity. We believe better performance could be achieved by joint streaming and authentication.

- Authentication for real-time media streaming: Real-time media streaming like VoIP and video conferencing requires both low sender delay and low receiver delay, because the media packets have to be generated, transmitted and consumed in real-time. However, existing stream authentication methods have either long sender delay, or long receiver delay, or both. For instance, if the signature packet is the first packet to be transmitted as illustrated in Figure 2-3, the sender delay will be high; if the signature packet is the last packet to be transmitted as illustrated in Figure 2-4, the receiver delay will be high; if erasure-code is used for authentication as illustrated in Figure 2-2, both sender delay and receiver delay are high. Therefore, the future work should be focused on how to reduce the sender delay and receiver delay to meet the requirement of read-time streaming.
- Alignment of coding dependency and authentication dependency: The Content-aware Optimized Stream Authentication method in Chapter 4 aligns coding dependency and authentication dependency for generic layered media data with

simple dependency. However, there is much more complicated coding dependency. For example, Scalable Video Coding (SVC) video stream has dependency in three dimensions: temporal, spatial and quality. Even along the temporal dimension itself, the dependency graph could be as complicated as a hierarchical prediction structure. The future work should be focused on how to align the authentication dependency with those complicated coding dependency, which could help to significantly improve the quality of the authenticated media.

- Stream authentication adaptive to channel conditions: With the Content-aware Optimized Stream Authentication methods, the authentication graph is designed based on a given network condition. However, the network condition may change depending on a variety of conditions, such as network congestion, fading phenomenon in wireless networks, traffic shaping by network operators, and so on. The future work could focus on how to make the authentication graph adaptive to channel conditions.

BIBLIOGRAPHY

[1] http://www.internetworldstats.com/stats.htm

[2] Information technology – JPEG 2000 image coding system, ISO/IEC International Standard 15444-1:2000

[3] D. Taubman and M. W. Marcellin, JPEG2000: Image Compression Fundamentals, Standards and Practice, Kluwer Academic Publisher: Dordrecht, 2001, pp. 275-379

[4] Information technology – JPEG 2000 image coding system: Secure JPEG 2000, ISO/IEC International Standard, 15444-8: 2007

[5] Information technology – JPEG 2000 image coding system: Wireless JPEG 2000, ISO/IEC International Standard, 15444-11: 2006

[6] Information technology – Coding of audio-visual objects: Advanced video coding, ISO/IEC Final Draft International Standard, 14496-10:2003

[7] T. Wiegand, G. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 7, July 2003, pp. 560-576

[8] I. Richardson, H.264 and MPEG-4 Video Compression: Video Coding for Next-Generation Multimedia, Wiley, 2003

[9] H. Schwarz, T. Hinz, H. Kirchhoffer, D. Marpe, and T. Wiegand, "Technical description of the HHI proposal for SVC CE1," ISO/IEC JTC 1/SC29/WG11, doc. M11244, Palma de Mallorca, Spain, Oct.2004.

[10] ISO Media File Format Specification, ISO/IEC JTC1/SC29/WG11 MPEG01/N4270-1, 2001

[11] L. Kontothanassis, R. Sitaraman, J. Wein, D. Hong, R. Kleinberg, B. Mancuso, D. Shaw, and D. Stodolsky. "A transport layer for live streaming in a content delivery network", Proceedings of the IEEE, pp. 1408-1419, 2004

[12] J. Li, "PeerStreaming: A practical receiver-driven Peer-to-Peer media streaming system," Microsoft Technical Report, MSR-TR-2004-101, Sept. 2004.

[13] E. Setton and B. Girod, “Rate-Distortion Analysis and Streaming of SP and SI Frames,” IEEE Transactions on Circuits and Systems for Video Technology, Vol.16, No. 6, pp.733–743, 2006.

[14] A. Ali, A. Mathur and H. Zhang, “Measurement of Commercial Peer-to-Peer Live Video Streaming,” Workshop in Recent Advances in Peer-to-Peer Streaming, August, 2006.

[15] S. Wee and J. Apostolopoulos, “Secure scalable streaming and secure transcoding with JPEG-2000,” in Proc. IEEE International Conference on Image Processing (ICIP), 2003.

[16] S. Wee and J. Apostolopoulos, “Secure transcoding with JPSEC confidentiality and authentication,” in Proc. IEEE International Conference on Image Processing (ICIP), 2004.

[17] S. Imaizumi, O. Watanabe, M. Fujiyoshi and H. Kiya, “Generalized hierarchical encryption of JPEG 2000 codestreams for access control”, in Proc. IEEE International Conference on Image Processing (ICIP), 2005.

