Multi-Source Stream Authentication Framework in Case of Composite MPEG-4 Stream

Tieyan Li, Huafei Zhu, and Yongdong Wu

Institute for Infocomm Research (I^2R),
21 Heng Mui Keng Terrace, Singapore 119613
{litieyan, huafei, wydong}@i2r.a-star.edu.sg

Abstract. Multimedia community is moving from monolithic applications to more flexible and scalable integrated solutions. Stream authentication is more complex since a stream may consist of multiple sources and be transcoded by intermediate proxies. In this paper, we propose a multi-source stream authentication (mSSA) framework based on MPEG-4 stream format. We describe the overall authentication architecture and elaborate the encoding, hashing, signing, amortizing and verifying methods used in the basic scheme. Further on, we utilize advanced cryptographic primitives-aggregate signature schemes, to reduce the signatures' size and improve the performance. We illustrate the scheme and discuss the extensions. Our analysis shows that the scheme is secure and efficient.

1 Introduction

Multimedia syndication has been put forward for years, for instance, [1] proposed an "InfoPyramid" scheme to interrelate different formatting and conversion options of multimedia objects together with composition strategies for complex multimedia documents. How to protect (*w.r.t.* access control and authentication) these complex media contents is a critical challenge in many applications. The security requirements are different for end roles: on the one hand, media providers want to protect the access to their content; on the other hand, the end users must make sure the authenticity of the content. However, most commercial Multimedia Digital Right Management (DRM) systems, *i.e.* Windows Media Rights Manager (WMRM) for Microsoft [2], are built for the former purpose and ignore the end users' requirement. We hereby focus on authentication of multi-source scalable streams. Moreover, instead of working on meta-data descriptions (*e.g.* specifying usage rules, XrML [3] in DRM), we work directly on specific streaming format-MPEG-4 [4,5] so that standard scalable MPEG-4 stream of multiple sources can be manipulated and verified flexibly.

We give two examples to motivate stream compositions and their respective authentication requirement. First, suppose a local news channel, authorized to broadcast live news from CNN, may change the subtitle from English to its local language. The local channel has to make new commitment on the modified parts so that the combined media stream can be verified by the end users. Thinking

S. Qing et al. (Eds.): ICICS 2005, LNCS 3783, pp. 389–401, 2005.

broadly, any tailoring of the original works such as movie advertisement, multi-screening, digital art creation *etc.* are categorized into this class. Second, consider a more complex mix-an interactive video conference, where multiple sources are able to synthesis, distribute and display of customized media content. Similarly, the verification must be robust enough to tolerate various media handlings such as transcoding, filtering, mixing, tilting and switching.

In this paper, we propose mSSA framework for authenticating multi-source scalable stream. Our basic stream authentication scheme, relying on traditional stream authentication mechanisms with signature amortization, enables end-to-end authentication with the transcoding operations. Furthermore, we identify two types of transcoding operations: *truncation-only* with which a proxy is only allowed to truncate a partial sub-stream of an original stream; and *grafting* where a proxy can not only truncate a partial stream, but also insert another sub-stream. In addition, to save bandwidth, we utilize aggregate signature schemes to reduce the size of multiple signatures into one. This first proposed multi-source stream authentication scheme can verify a flexible coded stream in many ways, extend easily and scale well. It can also be adapted into standard DRM systems. Our analysis shows that the scheme is secure and cost-effective.

Paper organization: Section 2 reviews some related works on traditional as well as adaptive stream authentication schemes. We then describe the mSSA framework in section 3. Section 4 introduces the preliminaries, MPEG-4 stream format and depicts the basic authentication mechanisms. In section 5 we elaborate multi-source authentication schemes with type-1 and type-2 transcoding operations. Following on, we analyze the security and performance issues in section 6. At last, we conclude our paper and point out our future tasks.

