11-2003

Content-Aware Authentication of Motion JPEG2000 Stream in Lossy Networks

Yongdong WU
Institute for Infocomm Research

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

DOI: https://doi.org/10.1109/TCE.2003.1261157

Citation

WU, Yongdong and DENG, Robert H.. Content-Aware Authentication of Motion JPEG2000 Stream in Lossy Networks. (2003). *IEEE Transactions on Consumer Electronics*. 49, (4), 792-801. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/141

Published in IEEE Transactions on Consumer Electronics, Volume: 49, Issue: 4 , Nov. 2003, Pages 792-801.
https://doi.org/ 10.1109/TCE.2003.1261157

Content-aware Authentication of Motion JPEG2000 Stream in Lossy Networks

Yongdong Wu, Robert H. Deng

Abstract**—***Stream authentication schemes (SAS) aim to achieve effective authentication of multicast streams over lossy networks. Almost all of the existing SASs are designed for stream data integrity protection only. In this paper, we argue that content integrity protection is more important than data integrity to human users. We present a content-aware SAS in Motion JPEG2000 streaming. In our scheme, a chunk of JPEG2000 codestreams is encapsulated into a block of packets using Multiple Description Coding (MDC). Our MDC exploits the inherent structure of Motion JPEG2000 codectreams and is used to preserve the semanteme/content of the stream over lossy networks. To achieve robust authentication, we encode digital signature and other integrity tokens with Forward Erasure Code against packet losses. The experiment result demonstrates that our proposed scheme allows for effective content authentication of Motion JPEF2000 streaming.*

Index Terms—Digital Signature, Stream Authentication Scheme (SAS), Forward Erasure Code (FEC), Multiple Description Code (MDC), Motion JPEG2000.

I. Introduction

MEDIA streaming is becoming more and more popular due to the explosive growth of Internet and multimedia technologies and applications. Content Delivery Networks (CDNs) have been used to provide low latency, scalability, fault tolerance and load balancing for the delivery of web content and more recently streaming media. Figure 1 shows a generic architecture of CDNs, where many surrogates (e.g. more than 10,000 edge servers in akamai.com) are located at the edge of the Internet to reduce client access time. Those servers communicate with each other over a dedicated broadband network. The clients may receive the stream using LAN, Wireless LAN, ISDN, PSTN, and Cable Network.

In critical application fields such as government, finance, healthcare and law, clients normally demand authenticity of the received content. Accordingly, streaming data integrity is of importance in these applications, thereby a security mechanism for authenticating stream is required. A straightforward stream authentication scheme (SAS) is to append a signature or Message Authentic Code (MAC) to each packet. This naïve solution increases the communication overhead and degrades the performance of the sending host. For example, it costs 20ms in Pentium III 800MHz to generate a 1024-bit RSA digital signature. Apparently, this one signature per packet approach is not viable for over 15fps video conferences (e.g. H.261 [1]). Watermark based authentication schemes may not help either to authenticate video streams because an attacker can make small modification on the stream without affecting the watermark validation. Therefore, besides necessary security function, a practical SAS must be lightweight in overhead and computational cost, and tolerant of packet losses.

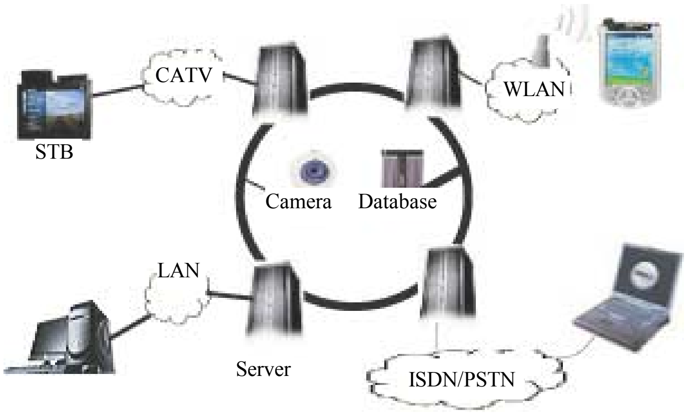


Fig.1. Generic architecture of a CDN

To improve efficiency, existing SASs divide a stream into blocks of multiple packets, and generate one signature for the block instead of for each packet. However, this paradigm suffers from packet losses. Conventional approaches for dealing with packet loss for static data, such as packet retransmissions, may not be suitable for streaming and multicast applications. Therefore, a SAS must be robust against packet loss.

This paper proposes an authentication scheme for Motion JPEG2000 streaming in lossy network. To our knowledge, there are few publications on Motion JPEG2000 streaming and none on secure Motion JPEG2000 streaming. Our objective here is to design a SAS which is content aware (to be made clear in Section II) and robust over lossy networks. According to the specification [2], a Motion JPEG2000 file consists of chunks of image codestreams. In our scheme, the image codestreams are first transcoded into media fragments; the media fragments are protected with integrity tokens and digital signature. Then the integrity tokens and signature are encoded into codewords using FEC for data loss resilience; each media fragment together the part of FEC codewords are encapsulated into a packet. This maps a chunk of codestreams

The authors are with Institute for Infocomm Research, 21, Heng Mui Keng Terrace, Singapore, 119613, Email:{wydong, deng}@i2r.a-star.edu.sg}

into a block of packets. As a result, a Motion JPEG2000 file is transformed into a sequence of blocks with each block consisting of multiple packets. When a fraction of packets in a block is received at the receiving end, the FEC is used to recover digital signature and integrity tokens which are then verified for content authentication. Due to the use of multiple description coding and sequence numbers, our scheme not only provides content integrity with much higher probability than previous SASs [6] - [17], but also defeats collage attack which the previous SASs suffer from.

The rest of this paper is organized as follows. In Section II, we introduce some basic concepts in Motion JPEG2000, FEC and MDC. We also review existing SASs and discuss their weaknesses. Section III elaborates the content-aware SAS. Section IV analyzes the performance of the proposed SAS in terms of computational cost and authentication probability. Section V describes our experiment results. Section VI draws a conclusion.

II. Related Work

In the media streaming applications, the receivers may receive only a fraction of packets multicast from the sender in a lossy network. SAS targets for authenticating the received packets with as few payload and computational cost as possible. The present SAS provides an error-resilient content integrity protection, in particular to authentication of Motion JPEG2000 stream. It integrates the error-resilience approaches with cost amortization method together. For completeness, the following subsections describe the concepts of motion JPEG2000, error resilience technologies and previous SASs.

A. JPEG2000

JPEG2000 [3]-[5] is a latest international standard for still image compression. It allows an image to be divided into rectangular non-overlapping regions known as tiles, which are compressed independently, as though they were entirely distinct images. For each tile, a multiple-level Discrete Wavelet Transform (DWT) is performed. A r^{th}-level DWT transforms subband LL_{r-1} into 4 subbands LL_r, LH_r, HL_r and HH_r, where LL_0 represents the original image. Each subband is partitioned into code-blcoks which are the code units. In order to provide locality for accessing certain portions of an image, contiguous code-blocks forms a precinct. The bitstreams generated from the code-blocks are organized into a JPEG2000 codestream. JPEG2000 was originally designed with the concepts of tiling and scalability in mind so as to stream a JPEG2000 image efficiently. For example, Deshpande et al [18] proposed a scalable streaming of JPEG2000 image of large size according to HTTP protocol so that the client can obtain the selected resolution or layer image.