[18] B. Schneier, Applied Cryptography, Wiley, 1996

[19] B. B. Zhu, C. Yuan, Y. Wang, S. Li, “Scalable protection for MPEG-4 fine granularity scalability”, IEEE Transactions on Multimedia Vol. 7, No. 2, pp. 222-233, 2005.

[20] Q. Sun and S.-F. Chang, “Signautre-based media authentication,” in Multimedia Security Handbook, Chapter 21, pp. 619-662, Edited by Furht and Kirovski, CRC Press, 2005

[21] C.-Y. Lin and S.-F. Chang, “A robust image authentication method distinguishing JPEG compression from malicious manipulation,” IEEE Transactions on Circuits and Systems for Video Technology, Vol 11, No. 2, pp.153-168, Feb. 2001

[22] Q. Sun and S.-F. Chang, “A secure and robust digital signature scheme for JPEG2000 image authentication”, IEEE Transactions on Multimedia, 7(3), pp. 480-494, June 2005.

[23] Q. Sun, D. He and Q. Tian, “A secure and robust authentication scheme for video transcoding,” IEEE Transactions on Circuits and Systems for Video Technology, Vol.16, No.10, 2006.

[24] Q. Sun, S. Ye, C-Y. Lin and S-F. Chang, “A crypto signature scheme for image authentication over wireless channel,” International Journal of Image and Graphics, Vol. 5, No.1, pp 1-14, 2005.

[25] S. Wenger, “H.264/AVC over IP”, IEEE Transaction on Circuits and Systems for Video Technology, Vol. 13, No. 7, pp. 645-656, JULY 2006

[26] J. M. Park, K. P. Chong and H. J. Siegel, “Efficient multicast packet authentication using signature amortization,” in Proc. Of IEEE Symposium

on Research in Security and Privacy, Oakland, CA, May 2002, IEEE Computer Society, 490-495

[27] J. M. Park, K. P. Chong and H. J. Siegel, "Efficient multicast stream authentication using erasure codes," ACM Transactions on Information and System Security, Vol. 6, No. 2, May 2003, Pages 258-285

[28] A. Pannetrat and R. Molva, "Efficient multicast packet authentication," in Proceeding of the Network and Distributed System Security Symposium, NDSS 2003, San Diego, California, USA

[29] C. K. Wong and S. Lam, "Digital Signatures for Flows and Multicasts", The University of Texas at Austin, Department of Computer Sciences, Technical Report TR-98-15. July 1998

[30] R. Merkle, "A certified digital signature," in Proc. Of the conference on advances in Cryptology (CRYPTO'89), Santa Barbara, CA, August 1989, S. Goldwasser, Ed. Springer-Verlag, New York, NY, 218-238

[31] R. Gennaro and P. Rohatgi. "How to sign digital streams," in Advances in Cryptology - CRYPTO '97, pp. 180–197.

[32] P. Rohatgi, "A compact and fast hybrid signature scheme for multicast packet authentication," in Proc. Of the 6th ACM conference on computer and communications security (CCS), Singapore Nov. 1999, ACM Press, New York, NY, 93-100

[33] A. Perrig, R. Canetti, J. Tygar and D. Song. "Efficient authentication and signing of multicast streams over lossy channels," in Proc. of IEEE Symposium on Security and Privacy, 2000, pp. 56-73.

[34] P. Golle and N. Modadugu. "Authenticating streamed data in the presence of random packet loss," ISOC Network and Distributed System Security Symposium, 2001, pp 13--22.

[35] S. Miner and J. Staddon, "Graph-based authentication of digital streams," in Proc. Of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, pp.232-246, May, 2001

[36] S. Lin and D. J. Costello, "Error control coding: fundamentals and applications," Prentice-Hall, 1983

[37] M. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," in Journal of ACM, Vol. 36, Issue 2, pp. 335-348, 1989

[38] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," IEEE Transactions on Multimedia, Vol. 8, No. 2, pp. 290-404, April 2006

[39] J. Chakareski, J. Apostolopoulos, S. Wee, W.-T. Tan, B. Girod, "Rate-Distortion Hint Tracks for Adaptive Video Streaming", IEEE Transactions on

Circuits and Systems for Video Technology, special issue on "Video Adaptation", October 2005.

[40] J. Chakareski, J. Apostolopoulos, S. Wee, W.-T. Tan, B. Girod, "R-D Hint Tracks for Low-Complexity R-D Optimized Video Streaming," IEEE ICME, June 2004.