2 Related Works

Single source stream authentication schemes. Stream authentication schemes have been intensively studied [10,11,12,13,14,15,16]. The traditional schemes can be categorized as hash graph-based [12], tree-based [11] and symmetric key-based [10]. Other approaches [13,14,15,16] assume an erasure channel, such as the emerging wireless networks-Bluetooth, WPAN, *etc.* where packets are lost from time to time. Erasure codes [9] are then used to tolerate arbitrary patterns of packet loss. However, these works concentrate on end-to-end stream authentication where only single stream source and single receiving end are assumed by default. These traditional stream authentication schemes are failed on Sender-Proxy-Receiver SPR model where an intermediate proxy can manipulate the streaming packets for adapting with fluctuant network conditions. However, these basic works provide us valuable building blocks on stream encoding, packing, hashing, signing and verifying processes.

Adaptive stream authentication schemes. Recently, stream authentication in the SPR model was studied in the literature [18,19,20]. [18] proposed a secure MPEG-4 stream authentication scheme that allows a proxy flexibly manipulate streaming packets while they can still be verified by the final receiver. Although

the Unequal Loss Verification (ULV) scheme is secure and efficient, it didn't address the multi-source authentication problem in detail. Suzuki *et al.* in [19] proposed a multimedia content delivery system that protects the end-to-end authenticity of adaptive multimedia. Moreover, the paper used a multi-hop signature scheme for aggregation. Unfortunately, the paper didn't assume an erasure channel where packets are lost arbitrarily. Also, it worked only on meta-data processing, instead of processing streaming content itself. Recently, Gentry *et al.* in [20] proposed two new provably secure schemes, LISSA and TRESSA, that ensure secure streaming media authentication with adaptive proxies. However, it didn't address the multi-source authentication problem. Additionally, it had the problems similar to [19] as it only considered generic multimedia content simply as message blocks instead of real stream format like MPEG-4.

3 mSSA Overview

Our basic stream authentication scheme borrows many building blocks from aforementioned traditional stream authentication schemes. Generally, a stream is divided into groups of packets and each group is processed independently. In an erasure channel, the objects/layers of a group at different priority levels are given unequal protection levels via erasure correction coding (ECC) [7]. The various layers are then packed into a group of packets. Furthermore, following the amortizing scheme of SAIDA [13], the producer signs on the hash value of each group, instead of on every packet. The group hash is generated such that recipients are able to verify the source of a stream. It is generally believed that by amortizing the authentication data over a group of packets the verification overheads are much smaller than that of signing on every packet.

Unlike traditional multicast erasure channel where no packet modification is allowed, the proxies in our model are not only a passive packet forwarder, but also have an active role of transcoding and then redistributing the stream into the end network. The transcoding mechanism allows a proxy to discard data layers from the lowest priority layer to higher layers until the resource restrictions are met. For example, Fine Granular Scalability (FGS) [6] is such a scalable mechanism to distribute an MPEG-4 stream efficiently and flexibly over heterogeneous wired and wireless networks. This transcoding strategy differs from packet dropping strategy. Because the transcoded stream can tolerate the same number of packet loss as the original stream, the error-resilience capability is not decreased. Thus, a receiver is able to verify authenticity of the packet origin even if the stream is transcoded. The objective of scalable stream authentication is to authenticate all possible resulting streams after legitimate manipulations on scalable coded streams. We further identify the following two types of transcoding operations (refer to section 5.3 for illustration):

- ◇ Type-1 transcoding operation, *w.r.t.* truncation-only: A proxy is only allowed to truncate one or multiple sub-streams from an original stream. that is to say, there is only one valid stream source.

◇ Type-2 transcoding operation, *w.r.t.* grafting: A proxy can selectively truncate one or multiple portions of an original stream, and insert one or more portions from multiple stream sources. All of them form a new stream with multiple sources.

Note that for type-1 transcoding operation, a verifier might verify not only the stream source's signature, but also the signatures from the proxies for committing their transcoding operations. For type-2 transcoding operation, a verifier has more complex verification procedures involving multiple stream sources and proxies (see section 4 and 5 for details).