B. Motion JPEG2000

Presently, motion JPEG2000 specification [2] proposed streaming applications for high-quality frame-based images with no inter-frame coding. The applications may include PC-based video capturing, high quality digital video recording for professional broadcasting and motion picture production from film-based to digital systems, and high-resolution medical and satellite imaging. However, the specification merely defines the file format but does not address any mechanism for streaming. In the specification, a motion JPEG2000 codestream is logically divided into tracks representing timed sequences of media. Video tracks contain visual frames, audio tracks contain audio media, while hint tracks contain instructions for a streaming server how to form packets for a streaming protocol. For the sake of simplicity, the present SAS considers the video track only and ignores other tracks.

C. Error Resilience

Packet losses [19]-[22] are common on the Internet due to the following factors:

- *Routing policy*: Network routers employ First-In-First-Out policy in which, the successively arrived packets are dropped if the router buffer is full.
- *Bandwidth limitation*: When a streaming is transmitted from a broadband network to a narrow-band network, for example, from LAN to PSTN, the Internet Service Provider has to discard a portion of arrived IP packets so as to customize the bandwidth.
- *Device limitation*: The devices such as PDA, have no power enough to deal with all the arriving packets, and have to ignore a portion of them.
- *Delay constrain*: On account of the real-time nature of video streaming session, packets arrived late are useless and discarded.
- *Packet error*: By checking the packet checksum, the receiver or router may detect tampered packets and discard them. From the viewpoint of users, this kind of packets is lost.

Retransmission-based error resilience is infeasible for Internet streaming because retransmission of lost data takes delay. Additionally, in multicast channel, the server has no way to retransmit the lost packets because different receivers may receive different subset of packets. Luckily, redundancy coding allows for more effective error correction, and most of the research on this topic falls into two categories: FEC [23] - [30] and MDC [31]-[32].

1) Forward Error Correction

In the encoding process, media data is divided into fundamental segments and enhanced segments based on their priorities. Fundamental segments are given a higher level of protection because they are most important. Conversely, the enhanced segments are used to improve the quality of the media and assigned a lower level of protection. That is to say, a receiver can recover fundamental segments with fewer number of received packets than enhanced segments.

2) Multiple Description Code

A video (or other media) can be represented with a number of independent and complementary descriptions. Each description is typically of roughly equal importance. If either description is received, basic quality video is available. The more descriptions are used, the better video quality is. In other words, each description can be decoded independently to give a usable video, and the video quality improves progressively with additional descriptions received. Generally, MDC will have lower compression rate than FEC.

D. Stream Authentication Schemes

Unlike traditional authentication of files, stream authentication aims for authenticating the IP packets when only a fraction of the sent packets arrives at the receiver end. Its applications include stored video streaming (e.g. movie) and real-time streaming (e.g. image monitoring and broadcasting).

Gennaro and Rohatgi [6] initially addressed stream authentication and suggested two solutions to authenticate a stream. In their off-line paradigm, the sender divides a finite stream into a sequence of packets P_i (i = 1,2,…,n) and generates a new packet sequence P'_i by appending a hash value of packet P'_{i+1} (initial value P'_{n+1} =0) to P_i. That is to say, $P'_i = P_i \parallel H(P'_{i+1})$ where $H(\cdot)$ is a one way function and $\parallel$ denotes concatenation. The first packet P'_0 includes $H(P'_1 \parallel n)$ and the signature on $H(P'_1 \parallel n)$ produced by the sender. After receiving the first packet P'_0, the receiver verifies the signature, and saves $H(P'_1 \parallel n)$ so as to authenticate the second packet P'_1. After authenticating P_1, the receiver extracts $H(P'_2)$ from packet P'_1 and keeps it to authenticate P_2. This process continues till P_n is authenticated. In their on-line solution, the size of the stream is unknown, each packet is appended with a one-time signature and a one-time public key for verifying the one-time signature of the next packet. The methods increase the traffic substantially and cannot tolerate packet loss.

To overcome the above shortcomings, a chaining technique for signing flows is presented in [7] based on tree chaining techniques. To construct an authentication tree, a block of packets is considered. The root is the digest of the block and the intermediate node value is the digest of its children nodes, and the leaf is hash value of a packet. Any packet can be authenticated individually because it carries its own authentication information that consists of the block signature, the packet position in the tree, and the siblings of each node in the packet's path to the root. Wong and Lam [7] extended Feige-Fiat-Shamir signature scheme to be adjustable and incremental so that verifiers with different resources can verify the stream at different security levels.

Perrig et al. [8] proposed two stream authentication schemes. One is Timed Efficient Stream Loss-tolerant Authentication (TESLA) relying on a synchronization margin between the sender and the receiver. The sender transmits to the receiver a packet P, a commitment to a key k and MAC $H(k \parallel P)$. After a certain time interval, the sender discloses key k so that the receiver can verify packet P. A prerequisite security condition of TESLA is that the receiver must obtain a packet before the next packet is sent from the sender. The other scheme is the Efficient Multi-chained Stream Signature (EMSS) scheme which aims to achieve non-repudiation besides source authentication. In the EMSS scheme, a packet P_i includes hash value $H(P_{i-1})$ of its previous P_{i-1}. The last packet is a signature on its previous packet (i.e., the last but next packet). If the signature is authenticated, all the previous packets are authenticated. To increase robustness, any packet P_i may be attached with several hash values of P_j where $j<i$.

Miner and Staddon [9] authenticated digital stream over a lossy network based on a graph. In their authentication scheme tolerant of random loss, each packet is assumed to be lost independently with the same probability. In a p-random graph, for all pairs of nodes (i,j) where $i<j$, there is a directed edge from node i to node j with probability p. A node is associated with a packet and an edge $\vec{e}(i,j)$ means $H(P_i) \in P_j$. The first packet P_1 is the signature packet that is always received by the receiver. Hence, packet P_i can be authenticated if and only if there is a path from P_i to P_1. The authors also constructed an authentication scheme that is tolerant of burst loss. In this scheme, the packets are categorized based on their priorities. The packets of the highest priority class are evenly spaced throughout the stream. To tolerate single burst, the packets of the highest priority class are chained and packets of other classes are chained to those packets of the highest priority class. The authors extended this scheme to tolerate multi-burst loss by adding the edges ending in the highest priority packets.

Park et al [10] constructed an authentication scheme SAIDA by encoding the hash values of packets and the signature on the whole block with an erasure code algorithm (e.g. Information Dispersal Algorithm). This strategy amortizes codewords for packet-data integrity and block signature among all the packets in a block to reduce space overhead and increase the tolerance of packet loss. Once obtaining sufficient number of packets, the receiver can recover the signature and authenticate the packets. This scheme has to balance the three factors: computation complexity, the recipient resource and authentication latency. To make the computation efficiency, the size of the block should be larger so as to amortize the cost of signature into a multiple of packets. However, the recipient has to wait for more packets so as to authenticate the content, and the buffer limitation of the recipient is another shortage. In error-prone network such as wireless, the payload may be more heavy.