[41] J. Chakareski and B. Girod, "Rate-distortion optimized packet scheduling and routing for media streaming with path diversity," in Proceeding of Data Compression Conference (DCC), 2003

[42] J. Chakareski, S. Han, and B. Girod, "Layered coding vs. multiple descriptions for video streaming over multiple paths," in Proc. 11th ACM International conference on multimedia, Berkeley, CA, Nov. 2003, pp. 422-431

[43] J. Chakareski and G. Girod, "Server diversity in rate-distortion optimized streaming of multimedia," in Porc. ICIP, Barcelona, Spain, Sep. 2003

[44] A. C. Begen, Y. Altunbasak, and M. A. Begen, "Rate-distortion optimized on-demand media streaming with server diversity," in Proc. ICIP, Barcelona, Spain, Sep. 2003

[45] J. Chakareski, P. A. Chou, and B. Girod, "Rate-distortion optimized streaming from the edge of the network," in Proc. Workshop on Multimedia Signal Processing, S. Thomas, Ed., Dec. 2003, pp. 29-52

[46] J. Chakareski, P. A. Chou, and B. Girod, "RaDiO Edge: Rate-distortion optimized proxy-driven streaming from the network edge," in IEEE/ACM Trans. Networking, Vol. 14, No. 6, December 2006

[47] M. kalman, P. Ramanathan, and B. Girod, "Rate-distortion optimized video streaming with multiple deadlines," in Proc. International conference on image processing, Barcelona, Spain, Sep. 2003

[48] J. Chakareski and B. Girod, "Rate-distortion optimized video streaming with rich acknowledgements," in Proc. SPIE Visual Communication and image processing, San Jose, CA, jan. 2004

[49] R. Zhang, S. L. Regunathan and K. Rose, "End-to-end distortion estimation for RD-based robust delivery of pre-compressed video," in conf. Rec. 35th Asilomar Conf. Signals, Systems and Computers, Asilomar, CA, Nov. 2001, pp. 210-214

[50] R. Zhang, S. L. Regunathan and K. Rose, "optimized video streaming over lossy networks with real-time estimation of end-to-end distortion," in Proc. IEEE ICME, Vol 1, Lausanne, Switzerland, Aug. 2002, pp. 861-864

[51] C. Papadopoulos and G. M. Parulkar, "Retransmission-Based Error Control for Continuous Media Applications," In Proc. 6th Intl. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), April 1996, pp. 5-12

[52] M. T. Lucas, B. J. Dempseyt and A. C. Weaver, "MESH: Distributed Error Recovery for Multimedia Streams in Wide-Area Multicast Networks," In Proc. IEEE International conference on Communication, June 1997, Vol. 2, pp. 1127-1132

[53] H. Radha, Y. Chen, K. Parthasarathy and R.Cohen, "Scalable Internet video using MPEG-4," Signal Processing: Image Communication 15 pp. 95-126, 1999

[54] Y. Desmedt, Y. Frankel, M. Yung, "Multi-receiver/multi-sender network security: efficient authenticated multicast feedback," IEEE INFOCOM'92, 1992 pp. 2045-2054

[55] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, B. Pinkas, "Multicast security: a taxonomy and efficient constructions," IEEE INFOCOM 1999

[56] D. Boneh, G. Durfee, M. Franklin, "Lower bounds for multicast message authentication," Eurocrypt'01, LNCS, vol. 2045 pp. 437-452, 2001

[57] F. Hiroshi, K. Wattanawong, K. Kaoru, "Combinatorial bounds and design of broadcast authentication," IEICE Trans. E79-A (4), pp. 502-506, 1996

[58] K. Kurosawa, S. Obana, "Characterization of (k,n) multi-receiver authentication," Information security and privacy, ACISP'97, LNCS, vol. 1270 pp. 204-215, 1997

[59] S. obana, K. Kurosawa, "Bounds and combinatorial structure of (k,n) multi-receiver A-codes," Designs, Codes and Cryptography 22 (1) pp. 47-63, 2001

[60] R. Safavi-Naini, H. Wang, "New results on multi-receiver authentication codes," Advances in Cryptology: ERUOCRYPT'98, LNCS, vol. 1403, pp. 527-541, 1998

[61] R. Safavi-Naini, H. Wang, "Multireceiver authentication codes: models, bounds, constructions and extensions," Information and Computation 151, pp. 148-172, 1999

[62] F. Bergadano, D. Cavagnino, B. Crispo, "Individual single source authentication on the MBone," IEEE International conference on Multimedia and Expo, 2000

[63] F. Bergadano, D. Cavagnino, B. Crispo, "Individual authentication in multiparty communications," Computers and Security 21 (8), pp. 719-735, 2002

[64] M. Mitzenmacher, A. Perrig, Bounds and improvement for BiBa signature schemes," Technical Report (TR-02-02), Harvard University, 2002

[65] A. Perrig, The BiBa one-time signature and broadcast authentication protocol, The Eighth ACM Conference on Computer and Communications Security, November, 2001.

