Multimedia Systems 8, 444–458 (2003)
Digital Object Identifier (DOI) 10.1007/s00530-002-0066-z

Multimedia Systems

A digital rights management scheme for broadcast video

Sabu Emmanuel[1], **Mohan S. Kankanhalli**[2]

[1] School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 693798 (e-mail: asemmanuel@ntu.edu.sg)
[2] School of Computing, National University of Singapore, Kent Ridge, Singapore 119260 (e-mail: mohan@comp.nus.edu.sg)

Abstract. Many watermarking techniques have been proposed for digital video and digital broadcast video. In general, they address the seller's or broadcaster's concerns, such as copyright-violation deterrence, copyright-violation detection and copy protection. Some of them provide for data integrity (tamper proofing), which is a shared concern of both subscriber and broadcaster. In all these cases, the watermark generation and embedding is performed or controlled by the seller or broadcaster. Therefore, a broadcaster with malicious intentions could, with manipulation, falsely implicate an innocent subscriber in copyright violation. This could be a major concern for subscribers.

In this paper, we propose an integrated solution to manage the broadcaster's copyright and subscriber's false-implication concern (subscriber's rights) for digital video broadcasts. The proposed novel approach makes use of interactive watermarking techniques and protocols to help protect digital rights of all parties involved in video broadcasting.

Key words: Digital rights management – Copyright protection – Subscriber's rights protection – Digital watermarks – Digital video broadcasting – Video watermarking – Pay TV

1 Introduction

As the digital technologies for the creation, processing, storage and transfer of media develop, the protection of intellectual property, and associated issues, faces a new challenge. The easy replication of digital media, otherwise an advantage, is a major concern in the context of copyright infringements. Many implementations tackle the copyright infringements through copy-control mechanisms implemented in hardware. The mechanisms may make use of watermarking technology to allow one-time copy, many-times copy and copy-never [20].

Copyright-protection schemes, on the other hand, try to resolve the copyright in the event of violations. This really does not stop a violator from making another copy of the copyrighted material. To prove the copyright ownership and integrity of the digital data, the seller will insert a watermark (copyright information) into the digital data [13,30,31]. This watermark, however, will not help in identifying the copyright violator. To identify the copyright violator, the watermarks used should be unique for individual customers. In the event of copyright violations, the seller can check for the watermark in the digital media and identify the legal recipient of that copy. The seller may then initiate legal proceedings against the legal recipient of that copy of the media.

In these schemes, the watermarking process is completely controlled by the seller. Therefore, it may be possible that the seller may, knowingly or unknowingly, create a second watermarked copy with the same watermark, which had been already used for someone else, and distribute the second copy to a second party. Later, the seller might try to sue the first legal recipient for the copy seen by the second recipient. This false implication by sellers is a major concern for customers. Moreover, an unscrupulous customer might freely distribute the legal copy, which has been received, and contest in the court saying that the illegal copies originated from the seller. Clearly the court has no proof to order in favor of the seller, as the seller also can create duplicate watermarked copies. Cleverly designed watermarking protocols are therefore required for ameliorating customer's false-implication concerns.

In this era of digital communications, often the transaction of digital media is between people of distant locations with little knowledge of each other. In this scenario, asserting one's copyright becomes difficult. The sellers need to know the physical identity of the customer, so if copyright violations have taken place, the customer could be traced using the physical identity. Also, the customers would want to know whether the seller from whom the customer is getting the copyrighted material really owns the copyright or distribution rights. Otherwise, the customer could be faced with legal proceedings from the genuine owner or seller. Therefore mutual authentication and proof of ownership or distribution rights are essential components in these transactions.

This paper focuses on digital video, since it constitutes a significant part of the entertainment and education systems. The digital-video data could be transmitted through the Internet or could use terrestrial, satellite or cable digital TV transmission systems.

Since video data is voluminous, addressing the abovementioned concerns are all the more complex and challenging. Firstly, the sellers may prefer to handle the video in the

compressed domain. Compression of video reduces the data size and hence creates lower storage requirements. Secondly, the sellers may want to broadcast the video to multiple receivers at the same time, which reduces the transmission channel bandwidth requirement. Therefore, the schemes proposed must support compressed-domain processing and broadcasting/multicasting environment.