Recently, Pannetrat et al [16] improved SAIDA by constructing a systematic code in place of Information Dispersal Algorithm. Therefore only the signature codeword and parity of integrity codeword are piggyback into the packets, the received packet-data can be re-used for computation of the integrity message so as to reduce the payload per packet.

In order to explain each received packet clearly, Wu et al [17] partitioned a stream object by object, so that each object is explained independently. This object-based solution alleviates the ambiguity of the stream.

However, the above SASs share two weaknesses which will be addressed in the following.

1) Content Ambiguity

Although the above error-resilience SASs can authenticate the received packets with a predefined probability, they provide data integrity protection other than content-integrity protection. In other words, the semantic meaning or content of the received data may be different from that of the original one if only a fraction of authentic packets arrives at the receiver side successfully. For example, a JPEG2000 image shown in Figure 2 consists of 4 tiles T_i (i=1,2,3,4) and each tile is encapsulated into one IP packet, denoted as P_1, P_2, P_3, and P_4. The sender delivers the packets to the receivers along with the signature and additional integrity tokens according to the scheme [11]. When the stream packets are transmitted over a lossy network, the receiver obtains packets P_1, P_2, and P_4 only, consequentially the reconstructed image is shown as Figure 3.

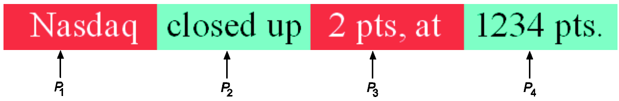


Fig.2. Original image. It includes four tiles. The codestream of each tile is encapsulated into one IP packet.

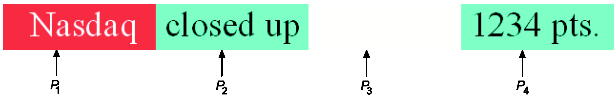
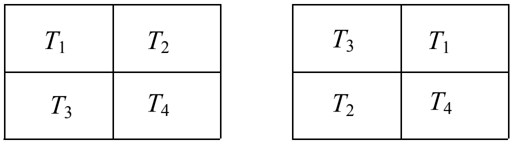


Fig.3. Reconstructed image. The third packet is lost in transit. Hence, one tile can not be shown. The semantic meaning of reconstructed image is far from that of the original one.

Is the reconstructed image trustworthy? Definitely not! In the above example, “authenticity” of the stream is worse than non-authenticity because the recipient may suspect the recovered image reconstructed with packets from a non-authenticated channel, but trust the reconstructed image from an authenticated channel.

2) Collage Attack

Because most SASs authenticate streams in a block-wise independent fashion, they are vulnerable to collage attack [33]. Specifically, given two or more authenticated stream blocks, an adversary can forge new streams by swapping the blocks. For example, in Figure 4, original image 4(a) includes 4 tiles.



(a) Original image divided into 4 tiles

(b) Forged image with the same tiles.

Fig.4. Collage attack by re-arranging the original tiles.

The codestream of each tile is a block authenticated independently. Unfortunately, an adversary can re-organize the positions of the tiles so as to produce another image 4(b). Of course, each tile is authentic because the attacker does not modify the tile data. Clearly, the forged image is different from the original image. To defeat this counterfeiting attack, the signature for each block should include the stream identification produced via Session Description Protocol(SDP) [34] and block positions in the stream.

III. Priminaries

Notation

$x \,||\, y$: The concatenation of strings x and y.

$H(\cdot)$: a one-way function.

$\mathbf{Enc}_{n,k}(X)$: Erasure encoding source X of k symbols to produce a codeword of n symbols.

$\mathbf{Dec}_{n,k}(Z)$: Erasure decoding from codeword fragment Z of k symbols to recover the source of k symbols in a (n,k) coding system.

$\mathbf{Sign}(m, P_s)$: Signing algorithm. Given a message m, only the signer owning private key P_s is able to generate the digital signature which can be verified with signer's public key.

A. Erasure Codes

An erasure encoding algorithm Y= $\mathbf{Enc}_{n,k}(X)$ takes a source X of k symbols and produces a codeword Y of n symbols. Given any codeword fragment Z of k symbols and their positions in the codeword Y, a decoder is able to recover source X with the decoding algorithm $\mathbf{Dec}_{n,k}(Z)$. For a good introduction to Reed-Solomon erasure code, please refer to Rizzo's work [35].

B. One-way Hash Function

A one-way hash function $H(m)$ takes a variable-length input string m and converts it to a fixed-length output string, called a hash value, or integrity token. This conversion is in one direction. That is to say, it is easy to compute a hash value $H(m)$ from a pre-image m; however, given a value c, it is hard to find a pre-image m' such that $c= H(m')$. The widely used one-way hash functions are MD5[36] and SHA[37].

C. Message Authentication Code

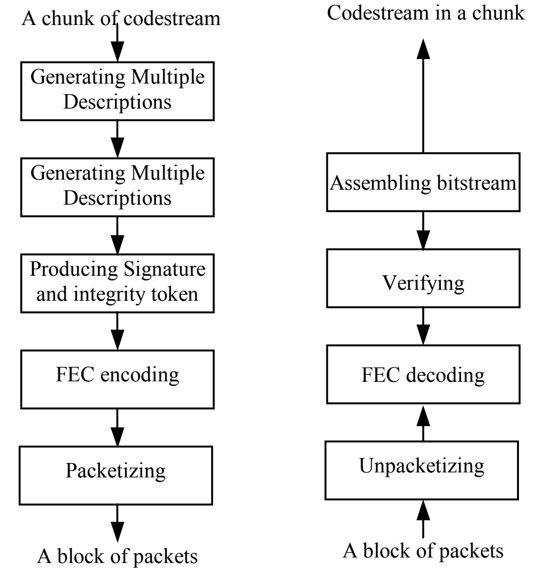
A Message Authentication Code, or MAC for short, is a keyed-hash function [38][39]. MAC requires that the sender and the receiver share a key K. Given a message m, the sender produces the MAC as mac= $H(m \,||\, K)$. The receiver, who knows the key K, can verify the integrity of message m with mac. However, MAC does not provide the non-repudiation evidence because both the sender and the receiver can generate mac.

D. Digital Signature

A digital signature algorithm is a cryptographic tool for generating non-repudiation evidence, authenticating the signed message as well as its origin. A digital signature algorithm includes three modules: $\mathbf{Kgen}(\cdot)$, $\mathbf{Sign}(\cdot)$, and $\mathbf{Ver}(\cdot)$. The $\mathbf{Kgen}(\cdot)$ generates a pair of private/public key. The private key all. With the algorithm $\mathbf{Sign}(\cdot)$ and the private key, the singer generates digital signatures on messages. The verifier, who knows the public key, can validate the signature against the message with $\mathbf{Ver}(\cdot)$. Consider the RSA signature scheme [40][41]. The algorithm $\mathbf{Kgen}(\cdot)$ selects $N = pq$, where p and q are two random primes. Denote $\varphi(N)$=$(p$-$1)(q$-$1)$. Then selects an integer $1 < e < \varphi(N)$ such that e is relatively prime to $\varphi(N)$ and computes an integer d=e^{-1} mod $\varphi(N)$. Assume signer's private key is (d, N) and the public key is (e, N). To sign a message m, the signer computes the signature $\sigma = \mathbf{Sign}(m, d) = (H(m))^d \bmod N$. In the verification precess, $\mathbf{Ver}(m, \sigma, e)$ will indicate whether message m is authentic or not. Specifically, to verify that σ is the claimant's signature on m, a verifier first obtains the claimant's authentic public key (e, N), and then computes $\beta = \sigma^e \bmod N$. If $\beta = H(m)$, the verifier accepts the signature σ on m; otherwise, he rejects the signature. Another commonly used signature algorithm is Digital Signature Algorithm (DSA)[42].