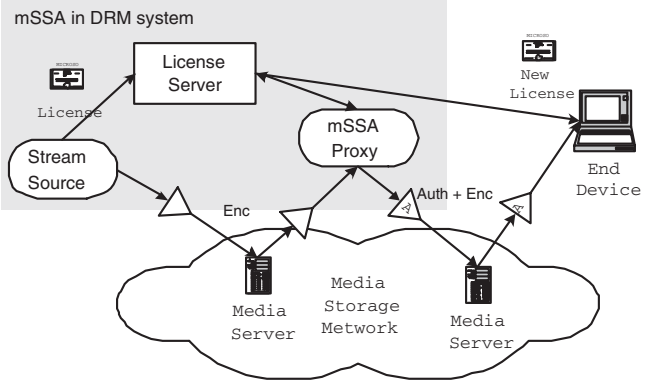

Fig. 1. Multi-Source Stream Authentication (mSSA) Framework

Fig. 1 sketches the overall mSSA framework. It can be embedded into the current DRM systems as a complimentary media stream authentication scheme. To process stream authentication, the framework consists of three main parts: stream preparation by sources, stream transcoding by mSSA proxies and stream verification by end users. We describe them as follows:

Part 1: Preparation The streaming video objects are first encoded according to the MPEG-4 standard. Each source prepares the packets for the object group based on the priorities of the video objects and layers. The source then generates authentication data including integrity units and its signature. The authentication data is amortized over the group of packets. The protected stream as a whole packet group is then ready to be delivered to the proxies[1].

Part 2: Transcoding On receiving the protected stream, a proxy can either apply type-1 transcoding operations to fit the stream into some narrow bandwidth, or employ another protected stream and apply type-2 transcoding operations. In either case, or combined transcoding operations, the proxy needs to sign the new stream with some aggregate signature scheme AggSigning (see

[1] The basic authentication scheme is presented in section 4.

section 5). The process can be repeated and finally, the constructed stream is ready to be downloaded by the end users.

Part 3: Verification The verification procedure is actually reversing the above two processes. Suppose a receiver retrieves a stream as well as its authentication data from some proxy. It then unpacks, decodes the packets. With recovered authentication data, the receiver can verify the stream integrity and signatures with proper aggregate signature verification scheme AggVfing.

4 Basic MPEG-4 Stream Authentication Scheme

4.1 Preliminaries and Notations

Notations: m denotes a message; $h(.)$ is a collision resistant hash function. K_s and K_p denote the private key and public key of the producer; Sign is the signing algorithm: $\sigma = \mathsf{Sign}(K_s, m)$; Vf is the verification algorithm: $\mathsf{Vf}(K_p, \sigma, m)$ outputs $\{true, false\}$. We also introduce some useful tools like Merkle Hash Tree (MHT) [8] and Erasure Correction Coding (ECC). *Merkle hash tree* has been widely used in many security applications, since it has good security property that it commits on one hash digest over a set of data items. In this paper, we use MHT for generating the integrity data. *Erasure correcting codes* are good means for error resilience of content dissemination in erasure channel. An ECC system with symbols in finite field $GF(2^w)$ includes two modules: encoding $Enc_{n,k}(\cdot)$ and decoding $Dec_{n,k}(\cdot)$, where n is the codeword length and k is the message length.

4.2 Syntactic Structure of MPEG-4 Stream

The scheme is based on structure of packets, we first introduce the encoding and packing of an MPEG-4 stream. According to [4,5], an MPEG-4 presentation is divided into sessions including units of aural, visual, or audiovisual content, called media objects. A Video Sequence (denoted as VS, or group) includes a series of Video Objects (VOs). Each VO is encoded into one or more Video Object Layers (VOLs). Each layer includes information corresponding to a given level of temporal and spatial resolution, so that scalable transmission and storage are possible. Each VOL contains a sequence of 2D representations of arbitrary shapes at different time intervals that is referred to as a Video Object Plane (VOP). VOPs are divided further into MacroBlocks (MBs) of size 16×16. Each MB is encoded into six Blocks $B_1, B_2, \cdots, B_6$ of size 8×8 when a 4:2:0 format is applied. In an MPEG-4 stream, VOs such as foreground objects and background objects, may have different priorities, indicated as *visual_object_priority* taking values $1 \sim 7$ from lowest to highest priority. In MPEG-4 syntax, each object layer has *visual_object_layer_priority* to represent the importance of different layers. The layer with the highest priority, called the base layer, contains data representing the most important features of the video sequence, while additional layers, called enhancement layers, progressively assigned with lower priorities,