One example of such an environment is where film production studios broadcast movies over the Internet to cinema halls worldwide [17]. The pay-channel mode of digital-video broadcast is another example of such an application where the broadcaster is the seller and the subscribers (subscribing receivers) the customers. Since there is a subscription fee involved, the broadcasters would like to have a confidentiality requirement against the non-subscribers (non-subscribing receivers) in addition to the copyright requirement against the subscribers. By confidentiality requirement, we mean that non-subscribers should not be able to view the video clearly; the idea being to force non-subscribers to subscribe to the channel to view it clearly. Usually, scrambling is employed to obtain this confidentiality [21,34]. The copyright issue requires that all the subscribers get individually watermarked video. This is for tracing any individual subscriber who compromises the owner's copyright. The broadcasting requirement, however, necessitates a single copy of the video being transmitted to all the receivers. Clearly, meeting both these demands simultaneously is a challenging task.

Our research presents a novel integrated solution to address the copyright-violation concern of broadcasters, false-implication concern of subscribers, and the authentication of each party to the other in a digital-video broadcasting scenario.

The rest of the paper is organized as follows. In Sect. 2, the related work is discussed, and we then describe our proposed scheme in Sect. 3. In Sect. 4, we present the analysis, and Sect. 5 contains discussion followed by the conclusion in Sect. 6. A preliminary version of this paper appeared in [10].

2 Related work

Techniques for hiding watermarks in digital data have steadily grown more sophisticated and increasingly robust against attacks. Many video researchers have used them to provide copyright protection for video.

The European Esprit VIVA project uses the watermarking technique for broadcast monitoring [30]. The broadcast materials are watermarked in the spatial domain prior to broadcasting, and the watermark is detected using a correlation detector. The broadcast chain consists of D/A conversion, A/D conversion, MPEG-2 compression, MPEG-2 decompression, D/A conversion and A/D conversion. The watermark can be detected even though the watermarked video undergoes all this processing. This technique can be used for verification of commercial transmissions, assessment of sponsorship effectiveness, statistical data collection and analysis of broadcast content. But this scheme does not support individual watermarking for copyright-violation detection in a broadcasting environment and also cannot be used for subscription-based video broadcasts where a confidentiality requirement is required against non-subscribers.

There have been a number of research efforts to address the broadcaster's copyright concern, such as the Chameleon scheme by Anderson and Manifavas [1], Watercasting by Brown, Perkins and Crowcroft [3], Chu, Qiao and Nahrstedt's secure multicast protocol with copyright protection [4], Nark by Briscoe and Fairman [2], WHIM by Judge and Ammar [15], and Parviainen and Parnes's large-scale distributed watermarking of multicast media through encryption [26]. Efficient integration of solutions to the subscriber's concerns into the above schemes appears to be difficult.

Chameleon is a scheme that allows a single broadcast cipher to be decrypted into slightly different plain texts by users with slightly different keys [1]. As acknowledged by the authors, the watermarking capability of this scheme is rather limited for MPEG video. Since the watermark bits are very few, the number of distinct watermarks are also few. This affects the scalability for broadcasting. Our proposal does masking in the compressed domain but unmasks leaving behind a watermark after MPEG decoding. Therefore, the watermarking can be performed to the just-noticeable-distortion (JND) level within the perceptual quality of the video.

Watercasting is a technique that has each receiver in a multicast group receive a slightly different version of the multicast data [3]. This scheme requires that the source watermark, encrypt and transmit n copies of the data. The network bandwidth requirement is high, as the source transmits n copies. Each sender must trust the chain of network routers. A chain of trusted network providers is required. Each of them has to be willing to reveal their tree topology to each sender. It also does not offer a solution to distinguish the copies of receivers on the same subnet. Our scheme requires only one masked copy. Therefore, the resource requirements both at the source and on the network are less when compared to the this case. Our method does not ascribe any active role to the network routers yet can distinguish every receiver.

The protocol presented by Chu et al. creates two watermarked streams, assigns a unique random binary sequence to each user and uses this sequence to arbitrate between the two watermarked streams [4]. The efficiency is hampered by the need to watermark, encrypt and transmit two copies of the stream and by the significant amount of key message traffic. Also, the authors state that it may be susceptible to collusion attacks.