IV. Content-aware Stream Authentication Scheme

Figure 5 illustrates the present SAS for any chunk of Motion JPEG2000 codestreams. Firstly, a chunk is represented in multiple descriptions by interleaving the contiguous image codestreams and partitioning any image codestream into media fragments which are the units for processing. Secondly, we compute a hash value, called integrity token, for each media fragment, and compute a digital signature over all the integrity tokens, block identification and stream identification. Thirdly, the digital signature and the integrity tokens are encoded into codewords with FEC. Fourthly, the codeword generated from signature are divided into signature fragments, and the codeword generated from integrity tokens is divided into integrity fragments. Finally, a media fragment, a signature fragment and an integrity fragment are encapsulated into an IP packet. As a result, a chunk of codestreams is transformed into a block of IP packets. The packets in the block are delivered into the network sequentially.



(a) Preparing Authentic Blocks(Sender)

(b) Verifying the stream (Receiver)

Fig.5.Diagram of the present stream authentication scheme. The sender transforms codestreams in a chunk into packets in an authentic block., while the receiver authenticates each received packet

Figure 5(a) depicts the process of generating authentic blocks at the sender side, and Figure 5(b) illustrates the process of verifying received packets of a block. In this verification procedure, the receiver extracts the media fragment m_i, signature fragment s_i and integrity fragment r_i from each packet P_i in the block. By employing the FEC decoder, the receiver reconstructs the integrity tokens and the signature so as to validate any received media fragment. The authentic media fragments will be assembled so as to recover frames in the nature orders. In the end, the reconstructed frame will be displayed one by one.

Correspondingly, the data structure for the stream is shown in Figure 5. In the figure, the whole stream is divided into blocks. Each block consists of IP packets. Any IP packet P_i consists of integrity fragment r_i, signature fragment s_i and media fragment m_i. Additionally, several media fragments form a MDC.

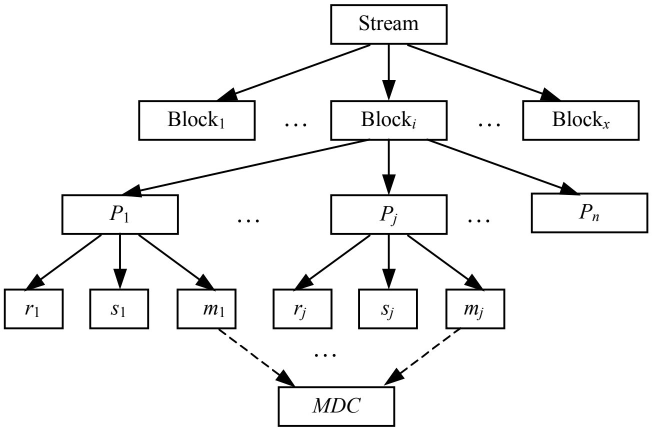


Fig.6. Data structure of a stream. If the length of the stream is unknown in advance, x is ∞.

A. Generating Multiple Descriptions

In this content-aware SAS, any chunk of codestreams is represented in multiple descriptions so that a complete image (frame) can be shown if any description is received. To this end, a chunk of Motion JPEG2000 codestreams is represented in two layers of MDC: Interleaving the frames and amortizing any image codestream into several IP packets.

1) Interleaving Frames

A motion JPEG2000 file [2] includes a lot of chunks, and a chunk includes sets of codestreams of contiguous frames. However, when the frames are transmitted into the network, the receiving order may be different from the sending order because the UDP (User Datagram Protocol) does not provide reliable transmission. For example, the first frame may be received later than the second frame even though the former is sent earlier. In other words, the sending sequence of the packets is not important for authentication. Thus, we ignore the time order requirement for organizing frames in one chunk. Instead, we identify frame sequence based on the IP packet identification number or timestamp in the media data. This modification allows for interleaving frames in one chunk so as to reduce the effect of bursty packet loss on human's visual perception on the received stream.

2) Amortizing Codestreams

As mentioned in subsection II-B, the specification of Motion JPEG2000 permits only JPEG2000 codestream and each frame is encoded independently. Therefore, we can process the Motion JPEG2000 stream frame by frame. Codestream amortization aims for minimizing the loss probability of content of a codestream in case of packet losses. The heuristic method is to load the bitstreams from different resolutions/layers and positions into an IP packet. For example, given a JPEG2000 codestream corresponding to the DWT coefficients shown in Figure 7, we can organize the bitstream in the following intuitive ways:

- ♦ The bitstreams in the different subbands of the same resolution level should be in different IP packets. For example, the bitstreams for A_1, A_2, A_3 should be in different IP packets. Therefore, when the packet encapsulated with A_1 is lost, the packet encapsulated with A_2 or A_3 can show the content with low quality. Alternatively, the code-blocks from different subbands in different positions may be aggregated into one IP packet. For example, bitstreams from codeblocks 1, 6 and 11 are encapsulated into one IP packets.
- ♦ The bitstream corresponding to the same precinct of the original image should be in different IP packets. For example, the bitstreams for A_1, B_1 and C_1 should be in different IP packets.

Certainly, the above guidelines are not compulsory and may be variable in different applications. The media data in an IP packet is called media fragment.

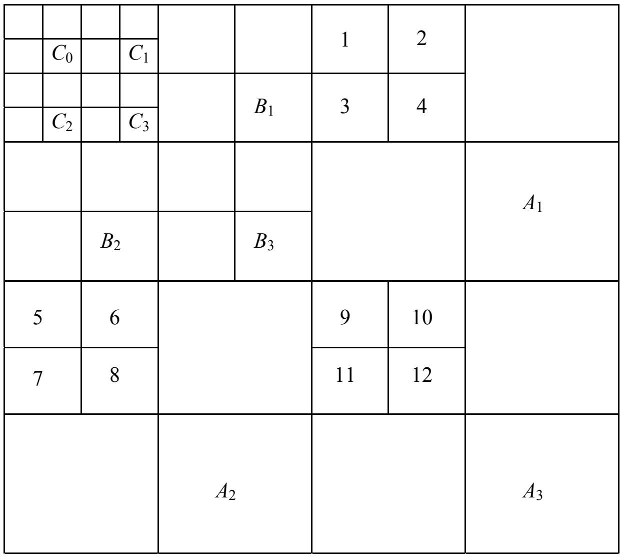


Fig. 7. Code-block and Precinct. An original image is processed with 3-level DWT. A_1, A_2 and A_3 forms a precinct which includes 12 code-blocks.