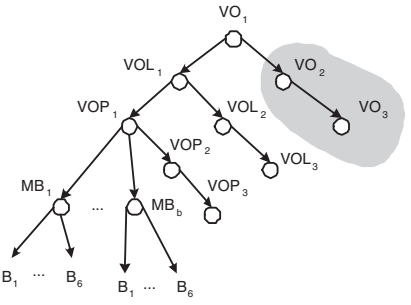

Fig. 2. A typical tree structure of an object group with priority levels from VO_1, VO_2, ... down to $VOLs$, $VOPs$, MBs and $Blocks$. The shadow part that covers subtrees (VO_2-VO_3) can be cut off by the transcoding operation.

contain data that further refine the quality of the base layer. The source generates a flow for each layer and assigns to it a unique discarding priority. In Fig.2, we illustrate a typical hierarchical object tree in one visual object group of an MPEG-4 stream.

4.3 Generating Authentication Data

The very first step of the basic stream authentication scheme is generating authentication data, which relies on some hashing and signing mechanisms on the packets. Given a group of encoded packets with above tree structure, we are able to generate the hash values bottom-up [18], which is a method similar to the TRESSA hashing scheme in [20], as shown in table 1.

Where R is the root of the object group and G_{ID} is the group ID. Thus, the producer can sign the group hash h_G using its private key K_s and get the signature as:

Table 1. Generating authentication data

Authentication Data Generation
At bottom layer, compute hash values of Blocks:
$h_{B_i} = h(Block_i \parallel i), i \in \{1, 2, \cdots, 6\}$
Following that, compute hash values of macroblocks:
$h_{MB_j} = h(h_{B_1} \parallel h_{B_2} \parallel \cdots \parallel h_{B_6})$
.....
Upward, suppose an upper layer node N has a set of child nodes $C = (C_1, C_2, ..., C_c)$, compute the hash value as:
$h_N = h(C_1 \parallel C_2 \parallel \cdots \parallel C_c)$
.....
At last, we can calculate the group hash as:
$h_G = h(R \parallel G_{ID})$

$$\sigma = \mathsf{Sign}(K_s, h_G) \tag{1}$$

By signing once on the root of the object group, the originator actually commits a whole virtual object group to the receivers. Suppose a stream consists of n virtual object groups, n signatures are to be generated to authenticate the stream. Hereafter, we use authentication data (denoted as $\lambda = \langle \sigma, H_G \rangle$) to represent both the signatures and the whole or partial[2] group hash values.

4.4 Amortizing Authentication Data

After generating the authentication data, the authentication data is to be amortized into the packets using existing information dispersal algorithm as in [14]. We employ ECC encoders to encode them and amortize them onto the packets before sending them out over an erasure channel (We use the method introduced in [15]). For simplicity, we process the authentication data uniformly with the same encoding rate as of the highest priority layer. For a set of n packets $P = \{P_1, P_2, \cdots, P_n\}$ and their authentication data H_G and σ in a (k,n) erasure channel. The encoding procedure uses the systematic ECC algorithms $Enc_{2n-k,n}(\cdot)$ and $Enc_{n,k}(\cdot)$ and computes the codeword C_r as integrity units $r_1, r_2, \cdots, r_n$, the codeword C_s as signature units $s_1, s_2, \cdots, s_n$, respectively. Next, we append integrity unit r_j and signature unit s_j on packet P_j, for all $j = 1, 2, \cdots, n$.