The Nark system presents a number of modular mechanisms to enable secure sessions tailored to each individual multicast receiver [2]. In addition to security, it also proposes solutions for non-repudiation and copyright protection, essentially using the Chameleon scheme. Other than the limitations of Chameleon, it also requires a tamper resistant processor at each receiver.

WHIM is a scheme that makes use of a hierarchy of intermediaries for creating and embedding watermarks [15]. This scheme suffers from low watermark embeddability. Also, each sender must trust the chain of active network intermediaries and network providers. Since the scheme does not combine the watermarking and decryption process at the receiver in one single process, the watermarking process can be bypassed.

The method presented by Parviainen and Parnes [26] creates two distinctly watermarked copies of each media packet. Both copies are then encrypted with two different randomly generated encryption keys and are then broadcast or multicast.

Any given receiver has access to the key of only one of the two encrypted packets of one media packet. For a media with k packets, the method requires $2k$ keys, and any one receiver possesses k keys. But this scheme has only limited collusion resistance as acknowledged by the authors.

The subscriber's concerns have been addressed in the papers by Qiao and Nahrstedt [27] and Memon and Wong [23]. Qiao and Nahrstedt propose an owner–customer watermarking protocol in which the customer generates a random bit sequence controlled by a secret key, which is known only to the customer. It is then signed, encrypted and transmitted to the owner. The owner then generates another random bit sequence that depends on the random bit sequence of the customer and a key known only to the owner. The owner then uses this new random bit sequence to watermark the original data. The watermarked data is then encrypted, signed and transmitted to the customers. In this protocol, the owner can hand over the same watermarked data to another customer, which can then be used by that customer. Therefore this protocol does not solve the problem of the subscriber's (customer's) concern. This protocol also does not support the broadcasting environment.

Memon and Wong present an elegant interactive buyer–seller watermarking protocol based on public-key cryptography to solve the subscriber's concerns about image data [23]. In this protocol, the seller does not get to know the exact watermarked copy that the buyer receives. Therefore, the seller cannot create copies of the original content containing the buyer's watermark. In case the seller finds another copy of the watermarked data, the seller can prove to a third party from which buyer the data has originated. This scheme cannot be applied directly to video broadcasting. It needs the video encrypted using the subscriber's public-key to be transferred to the subscriber, which means we cannot use broadcasting. Apart from that, public-key encryption of video data is an expensive affair in terms of computational costs. However, their solution can be modified to the mask-based watermarking scheme, described in the next section, to address the subscriber's concern for broadcast video. We discuss the proposed scheme in detail in the next section.

3 Our scheme

Before beginning the description of the proposed scheme in detail, we briefly describe what are the main objectives of the proposed scheme and also the techniques used.

Objective. The scheme aims to provide the following:

(i) Copyright-violation detection through the use of digital watermarking techniques.
(ii) Address the subscriber's concern through public-key cryptosystem- based watermarking protocols.
(iii) Mutual-authentication through public-key cryptosystem-based mutual authentication protocols.
(iv) Confidentiality requirement against non-subscribers through mask-based scrambling.

The scheme aims to consume as few resources as possible, such as computing power and bandwidth. It also aims to provide the dynamic join and leave functions required for pay per view per time. All these objectives are to be met for compressed-domain video broadcasting.

Architecture overview. In our method, the broadcaster of the video first creates a masked video by blending/embedding an opaque mask frame onto the original video or compressed video, frame by frame. This masking implements the confidentiality requirement, therefore non-subscribers are forced to subscribe to the broadcast for viewing the channel clearly. The masked video is created only once and it can be broadcast through any medium (air, network, etc.).

Then the receiver who wants to join the broadcast initiates a protocol for mutual authentication, which addresses the subscriber's concern. At the end of the protocol, the situation would be as follows:

– Both the subscriber and the broadcaster would know to whom they are talking.
– The subscriber would have transacted the payment information for the subscription.
– The broadcaster would have handed over the access-control data, the unmasking frame.

The protocol makes use of public-key cryptography.