In all, to represent a chunk in various descriptions, the codestreams are divided into media fragments which are loaded into a number of IP packets, and a set of IP packets is selected for each description so that the media fragments of the packets represent almost the same image. For example, Table I lists the media fragments in packets P_{11} & P_{12}, P_{21} & P_{22}, and P_{31} & P_{32} with respect to image coefficients shown in Figure 7. Packets P_{11}& P_{12} includes bitstreams from A_1, B_3 and C_2; Packets P_{21} & P_{22} includes bitstreams from A_2, B_1 and C_3; Packets P_{31} & P_{32} includes bitstreams from A_3, B_2 and C_1. The bitstreams in lowest resolution (e.g., C_0) will be distributed into each packet uniformly (not shown in Table I). Thus, the media fragments of packets P_{11} and P_{12} form descriptions MDC_1; The media fragments of packets P_{21} and P_{22} form descriptions MDC_2; The media fragments of P_{31} and P_{32} belong to descriptions MDC_3.

TABLE I

Example of MDC

media data in P_{11} & P_{12}	media data in P_{21} & P_{22}	media data in P_{31} & P_{32}
A_1	A_2	A_3
B_3	B_1	B_2
C_2	C_3	C_1

B. Producing Signature and Integrity Token

Since a Motion JPEg2000 stream is divided into blocks which are constructed independently, we focus on a t^{th} block G only. Figure 8 describes the process of generating the digital signature and the integrity tokens for block G. Assume block G includes n packets P_1, P_2, …, P_n. Each packet P_j includes media fragment m_j which is formed in the above subsection IV-A.2. Specifically, the signature and integrity tokens for block G are generated as follows,

- ♦ For media fragment m_j in packet P_j, calculating hash value or integrity token $\hbar_j = H(m_j \parallel j)$, j=1, 2, …, n using a one-way hash function $H(\cdot)$.
- ♦ Calculating the block hash

$$H_G = H(\hbar_1 \parallel \hbar_2 \parallel \ldots \parallel \hbar_n \parallel t \parallel ID),$$

where ID is the stream identification number which is negotiated in advance.
- ♦ Signing the block hash H_G by a digital signature schemeusing the sender's private key K_s, i.e., signature σ=**Sign**(H_G, K_s).

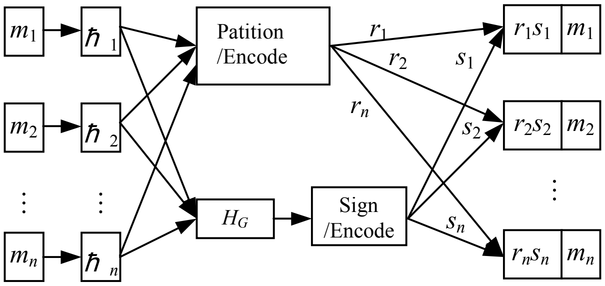


Fig.8. Generating authentication message

C. FEC Encoding and Paketizing

Following the process of generating the signature and integrity tokens, an FEC encoder is employed to encode them so as to tolerate packet losses. Specifically

- ♦ Dividing the concatenation $\hbar_1 \parallel \hbar_2 \parallel \ldots \parallel \hbar_n$ into k symbols H_i, i=1,2,…, k. With the erasure code algorithm such as Reed-Solomon code, a codeword $C_r = \mathbf{Enc}_{n,k}(H_1, H_2, \ldots, H_k)$ is produced.

- Partitioning the codeword C_r into n integrity fragments $r_1, r_2, \ldots, r_n$ equally.
- Similarly, dividing signature σ into k symbols σ_i of the same size, i=1,2, …, k. (Note: σ_i and H_i may be of different sizes.) Then apply FEC on the signature to produce a signature codeword $C_s = \mathbf{Enc}_{n,k}(\sigma_1,\sigma_2,\ldots, \sigma_k)$.
- Partitioning the codeword C_s into n signature fragments $s_1, s_2, \ldots, s_n$ equally.
- Inserting integrity fragment r_j and signature fragment s_j into IP packet P_j which has already included the media fragment m_j, where j=1, 2, …, n. That is to say, packet P_j consists of r_j, s_j and m_j.

D. Delivering Packets

It is well known that lost packets in the Internet often happen in bursts. Discrete Markov chain model [43][44] has been widely used to approximate the behavior of packet losses over the Internet. In the two-state mode, state 0 means that the network packets are transmitted successfully, and state 1 indicates the network packets are lost. With the model, the discrete time step corresponds to the event of sending a packet. The process of the discrete Markov chain undergoing τ discrete time steps is equivalent to the process of sending τ packets through the network. Assume P_{ij} is the transition probability from state i to state j. Denote $p_{01}=\alpha$, $p_{10}=\beta$, then the transition Matrix

$$T=\begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix}=\begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}$$

$$=\begin{pmatrix} 1 & -\alpha \\ 1 & \beta \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & \lambda \end{pmatrix}\begin{pmatrix} \beta & \alpha \\ -1 & 1 \end{pmatrix}(\alpha+\beta)^{-1}$$

where λ=1-α-β< 1, and ι-step transition matrix is

$$T^{\tau}=\begin{pmatrix} 1 & -\alpha \\ 1 & \beta \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & \lambda^{\tau} \end{pmatrix}\begin{pmatrix} \beta & \alpha \\ -1 & 1 \end{pmatrix}(\alpha+\beta)^{-1}$$

$$=\begin{pmatrix} 1 & -\alpha\lambda^{\tau} \\ 1 & \beta\lambda^{\tau} \end{pmatrix}\begin{pmatrix} \beta & \alpha \\ -1 & 1 \end{pmatrix}(\alpha+\beta)^{-1}$$

$$=\begin{pmatrix} \beta+\alpha\lambda^{\tau} & \alpha-\alpha\lambda^{\tau} \\ \beta-\alpha\lambda^{\tau} & \alpha+\alpha\lambda^{\tau^{\tau}} \end{pmatrix}(\alpha+\beta)^{-1}$$

Thus

$$p_{11}^{(\tau)}=\frac{\alpha+\beta\lambda^{\tau}}{\alpha+\beta} \qquad (1)$$

From formula (1), the larger τ is, the smaller $p_{11}^{(\tau)}$ is. Thus, to provide highest probability of content integrity, the time interval between sending two packets representing the same content should be as big as possible. That is to say, a block should be organized MDC by MDC such that the average time interval between sending packets for the same content is sufficiently large. For example, a JPEG2000 image sequence is represented as Figure 9 so that each content has almost the same receiving probability. After organizing the IP packets as the structure shown in Figure 9, the IP packet can be delivered sequentially.

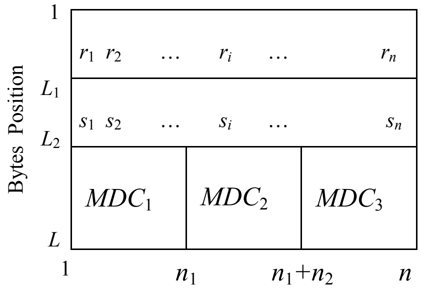


Fig.9. Block structure. The block includes *n* IP packets which are assigned into three MDCs. Each packet includes fragments from signature codeword, integrity message and meda data.