4.5 Verifying Authentication Data

The verification process includes unpacking, decoding and verifying, which reverses the generation process. Based on the erasure coding, at least k out of n packets of a group should be received in order to recover the authentication data. Suppose k packets $\{P_1, P_2, \cdots, P_k\}$ are received successfully. The integrity units $\hat{r}_1, \hat{r}_2, \cdots, \hat{r}_k$ and the signature units $\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_k$ are recovered from the received packets. With the decoder $Dec_{n,k}(.)$ and $Dec_{2n-k,n}(.)$, the authentication data is recovered as $\hat{\lambda} = \{\hat{\sigma}, \hat{H_G}\}$. Then, the signature can be verified with algorithm $\mathsf{Vf}(K_p, \hat{\sigma}, \hat{H}_G)$, where K_p is the public key of the stream source. If $\mathsf{Vf}(.)$ is true, then continue to verify the integrity unit; if not, the object group is bogus and discarded. To this end, the end user reconstructs the hash tree H'_G according to the formulas in table 1 and compares it with the extracted integrity unit $\hat{H_G}$. We desire $H'_G = \hat{H_G}$ for successful verification.

5 Multi-Source MPEG-4 Stream Authentication

5.1 Basic Transcoding Process

On receiving a stream, a proxy is allowed to do type-1 or type-2 transcoding operations before retransmission. First, we focus on truncate-only transcoding dis-

[2] Depending on how much of a hash tree will be taken as the authentication evidence (the integrity unit), which is the core part of *flexible verification* [18].

cussed in section 3. Based on MPEG-4 stream structure of Fig. 2, truncation-only means that we preserve certain (important) branches of an MHT and truncate other (unimportant) branches to fit the stream into narrow network bandwidth. *I.e.,* the shadow part in Fig. 2 could be truncated if necessary. In this example, we discard the subtree ($VO_2 - VO_3$), retain the hash of the subtree root and keep the subtree (VO_1). Apparently, the original authentication data λ has to be changed to a new one λ'. The new data shall contain the original signature σ, the new integrity unit H_{VO_1} and the new signature σ_P (signed by the proxy on the root of the subtree for committing its modification). We get the new authentication data $\lambda' = \{\sigma, \sigma_P, H_{VO_1}\}$. Using above amortization method, we can append them onto the packets and send them out.

Secondly, for grafting transcoding: suppose in above case, the subtree ($VO_2 - VO_3$) is replaced by another subtree (T_A) with authentication data $\lambda_A = \{\sigma_A, H_{T_A}\}$. Apparently, the authentication data of the composite stream should be $\lambda'' = \{\sigma, \sigma_P, H_{VO_1}, \sigma_A, H_{T_A}\}$. And so forth for additional grafting operations (with limitations only by the system capabilities). Same as above, λ'' is amortized onto the packets and sent out.

Noted that as the above processes continue, the signature size will be linearly increased and proportional to the signing parties. The overheads of these signatures will soon become unaffordable after several transcoding operations. Fortunately, there are two methods to reduce the size of the authentication data. Firstly, we can reduce the size of integrity unit by dropping some hash values of the subtrees. Support we generate a hash tree over totally n leaf nodes. If we cut off all hash values of those leave nodes, we remain $n-1$ hash values. Bottom up, if we cut off lower m levels of the tree, we have only $n/2^{m-1} - 1$ hash values. However, the tradeoff of using this method is that it can not refine the verification. Thus, we hereby concentrate on reducing the signature size that do not loss any verification granularity as well as security. We employ a cryptographic primitive, aggregate signature schemes such as [21][22], to significantly reduce the size of multiple signatures into one only.

5.2 Transcoding with AggSigning and AggVfing

One special variant-a sequential signature scheme [22][3] can be applied into our type-1 transcoding operation for reducing the signature size. Here, the proxy P_i takes as inputs the signature σ_{i-1} from its predecessor and all integrity data Σ_i as a whole, and outputs aggregate signature σ_i. Thus, $\lambda_i = \{\sigma_i, \Sigma_i\}$. The new authentication data is amortized onto the packets and sent to the end users. Suppose an end user received at least k out of n packets of a stream in order to recover the authentication data. The integrity unit $\hat{\Sigma}_i$ and the signature unit $\hat{\sigma}_i$ are recovered from the received packets. Then, the signature can be verified with above algorithm $\mathsf{AggVf}(\hat{\sigma}_i, \hat{\Sigma}_i)$ with corresponding public keys $\{PK_1, \cdots, PK_i\}$. If $\mathsf{AggVf}(.)$ is true, then continue to verify the integrity unit; if not, the object group is bogus and discarded. The end user reconstructs the hash tree Σ_i'

[3] Since the scheme is built on standard primitives like RSA, that is widely adopted.

and compares it with $\hat{\Sigma}_i$. If $\Sigma'_i = \hat{\Sigma}_i$, it means successful verification. If not, it indicates a mismatch between two hash trees.