The subscribers then unmask the received masked video using an unmasking frame (customized for each subscriber) leaving behind an invisible watermark (which is robust, as explained in Sect. 4.3) in the form of a residue in the unmasked video. The unmasking process is a single atomic process, which results in the simultaneous insertion of two watermarks into the unmasked video. One of the watermarks is for addressing copyright-violation detection, and this watermark is created by the broadcaster and is known only to the broadcaster. The second watermark is to address the subscriber's concern. This watermark is not known completely to either broadcaster or subscriber. Therefore, neither of them can create or remove it alone. Since the insertion process is combined with the unmasking process as a single process, the subscriber is forced to perform watermarking to see the video clearly.

The masking process is done in the transform (compressed) domain at the encoder for the compressed-domain processing, and for processing in the spatial (raw) domain, the masking is done in the spatial domain itself. The unmasking process is carried out in the spatial domain by the decoder. The unmasking frame is transferred to the subscribers through an interactive protocol and is designed to solve the subscriber's concerns. The proposed technique is depicted in Fig. 1.

We will now explain in detail the mask blending/embedding process, the interactive protocol and the unmasking procedure.

3.1 Confidentiality requirement

Whenever there is a subscription fee involved for joining the broadcast, this invokes a confidentiality requirement against non-subscribers. Usually this is addressed by video scrambling. We approximate this by the mask blending/embedding step which overlays a translucent mask of controllable opacity. This serves to inhibit the viewing clarity of non-subscribers and induces them to become legal subscribers. The mask blending can be done in the raw domain or in the compressed domain. First, we explain in detail the raw-domain mask blending followed by the compressed-domain mask blending.

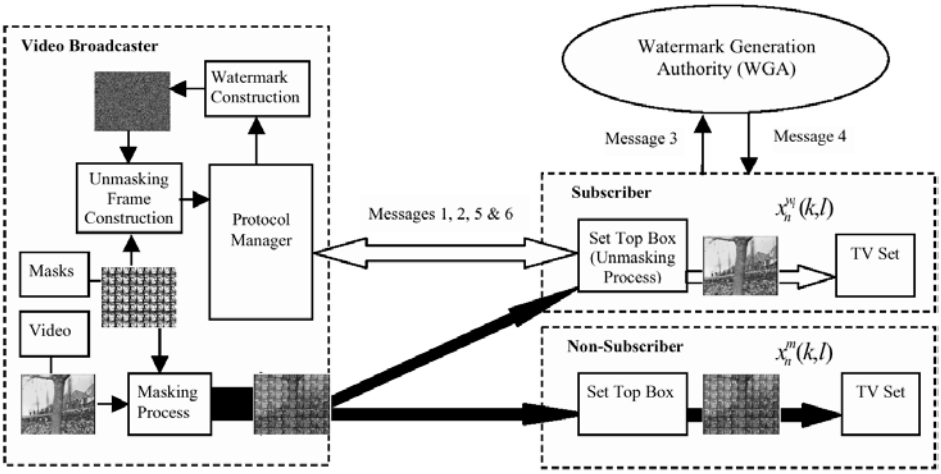

Fig. 1. The proposed scheme

3.1.1 Raw-domain mask-blending process

A video can be considered to be a set of frames. Let us denote the nth frame as $x_n(k,l)$. Each frame consists of $K \times L$ pixels. Next the broadcaster constructs a mask frame $v(k,l)$ of dimension $K \times L$ with the view that the video frames are obscured after mask blending. The mask $v(k,l)$ is blended with the video, frame by frame using the following expression:

$$\forall k,l: \quad x_n^m(k,l) = \alpha\, x_n(k,l) + \beta\, v(k,l), \qquad (1)$$

where $x_n^m(k,l)$ is the nth masked video frame and α, β are scaling factors such that $\alpha + \beta = 1$ and $0 < \alpha, \beta \leq 1$. The masking process simulates the scrambling effect. The scaling factors are necessary to adjust the strength of the mask [16]. The scaling factors are assumed to be public, therefore they don't need to be kept secret. Equation (1) defines the mask-blending process in the spatial domain. The masked video is then broadcast. The receivers who are non-subscribers would only be able to view $x_n^m(k,l)$, which is obscured. The requirement here is that the output of the decoder or set-top box to the display should be $x_n^m(k,l)$ for non-subscribers. Next, we explain how we meet this requirement in the MPEG-2 compressed domain.