E. Verifying IP Packets

With respect to Figure 10, the verification process roughly reverses the generation process for an authentic block. Based on the FEC coding, at least k packets of a block should be received so as to recover the signature and integrity tokens. Without loss of generality, suppose the first k packets P_1, P_2, …, P_k are obtained successfully. Thus, the receiver retrieves the integrity fragments r_1, r_2, …, r_k from the received packets so as to recover the codeword C_r with decoding algorithm $\mathbf{Dec}_{n,k}(r_1,r_2,\ldots, r_k)$. That is to say, the hash value g_i (i=1, 2, …, n) for the media fragment m_i is obtained. The reconstructed hash for the t^{th} block is

$$h_G = H(g_1 \,\|\, g_2 \,\|\, \ldots \|\, g_n \,\|\, t \,\|\, ID).$$

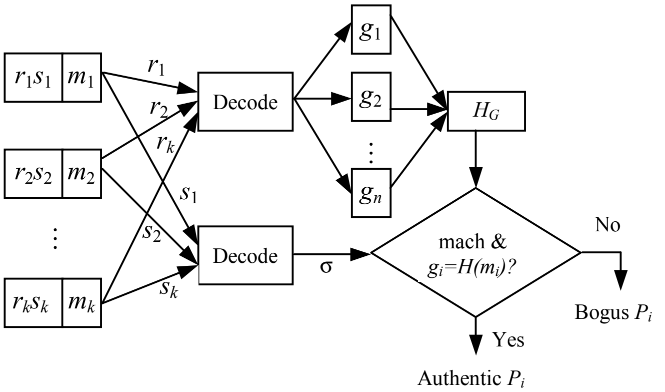


Fig.10. Verifying Packets. Without loss of generality, suppose the first *k* packets are received, the verifier checks the authenticity of received packets. If a tampered packet is used for reconstructing the signature, the verification process should be repeated.

Meanwhile, signature fragments s_1, s_2, …, s_k are retrieved too. With the decoder $\mathbf{Dec}_{n,k}(s_1, s_2, \ldots, s_k)$, the signature σ for the block is reconstructed. Therefore, the signature is able to be verified against h_G with algorithm $\mathbf{Ver}(h_G, \sigma, e)$, where e is the authentic public key of the signer. If not match, the whole block is bogus and discarded. Otherwise, the receiver will check any media fragment m_i against the integrity token g_i. The tampered packets are discarded. Furthermore, if the tampered packet is used for reconstructing the signature, the verification process is repeated with a new set of packets.

V. Performance

In this section, we investigate the performance of the present SAS in the computation cost at the sender side and receiver side. Next, we study the authentication probability because it is another important concern in the stream authentication. Finally, we suggest how to transcode the Motion JPEG2000 stream in a narrowband network so that the devices with limited resource such as PDA can receive the Motion JPEG2000 stream.

A. Computational Cost

Besides the necessary computational cost of Motion JPEG2000 coding, the computational cost at the sender side is to generate the authentication message. The operation includes generation of one signature and two codewords for each block. Although the signature generation is time-consuming, its computational cost is amortized into many IP packets or frames. Hence the computational budget for each packet or frame is small. The computational cost at the receiver end is much smaller because the system parameters may be selected for efficient verification. For example, a properly selected verification key in RSA signature scheme enables the verifying time is less than 1% of signing time. The other computation such as decoding the codewords is lightweight. Thus the proposed solution is efficient in computational cost.

B. Authentication Probability

In the Markov model, suppose stationary probabilities of lost packet and received packet are π_0 and π_1 respectively.

LEMMA 1: Assume the expected burst length to be γ in a lossy network, then four transition probabilities can be expressed as follows:

$$p_{00} = 1- \pi_1/(\gamma\pi_0), \quad p_{01} = \pi_1/(\gamma\pi_0)$$
$$p_{10} = \gamma^{-1}, \quad p_{11} = 1-\gamma^{-1}$$

PROOF. see [11].

To evaluate the authentication probability, we take consideration of the experiment results from Table II in [43], where $\alpha = p_{01} = 0.0192$, $\beta = p_{10} = 0.8454$. Then the expected burst-loss length $\gamma = 1.1829$, and stationary no-loss probability $\pi_0 = 0.9778$, and stationary loss probability $\pi_1 = 0.0222$.

Let the number of lost packets as $L(n)$ in a block including n packets. Then the number of expected lost packets is

$$E[L(n)] = \pi_1 n = 0.0222n.$$

To authenticate the received packets, $X \le n\text{-}k$ in an encoder $\mathbf{Enc}_{n,k}(\cdot)$ FEC, where X is the number of lost packets, i.e.

$$P(X \le n\text{-}k) = \sum_{i=0}^{n-k}\binom{n}{i}\pi_0^{n-i}\pi_1^{i} = \sum_{i=0}^{n-k}\binom{n}{i}0.9778^{n-i}0.0222^{i}$$

For example, when there are 200 IP packets in a block (solid line in Figure 11), $\mathbf{Enc}_{100,93}(\cdot)$ Reed-Solomon code should be employed with authentication probability is about 92%. Usually, for a fix k, the smaller n in $\mathbf{Enc}_{n,k}(\cdot)$ is, the higher the authentication probability is, but the more the payload in each packet is.

C. Capability of Transcoding Stream

In the content delivery, a stream is usually transcoded so as to meet the requirements of network bandwidth and receiver resources [45]. In JPEG2000 specification, an image is segmented into tiles, then each tile is encoded independently using SNR or spatially scalable techniques. Additionally, the motion JPEG2000 stream generated from the present SAS is represented in *MDC*s. Consequentially, it is trivial to transcode the stream. What a transcoder (e.g. proxy) needs to do is to discard the codestreams of the unimportant frames, *MDC*s, resolutions, or layers so as to meet the requirements of network/clients. Unfortunately, the content may be no longer authentic if too many packets are discarded.

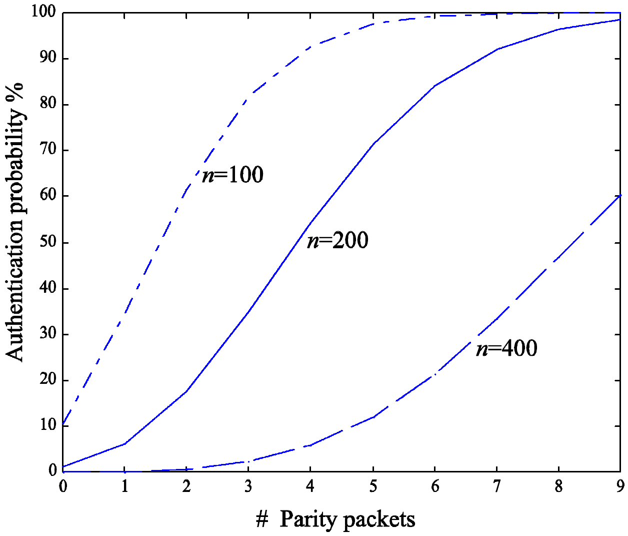


Fig. 11. Authentication probability vs. number of parity packets.

VI. Experiment

Continuing the example shown in Figure 2 where the original image is segmented into 4 tiles and each tile codestream is encapsulated into one IP packet. The stream includes only one block which consists of four packets. In the following, the signer produces an authentic block and sends the packets of the block into the network. The receiver verifies the packets in case of packet loss. This experiment targets for demonstrating the content authentication in the packet loss networks.