5.3 Illustration

In this section we illustrate above transcoding concepts with an encoded MPEG-4 stream. Shown in Fig. 3, the images are generated artificially with English characters as foreground objects. The background objects are considered important and are coded with $Enc_{5,4}(\cdot)$ ECC, but the foreground objects correspond to layers of lower priority and are not coded for error resilience. We draw some assumptions to simplify the demo: 1, we ignore the experiments for packet loss; 2, image pixels are used instead of DCT coefficients for data units in the process of hash computation which could be sufficient for demonstrating our scheme; 3, each layer includes two contiguous bit planes; and 4, we consider only two critical concepts, VO and VOP, in MPEG-4 visual objects. The syntactic structure of the simplified image sequence is shown in Fig. 4.

In order to allow authentication of the transcoded stream, for type-1 trans coding, the proxy generates the hash for the covering-subtree value HVF and incorporates one patch into each packet. For type-2 transcoding, the proxy cuts and pastes a subtree VF' replacing VF. Figure 5 and 6 illustrate the syntactic structure of type-1 and type-2 transcoded objects, respectively. Then, the proxy sends the modified packets to the clients.

In type-1 transcoding, *e.g.,* the MPEG-4 stream is adapted to a narrow bandwidth wireless network, the proxy filters out the foreground objects and

Fig. 3. Sample image sequence. The background images forms the basic layer. While the foreground objects (English words) form a sentence.

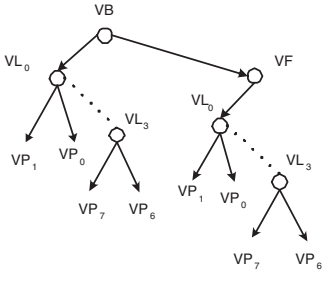

Fig. 4. Syntactic structure of image objects. Where VB denotes the background object, and VF denotes the foreground object.

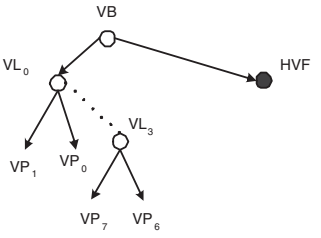

Fig. 5. Syntactic structure of type-1 transcoded objects

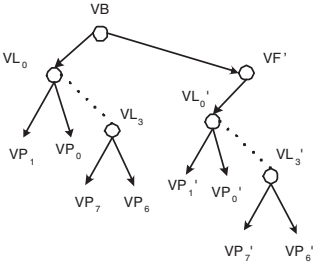

Fig. 6. Syntactic structure of type-2 transcoded objects

Fig. 7. Received sample image sequence with transcoded foreground objects

just transmits the background objects to clients. In type-2 transcoding, *e.g.,* the proxy changes the foreground objects with some advertisements. Fig. 7 illustrates the image sequence received by the clients for type-2 transcoding.

5.4 Discussions

Above we illustrate how a portion of a stream can be replaced with a portion of another stream. In fact, multi-source stream can be flexibly composed. For instance, in case of multi-screening, a super composer can summarize multiple streams in a big screen, where each stream is scaled down to fit into its small screen. To sign such a super stream is straightforward, the composer may aggregate the signatures attached with those streams and generate its aggregate signature. However, if a portion of a stream is grafted as in Fig. 6, *i.e.* suppose VF' is a subtree rooted at VB' where only VB' is signed but not VT'[4], then where can we sign? Fortunately, the scheme can be extended easily by providing proof of trust from the signed root to the replacing portion. *I.e.* we need to prove VF' is indeed a subtree of VB' by taking as evidence the corresponding hash values. Another way is to sign on every possible portions of the stream so that we take that portion directly and generate the aggregate signature. Both methods may introduce some overheads.