3.1.2 Compressed-domain processing

Often video data is stored in a compressed format due to the huge volume of video. There are two ways of dealing with such data. The first method is to decompress the data to obtain the spatial-domain data, process the spatial-domain data and then recompress back. However, this is not a good solution on two accounts. First, decompression and recompression require more computing power. Second, if the compression scheme were lossy, it would result in more loss to the data. The second method is to process the data in the compressed domain itself, which does not involve loss due to recompression and requires less computing power.

In our scheme, we assume that the MPEG-2 compressed video is stored in the broadcaster's database and that the compressed data stream is available for broadcasting. The decoder or set-top box at the receiver decompresses the received MPEG-2 data before sending it to the display. For a non-subscriber, the data sent to the display is $x_n^m(k,l)$ (the obscured masked video).

This can be achieved by appropriately processing the video in the compressed domain at the encoder before broadcasting. The obscured output at the non-subscriber's decoder would be given by Eq. (1): $x_n^m(k,l) = \alpha x_n'(k,l) + \beta v(k,l)$. The term $x_n'(k,l)$ is used instead of $x_n(k,l)$ to reflect the loss due to MPEG-2 compression. We first show how $\alpha x_n'(k,l)$ is obtained by processing the quantized error DCT (discrete cosine transform) coefficients and then the addition of $\beta v(k,l)$ to $\alpha x_n'(k,l)$ is shown. Figure 2 depicts the compressed-domain mask-blending process.

We use the following notation in this section: $\langle a; b\rangle$ refers to an 8×8 pixel block, where a is the DC DCT coefficient, b represents the 63 AC DCT coefficients, $\{\langle a; b\rangle\}$ refers to all the blocks of a frame, and $\phi\,[\{\langle a; b\rangle\}]$ represents operator ϕ applied to the frame.

3.1.2.1 Compressed-domain scaling of video frames. We now describe the computing of $\alpha x_n'(k,l)$. Figure 2 shows the MPEG-2 stream passed to the VLC (variable-length coder) decoder and demultiplexer box that outputs error DCT coefficients, motion vectors and control parameters. We use these motion vectors and control parameters for scaling the video frames and also for mask blending. The quantized error DCT coefficients are then scaled by a factor α, as required by Eq. (1). The box SW in Fig. 2 is a switch. The next box, which implements "subtract $\frac{(1-\alpha)\times 128\times N}{s}$ from DC DCT" where N comes from the $N \times N$ point DCT, here $N = 8$, and s is the intra DC differential quantization step size (a control parameter used during MPEG-2 encoding of video frames), is necessary due to the use of the fixed prediction value of 128 for the intra macroblocks at the decoder. This will result in $\alpha x_n'(k,l)$. The proof is excluded here for brevity and is available in [11]. After multiplying the error DCT coefficients by the scaling factor (if intra-coded blocks, the DC is shifted), we round the result to an integer as required by MPEG-2, as shown in Fig. 2. This rounding causes losses to data; however, it is inevitable loss in cases where blending is done after scaling. Next, we discuss how we obtain the addition of $\beta v(k,l)$ to the $\alpha x_n'(k,l)$ as required.