A. Generating Authentic Block

A 3-level Discrete Wavelet Transform (DWT) is applied to each tile T_i in Figure 2, i=1,2,3,4. For simplicity, we describe the DWT coefficients directly instead of the JPEG2000 bitstream. Denote the DWT coefficients for the subbands LL_3, LH_3, HL_3, HH_3 to be l_i, and the rest coefficients to be h_i, for any tile i=1,2,3,4. To generate an authentic block, we improve the algorithm proposed in [11] such that the system is resilient to collage attack. Specifically, to generate the authentic block at the sender side, the signer should process as follows.

- Calculate the integrity tokens

  $\hbar_1 = H(l_1 \| h_3 \| 1 \| 3)$ (2)

  $\hbar_2 = H(l_2 \| h_4 \| 2 \| 4)$ (3)

  $\hbar_3 = H(l_3 \| h_1 \| 3 \| 1)$ (4)

  $\hbar_4 = H(l_4 \| h_2 \| 4 \| 2)$ (5)

  In the above equation $H(x \| y \| u \| v)$, x and y forms a media fragment while u and v represent their locations. Divide $(\hbar_1 \| \hbar_2 \| \hbar_3 \| \hbar_4)$ into 3 symbols equally, denoted as b_1, b_2, b_3.
- Calculate the integrity fragments

  $(r_1, r_2, r_3, r_4) = \mathbf{Enc}_{4,3}(b_1, b_2, b_3)$.

  Where r_1, r_2, r_3, and r_4 are of the same length. Here, we utilize Reed-Solomon algorithm $\mathbf{Enc}_{4,3}$ code so as to tolerate one packet loss within this block.
- Compute the block hash

  $h_G = \mathrm{H}\ \hbar_1 \| \hbar_2 \| \hbar_3 \| \hbar_4 \| ID)$ (6)

  where ID is the well-known identification of the streaming, the block identification is omitted since there is only one block in the exemplary stream.
- Sign on h_G with signer's private key d so as to provide a signature σ=**Sign**(h_G, d). Evenly divide σ into 3 segments σ_1, σ_2, σ_3.
- Calculate signature fragments

  $(s_1, s_2, s_3, s_4) = \mathbf{Enc}_{4,3}(\sigma_1, \sigma_2, \sigma_3)$,

  where s_1, s_2, s_3, and s_4 are of same length. Then form the packets as Figure 12. From Figure12, we know that there are two MDCs. MDC_1 includes l_1, h_3, l_2 and h_4, MDC_2 includes l_3, h_1, l_4 and h_2. Either MDC_1 or MDC_2 can represent the whole image because anyone has the bitstreams from all tiles. Thus, the image content is complete if only one packet is lost.

r_1	r_2	r_3	r_4
s_1	s_2	s_3	s_4
h_3	h_4	h_1	h_2
l_1	l_2	l_3	l_4
P_1	P_2	P_3	P_4

Fig.12. Block including four IP packets. *P*$_1$ and *P*$_2$ forms one *MDC*, and *P*$_3$ and *P*$_4$ forms another *MDC*. Each packet include bitstreams of low subbands of one tile and bitstream of high subbands of another tile, as well as signature fragment and integrity segment.

B. Verifying the Packets

After the block in the Figure 12 is multicast packet by packet, a receiver may obtain packets P_1, P_2 and P_4 only, but miss packet P_3. Thanks for the FEC coding, the receiver can check the authenticity of the received packets. The verification process is as follows.

- Based on the Reed-Solomon coding algorithm, b_1, b_2, and b_3 can be recovered from r_1, r_2, and r_4, i.e., $\hbar_1$, $\hbar_2$, $\hbar_3$, $\hbar_4$ are available. h_G is deduced easily following equation (6).
- Similarly, the receiver also recovers σ_1, σ_2, and σ_3. That is to say, the signature σ is available.
- With the public key of the signer, the receiver checks whether the signature σ matches h_G. If not, the receiver will reject all the P_1, P_2 and P_4. Otherwise, the receiver will check them based on equations (2), (3), and (5).
- For those authentic packets, the receiver will re-assemble the DWT coefficients in the nature order. For those lost bitstream e.g. h_1 and l_3 due to packet losses, the media fragment data are set to 0 and image enhancement technologies may be applied to increase the image quality.

The recovered image is shown in Figure 13 if there is no tampering except one packet loss. From the recovered image, we observe that the tiles for "Nasdaq" and "2pts,at" are intelligible although their qualities are low. That is to say, the content of the image is reserved even though the quality is reduced.

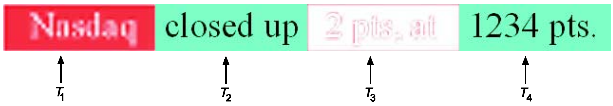


Fig. 13. Recovered image with authentic packets. The lost packet *P*$_3$ consists of the subband data of high resolution of tile *T*$_1$ and the subband data of low resolution of tile *T*$_3$. Hence, the qualities of both tiles are degraded but recognizable

VII. Conclusion

Although stream authentication attracts a lot of researchers, it is still open for a widely acceptable solution. This paper points out that the previous SASs [6]-[17] are vulnerable to collage attack, and ignorant of content integrity in a lossy network. To defense against collage attack, our scheme incorporates the unique stream identification, block identification and packet sequences into the generation process of the signature and integrity tokens. The signature and integrity tokens are amortized into all the packets using the Forward Erasure Code so as to tolerate packet losses. We also represent a chunk of codestreams in multiple descriptions by exploiting the property of Motion JPEG2000 codestream. The Multiple descriptions not only increase the resilience of packet loss, but also reduce content ambiguity. The present scheme can be extended to other stream format such as MPEG4.

References

[1] T. Turletti, and C. Huitema, "RTP Payload Format for H.261Video Stream, " RFC 2032, 1996

[2] Takahiro Fukuhara, and David Singer, "15444-3 Amendment 2, Motion JPEG 2000, Motion JPEG 2000 Version 2, MJP2 derived from ISO Media File Format, " ISO/IEC JTC 1/SC 29/WG1 N2780F, 24/01/2003

[3] D. S. Taubman, and M. W. Marcellin, "JPEG2000 Image Compression Fundamentals: Standard and Practice, " Kluwer Academic Publishers, 2000

[4] ISO 154447 ITU-T Recommendation T.800, http://www.jpeg.org

[5] Majid Rabbani, and Rajan Joshi, "An overview of the JPEG2000 still image compression standard, " *Signal Processing: Image Communication*, 17, pp. 3-48, 2002.

[6] Rosario Gennaro, and Rankaj Rohatgi, “How to Sign Digital Streams, ” CRYPTO'97, LNCS 1294, pp.180-197

[7] Chung Kei Wong, and Simon S. Lam, “Digital Signatures for Flows and Multicasts, ” IEEE ICNP'98

[8] A.Perrig, R.Canetti, D.Tygar, and D.Song, “Efficient Authentication and Signature of Multicast Streams over Lossy Channels, ” IEEE Symposium on Security and Privacy, 2000.