One critical concern is whether the scheme can be embedded into the legacy DRM systems, as mentioned in section 3. First, let's review the process of scalable multimedia. Raw multimedia stream is compressed once with a scalable

[4] In our scheme, each stream is signed once on the root implicitly.

coding scheme and the resulting codestream can be decoded adaptively. The protection strategies must conform with the scalability of the codestream, thus the fundamental property of (generally upper layer's) authentication and encryption is to preserve the scalability. Current DRM systems are focusing on protecting illegal access to the content. End users download content from content distributors and separately, obtain the decryption keys from a license server. Authentication is applied after the coding phase and normally before the encryption process. Here, authentication/verification and encryption/decryption is two separate processes since they are based on different (Asymmetric/symmetric cipher) techniques. In a shared key case, it is also possible to use authenticated encryption schemes such as OCB [23][5].

6 Security and Performance Analysis

The security of our scheme relies on the security of the Merkle hash tree and the aggregate signature scheme. Fortunately, Merkle hash tree has very nice security properties [8] and the security of aggregate signature is analyzed in [21,22]. Thus, we concentrate on the integrity unit that is how much percentage of a stream is verifiable. Then, we analyze the computational cost of each party in the scheme.

6.1 Percentage of Verification

Assuming the authentication data is recoverable as it has the highest priority as the base layer. Additionally, assuming an erasure channel with independent packet losses, given ρ the packet loss probability. The group of n packets transferred over the erasure channel may have probably $\binom{n}{k}\rho^{n-k}(1-\rho)^k$ packets received. The verification delay for a group of n packets is $O(n)$. In our definition, only those recovered content of a received stream can be verified. From the recovered authentication data $\hat{\lambda_i}$ and $\mathsf{AggVf}(\hat{\sigma_i}, \hat{\Sigma_i})$. We know the validity of the signatures. Further on, if we don't receive enough packets to recover all stream, we are not able to verify the full integrity unit $\hat{\Sigma_i}$ from the reconstructed hash tree Σ_i'. Let $P = \Sigma_i'/\hat{\Sigma_i}$ denote the percentage of verification. We claim that the rate of the reconstructed hash tree Σ_i' over the recovered hash tree $\hat{\Sigma_i}$ from the received packets directly determines the verification rate P over the object group, given the signature on $\hat{\Sigma_i}$ is valid.

6.2 Computational Cost

We assume that the computational cost relates to security operations without encoding/decoding costs. Additionally, the computational cost for signature generation and verification depends on the signature scheme selected and normally, the signature verification is considered much faster than signature generation.

[5] This will be a totally different scheme and not discussed further due to space limitation.

And, for an MHT with n leaf items, the total number of hash operations is roughly $2n$.

For type-1 transcoding, there is only one stream source with one signature generation and $2n$ hashing operations. Each intermediate proxy P_i must first verify $i-1$ former signatures and generate one sequential aggregate signature for all its transcoding operations. The final receiver has to verify all the signatures of the aggregate signature, but at relatively lower cost. On the other hand, the receiver will also spend time on reconstructing the (probably partial) hash tree over the object group. For type-2 transcoding, each stream source S_i generate one signature and $2n_i$ hashing operations. Each intermediate proxy P_i must first verify $i-1$ former signatures and generate one general aggregate signature for all its type-2 transcoding operations. The cost at the final receiver is the same as that of type-1.

7 Conclusions and Future Works

We proposed the first secure multi-source authentication scheme for composite MPEG-4 stream. The scheme works under the general assumption of "erasure channel", but can be adapted to "polluted erasure channel", *e.g.* by using distillation code [17]. We elaborated how the encoding, hashing, transcoding, signing and verifying mechanisms are integrated in the mSSA framework. The scheme extends easily and scales well. Our analysis shows that it is secure and cost-effective. The detailed scheme can complement to some standard DRM platforms to protect multimedia content. In the futue, we will develop the innovative prototype within a legacy DRM framework.