3.1.2.2. Compressed-domain mask blending. The appropriately generated mask error DCT coefficients are then added to

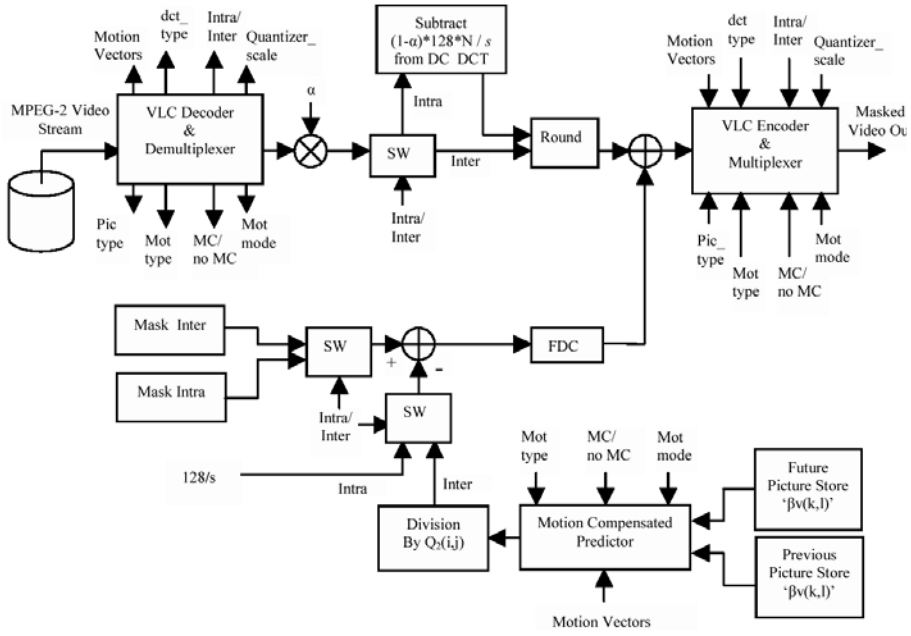

Fig. 2. Compressed-domain mask-blending process

the rounded, scaled, quantized video error DCT coefficients as shown in Fig. 2. These values along with the motion vectors and control parameters are then VLC encoded and transmitted. We use the same motion vectors and control parameters obtained at the output of the VLC decoder and demultiplexer (Fig. 2) for generating the appropriate mask error DCT coefficients. This means that the same motion vectors and control parameters used for video encoding are used for generating the appropriate mask error DCT coefficients. Further, any constraint placed on the control parameters would apply to the video encoding as well as to the mask.

The terms $\alpha x'_n(k,l)$ and $\beta v(k,l)$ can be considered results of separate inputs to the MPEG-2 decoder system. One input to the decoder results in $\alpha x'_n(k,l)$, and the other produces $\beta v(k,l)$. We begin with the mask $v(k,l)$ (luminance only), which is scaled by β to obtain $\beta v(k,l)$. The blending process is done only for the luminance blocks. From $\beta v(k,l)$, we create two scaled versions of this frame: one frame $mask\_i(k,l)$ to mask corresponding intra-coded blocks (which have no motion compensation) of the video, and another frame $mask\_n(k,l)$ to mask inter-coded blocks of the video:

$$mask\_i(k,l) = \mathrm{IDCT}\left[\left\{\left\langle \frac{\beta V(0,0)}{s}; \frac{\beta V(f_1,f_2)}{Q(i,j)} \right. \right. \right. \quad (2)$$
$$\left. \left. \left. \text{for } i = f_1 = 1,\ldots,7,\ j = f_2 = 1,\ldots,7 \right\rangle \right\}\right];$$

$$mask\_n(k,l) = \mathrm{IDCT}\left[\left\{\left\langle \frac{\beta V(0,0)}{Q_2(0,0)}; \frac{\beta V(f_1,f_2)}{Q(i,j)} \right. \right. \right. \quad (3)$$
$$\left. \left. \left. \text{for } i = f_1 = 1,\ldots,7,\ j = f_2 = 1,\ldots,7 \right\rangle \right\}\right].$$

We assume that the intra and inter quantization matrix values are the same for the AC DCT coefficients. Therefore,

$$Q(i,j) = \frac{2 \times q\_scale \times IntraQmat(i,j)}{32} \quad \text{for } i = 1,\ldots,7,\ j = 1,\ldots,7$$
$$= \frac{2 \times q\_scale \times InterQmat(i,j)}{32} \quad \text{for } i = 1,\ldots,7\ j = 1,\ldots,7, \quad (4)$$

where $IntraQmat(i,j)$ is the intra quantization matrix and $InterQmat(i,j)$ is the inter quantization matrix.

$$Q_2(i,j) = \frac{2 \times q\_scale \times InterQmat(i,j)}{32} \quad \text{for } i = 0,1,\ldots,7,\ j = 0,1,\ldots,7. \quad (5)$$

The factor q_scale is the quantization scale factor assumed to be constant. The intra DC differential quantization step size s can be 2, 4 or 8.