[9] Sara Miner, and Jessica Staddon, “Graph-Based Authentication of Digital Streams, ” IEEE Symposium on Security and Privacy, 2001

[10] Jung Min Park, Edwin K. P. Chong, and Howard Jay Siegel, “Efficient Multicast Packet Authentication Using Signature Amortisation, ” IEEE Symposium on Security and Privacy, pp.227-240, 2002

[11] Jung Min Park, Edwin K. P. Chong, and Howard Jay Siegel, “Efficient multicast stream authentication using erasure codes, ” *ACM Transactions on Information and System Security*, 6(2):258-285 , 2003

[12] M.Bellare, R.Canetti, and H.Krawczyk, “Keying Hash Functions for Message Authentication, ” CRYPTO'96, LNCS 1109, pp.1-15, 1996

[13] P. Rohatgi. “A Compact and Fast Hybrid Signature Scheme for Multicast Packet Authentication, ” 6th ACM Conference on Computer and Communication Security, pp.93-100, 1999.

[14] A.Perrig, R.Canetti, D.Tygar, and D. Song, “Efficient and Secure Source Authentication for Multicast, ” ISOC Network and Distributed System Security Symposium (NDSS), 2001

[15] Philippe Golle, and Nagendra Modadugu, “Authenticating Streamed Data in the Presence of Random Packet Loss, ” NDSS 2001

[16] Alain Pannetrat, and Refik Molva, “Efficient Multicast Packet Authentication, ” NDSS 2003

[17] Yongdong Wu, Di Ma, and Changsheng Xu, “Efficient Object-based Stream Authentication, ” Indocrypto 2002, LNCS 2551, pp.354-367

[18] S. Deshpande, and W. Zeng, “Scalable streaming of JPEG2000 images using Hypertext Transfer Protocol,” ACM Multimedia, 2001

[19] Colin Perkins, Orion Hodson, and Vicky Hardman, “A Survey of Packet-Loss Recovery Techniques for Streaming Audio,” IEEE Network, 1998, pp.40-48

[20] Raphael Bornard, Emmanuelle Lecan, Louis Laborelli, and Jean-Hugues Chenot, “Missing Data Correction in Still Images and Image Sequences,” ACM Multimedia 2002

[21] B. W. Wah, X. Su, and D. Lin, “A Survey of Error-Concealment Schemes for Real-Time Audio and Video Transmissions over the Internet, ” Int'l Symposium on Multimedia Software Engineering, 2000, pp. 17-24

[22] Andre Kaup, “Error Concealment for SNR Scalable Video Coding in Wireless Communication,” Visual Communications and Image Processing 2000}, Vol. 4067, pp. 175-186.

[23] Thinh Nguyen, and Avideh Zakhor, “Distributed Video Streaming with Forward Error Correction,” International Packetvideo Workshop 2002 http://www.pv02.org/

[24] Nick Feamster, and Hari Balakrishnan, “Packet Loss Recovery for Streaming Video, ” International Packetvideo Workshop 2002

[25] M. van der Schaar, and Hayder Radha, “Unequal packet loss resilience for fine-granular-scalability video, ” *IEEE Transactions on Multimedia*, 3(4):381-394, 2001

[26] Alatan} A.A. Alatan, M. Zhao, and A. N Akansu, “Unequal error protection of SPIHT encoded image bit streams,” *IEEE Journal on Selected Areas in Communications1*, 8(6):814-818, 2000

[27] A.E. Mohr, E.A. Riskin, and R.E Ladner, “Unequal loss protection: graceful degradation of image quality over packet erasure channels through forward error correction,” *IEEE Journal on Selected Areas in Communications*, 18(6):819-828, 2000

[28] I. Moccagatta, S. Soudagar, J. Liang, and H. Chen, “Error-resilient coding in JPEG-2000 and MPEG-4,” *IEEE Journal on Selected Areas in Communications*, 18(6):899-914, 2000

[29] Joohee Kim, R.M. Mersereau, and Y. Altunbasak, “Error-resilient image and video transmission over the Internet using unequal error protection,” *IEEE Transactions on Image Processing*, 12(2):121-131, 2003

[30] Wai-Tian Tan, and A. Zakhor, “Video multicast using layered FEC and scalable compression, ” *IEEE Trans. on Circuits and Systems for Video Technology*, 11(3):373-386, 2001

[31] Y. Wang, M. Orchard, V. Vaishampayan, and A. R. Reibman, “Multiple Description Coding using Pairwise Correlating Transforms,” *IEEE Transactions on Image Processing*, 10:3(351-366), 2001.

[32] John Apostolopoulos, Tina Wong, Wai-tian Tan, and Susie Wee, “On Multiple Description Streaming with Content Delivery Networks,” IEEE Infocom 2002

[33] M. Holliman, and N. Memon, “Counterfeiting attacks on oblivious block-wise independent invisible watermarking schemes, ” *IEEE Transactions on Image Processing*, 9(3):432-441, 2000

[34] M. Handley, and V. Jacobson, “SDP: Session Description Protocol,” RFC 2327, 1998

[35] L. Rizzo, “Effective erasure codes for reliable computer communication protocols, ” ACM Computer Communication Review, 27, 1997.

[36] R. Rivest, “The MD5 Message Digest Algorithm,” RFC 1321, 1992

[37] National Institute of Standards and Technology, “Secure Hash Standard (SHS),”' FIPS Publication 180-1, 1995.

[38] FIPS PUB 198, “The Keyed-Hash Message Authentication Code,” 2002

[39] Federal Information Processing Standards Publication (FIPS PUB) 180-2, Secure Hash Standard, 2002

[40] R. L. Rivest, A. Shamir, and L. M. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” Communications of the ACM, Vol. 21, No. 2, pp. 120-126, 1978.

[41] RSA Labs, PKCS\#1 v2.1, RSA Cryptography Standard, 2002

[42] National Institute of Standards and Technology, “Proposed Federal Information Processing Standard for Digital Signature Standard (DSS),” Federal Register,Vol. 56, No. 169, pp. 42980-42982, 1991.

[43] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, “Measurement and modelling of the temporal dependence in packet loss,” IEEE Infocom 1999.

[44] J. Wenyu, and H. Schulzrinne, “Modeling of packet loss and delay and their effects on real-time multimedia service quality,” 10th Int. Workshop Network and Operations System Support for Digital Audio and Video, 2000.

[45] Susie J. Wee, and John G. Apostolopoulos, “Secure Scalable Streaming Enabling Transcoding without Decryption,” IEEE International Conference on Image Processing, 2001.

Yongdong Wu Received the B.A and M.S. in Automation Control from Beijing University of Aeronautics and Astronautics in 1991 and 1994 respectively, and the Ph.D. degree in Pattern Recognition and Intelligent Control from Institute of Automation, Chinese Academyof Science, 1997.

He is currently a Senior Scientist with Infocomm Security Department, Institute of Infocomm Research ($\mathbf{I^2R}$), a-star, Singapore. His interests includes multimedia security, e-Business, Digital Right Management and Network security.

Robert H. Deng Received his B.Eng from National University of Defense Technology, China, in 1981, his M.Sc and PhD from Illionis Institute of Technology, Chicago, in 1983 and 1985, respectively.

Since 1985, he has held various university and research positions. His research interests include error control coding, computer networks and information security. He has more than 120 technical publications on international conference and journals and has served on numerous program committees of international conferences. He received the University Outstanding Researcher Award from the National University of Singapore in 1999. He is presently Principal Research Scientist and Manager of Infocomm Security Department, Institute for Infocomm Research, Singapore.