[66] A. Perrig, R. Canetti, D. Song, J.D. Tygar, Efficient and secure source authentication for multicast, Eighth Annual Internet Society Symposium on Network and Distributed System Security, 2001.

[67] A. Perrig, R. Canetti, J.D. Tygar, D. Song, The TESLA broadcast authentication protocol, RSA CryptoBytes 5, 2002

[68] L. Reyzin, N. Reyzin, Better than BiBa: short one-time signatures with fast signing and verifying, Seventh Australian Conference on Information Security and Privacy, LNCS, vol. 2384, 2002 pp. 144–153.

[69] http://www.cryptopp.com/benchmarks-p4.html

[70] J. C. de Martin, "Source-driven packet marking for speech transmission over differentiated-services networks," in Proc. ICASSP, Salt Lake City, UT, May 2001.

[71] J. C. de Martin and D. Quaglia, "Distortion-based packet marking for MPEG video transmission over DiffServ networks," in Proc. ICME, Tokyo, Japan, Aug. 2001.

[72] F. De Vito, L. Farinetti, and J. C. De Martin, "Perceptual classification of MPEG video for differentiated-services communications," in Proc. ICME, vol. 1. Lausanne, Switzerland, Aug. 2002, pp. 141–144.

[73] J.-C. Bolot and A. Vega-Garcia. The case for FEC-based error control for packet audio in the Internet. [Online]. Available: http://wwwsop.inria.fr/rodeo/personnel/bolot/papers.html

[74] M. Podolsky, C. Romer, and S. McCanne, "Simulation of FEC-based error control for packet audio on the internet," in Proc. IEEE Infocom, San Francsico, CA, Mar. 1998.

[75] J. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-based error control for interactive audio on the Internet," in Proc. IEEE Infocom, New York, Mar. 1999.

[76] M. Podolsky, S. McCanne, and M. Vetterli, "Soft ARQ for layered streaming media," Univ. California, Comput. Sci. Div., Berkeley, Tech. Rep. UCB/CSD-98-1024, Nov. 1998.

[77] M. Podolsky, S. McCanne, and M. Vetterli, "Soft ARQ for layered streaming media," J. VLSI Signal Process. Syst. Signal, Image Video Technol., Special Issue on Multimedia Signal Processing, vol. 27, no. 1–2, pp. 81–97, Feb. 2001.

[78] P.-C. Hu, Z.-L. Zhang, and M. Kaveh, "Channel condition ARQ rate control for real-time wireless video under buffer constraints," in Proc. ICIP, vol. 2, Vancouver, BC, Canada, Oct. 2000, pp. 124–127.

[79] P. A. Chou, A. E. Mohr, A. Wang, and S. Mehrotra, "FEC and pseudo-ARQ for receiver-driven layered multicast of audio and video," in Proc. Data Compression Conf.. Snowbird, UT, Mar. 2000, pp. 440–449.

[80] Y. J. Liang, J. G. Apostolopoulos and B. Girod, "Analysis of packet loss for compressed video: does burst-length matter?," in Proc. IEEE International conference on Acoustic, Speech and Signal Processing (ICASSP), Hong Kong, April 2003, pp. 684-687

[81] RFC 2475, "An Architecture for Differentiated Services,", IETF, 1998

[82] www.realnetwork.com

[83] www.microsoft.com/windows/windowsmedia

[84] R. Fletcher, "Practical method of optimization," Wiley, 2nd edition, 1987

[85] J. Mogul and S. Deering, "Path MTU discovery," RFC 1191, November, 1990

[86] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: a transport protocol for real-time applications," RFC 3550, July, 2003

[87] Y.-K. Wang, S. Wenger, and M. M. Hannuksela, "Common conditions for SVC error resilience testing," ISO/IEC JTC 1/SC 29/WG11 and ITU-T SG16 Q.6 JVT-P206d0, 2005

[88] V. Paxson, "End-to-End Internet Packet Dynamics,", in IEEE/ACM Transactions on Networking (TON), Vol. 7, No. 3, 1999, pp. 277-292

[89] The Network Simulator (NS-2). [Online]. Available: http://www.isi.edu/nsnam/ns/

[90] H.264/AVC Reference Software, JM Version 10.2. [Online] Available: http://iphome.hhi.de/suehring/tml

[91] S. Wenger, M. M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, "RTP payload format for H.264 Video," RFC 2984, 2005