References

1. J.R. Smith, R. Mohan, C. S. Li, *Scalable Multimedia Delivery for Pervasive Computing.* In Proc. ACM Intl. Conf. on Multimedia (ACMM'99), Orlando, FL, 1999.
2. Microsoft, *Architecture of Windows Media Rights Manager,* `http://www.microsoft.com/windows/windowsmedia/howto/articles/drmarchitecture.aspx`
3. eXtensible right Markup Language (XrML). `http://www.xrml.org`
4. ISO/IEC 14496-1:2001 Information Technology-Coding of Audio-Visual Objects-Part 1: Systems.
5. ISO/IEC 14496-2:2003 Information Technology-Coding of Audio-Visual Objects-Part 2: Visual.
6. Weiping Li, *Overview of fine granularity scalability in MPEG-4 video standard,* IEEE Trans. on Circuits and Systems for Video Technology, 11(3):301-317, 2001.
7. A. E. Mohr, E. A. Riskin, 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.
8. R. C. Merkle, *A certified digital signature*, Crypto'89, Lecture Notes on Computer Science, Vol. 0435, pp. 218-238, Spriner-Verlag, 1989.
9. M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, *Practical loss-resilient codes,* in Proc. 29th Annual ACM Symposium on Theory of Computing (STOC'97), El Paso, TX, May 1997.

10. A. Perrig, R. Canetti, J. D. Tygar, and D. Song. *Efficient authentication and signature of multicast streams over lossy channels.* In Proceedings of the IEEE Symposium on Research in Security and Privacy (S&P'00), pages 56-73, May 2000.
11. P. Golle and N. Modadugu, *Authenticated streamed data in the presernce of random packet loss,* in Proc. Network and Distributed System Security Symposium (NDSS'01), San Diego, CA, Feb. 2001.
12. S. Miner and J. Staddon. *Graph-based authentication of digital streams.* In Proceedings of the IEEE Symposium on Research in Security and Privacy (S&P'01), pages 232-246, May 2001.
13. J. M. Park, E. K. Chong, and H. J. Siegel. *Efficient multicast packet authentication using signature amortization.* In Proceedings of the IEEE Symposium on Research in Security and Privacy (S&P'02), pages 227-240, May 2002.
14. J. M. Park, E. Chong, and H. J. Siegel. *Efficient multicast packet authentication using erasure codes.* ACM Transactions on Information and System Security (TISSEC'03), 6(2):258-285, May 2003.
15. A. Pannetrat and R. Molva, *Efficient multicast packet authentication,* in Proc. Network and Distributed System Security Symposium (NDSS'03), San Diego, CA, Feb. 2003.
16. Maxwell N. Krohn, Michael J. Freedman, David Mazires *On-the-Fly Verification of Rateless Erasure Codes for Efficient Content Distribution.* IEEE Symposium on Security and Privacy (S&P'04), California, USA.
17. C. Karlof, N. Sastry, Y. Li, A. Perrig, and J. Tygar, *Distillation codes and applications to DoS resistant multicast authentication,* in Proc. 11th Network and Distributed Systems Security Symposium (NDSS'04), San Diego, CA, Feb. 2004.
18. Tieyan Li, Yongdong Wu, Di Ma, Huafei Zhu, Robert H. Deng, *Flexible Verification of MPEG-4 Stream in Peer-to-Peer CDN*, 6th International Conference on Information and Communications Security (ICICS'04), LNCS 3269, Spain, 2004.
19. Takashi Suzuki, et al. *A system for end-to-end authentication of adaptive multimedia content.* Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS'04), Sept. 2004.
20. C. Gentry, A. Hevia, R. Jain, T. Kawahara, and Z. Ramzan. *End-to-End Security in the Presence of Intelligent Data Adapting Proxies: the Case of Authenticating Transcoded Streaming Media.* J. Selected Areas of Communication, Q1, 2005.
21. Dan Boneh, Craig Gentry, Ben Lynn, Hovav Shacham. *Aggregate and Verifiably Encrypted Signatures from Bilinear Maps.* EUROCRYPT 2003.
22. Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, Hovav Shacham. *Sequential Aggregate Signatures from trapdoor one-way permutations.* EUROCRYPT 2004.
23. P. Rogaway, M. Bellare, J. Black and T. Krovetz, *OCB: A block cipher mode of operation for efficient authenticated encryption*, in Proc. of 8th ACM CCS'01.