For masking an intra-type block at location (x,y) in the frame, we just need to add the DCT of the prediction error between the block in $mask\_i(k,l)$, at the same location (x,y), and $128/s$ as seen in Fig. 2. For masking the forward, backward or interpolated types of block at location (x,y), we add the DCT of the prediction error between the block in $mask\_n(k,l)$ at the same location and the motion compensated prediction for the inter-coded block. The motion compensated prediction for inter-coded block is divided by $Q_2(i,j)$. This will not be lossy as long as s is divisible by $Q_2(0,0)$. For the skipped blocks, nothing needs to be added; just the macroblock skip information is to be transmitted. But for the pattern-coded macroblocks, one has to use the union of the coded-block pattern of the video and the masking process.

The above procedure for blending the mask in the compressed domain will result in a constant $\beta v(k,l)$ frame to be output by the MPEG-2 decoder. The proof is excluded here for brevity and is available in [11]. The presence of $\beta v(k,l)$ will cause the video to be obscured. Subscribers would be provided with an unmasking frame to view the video clearly. Unmasking is done in the spatial domain (i.e., after the decoding) as explained in Sect. 3.3.2. The masked video is then broadcast.

3.2 Mutual authentication and subscriber's concerns

For subscription-based, copyrighted-material broadcasting, mutual authentication is necessary to identify each party to the other and to develop a certain amount of trust. Especially when the broadcaster and the subscriber are widely separated geographically and are connected through wired or wireless networks, the designed protocols must succeed in unambiguously authenticating one to the other and must be robust against various attacks in the open environment. Even after authentication, it is necessary to protect each party against possible wrong doings by the other. Being copyright-protected material the customer's are concerned about the false implication of copyright violations by broadcasters. They are also concerned about the true ownership of the copyrighted material. Broadcasters on the other hand are concerned about copyright violations.

The interactive protocol that we present here is designed to provide the authentication, transact the payment, address the subscriber's concerns and, in the end, result in handing over the unmasking frame that carries watermark information to the subscriber. We now explain the interactive protocol in detail.

3.2.1 The interactive protocol

The protocol involves the broadcaster B, the receiver R_i and a watermark generation authority (WGA) known and trusted by both the receiver and the broadcaster. The necessity of WGA is explained in Sect. 4. We assume that the receiver R_i has obtained the digital certificate $DigCertR_i$ from issuer bank, and the broadcaster has acquired the digital certificate $DigCertB$ from the acquirer bank. These digital certificates contain the public key of the user, the identity of the user, the identity of the signing authority, the signing algorithm, and the period of validity of the certificate, and are signed by the certifying authority. The digital certificates vouch for the authenticity and integrity of the public keys of the user. We assume here that the signatures of certifying authorities can be verified by the broadcaster and receivers (either they know the certifying authorities public keys or use a chain of certificates to get the authentic public keys of the certifying authorities). We take the banks as the certifying authority of the digital certificates because they can stand as guarantors for the financial transactions and geographic identities of the receiver and broadcaster. The issuer bank and acquirer bank are assumed to be connected through a backend payment network such as VISA or Mastercard. The broadcaster also possesses a digital proof of ownership or distributorship for the digital material. This digital proof must be tied to the digital content it tries to prove.

Whenever a new receiver wants to view the unobscured broadcast, the receiver has to execute the following protocol, which is depicted in Fig. 3. Thus for every new transaction the following protocol is executed. The protocol consists of six messages.

The notation $A \rightarrow B : M$ means message M is transferred from A to B. We use a variation of Memon's protocol [23] for images to address the subscriber's concern. The protocol has the following messages transacted in the given sequence.

Message 1 $R_i \rightarrow B : \mathrm{SignR}_i[t_{Ri}, r_{Ri}, B, Rqst\_proof], DigCertR_i$

Message 2 $B \rightarrow R_i : \mathrm{SignB}[t_B, r_B, R_i, r_{Ri}, \alpha, Proof\_msg], DigCertB$

Message 3 $R_i \rightarrow WGA : \mathrm{SignR}_i[t_{RiWGA}, r_{RiWGA}, WGA, r_B, k_i, \alpha], DigCertR_i$

Message 4 $WGA \rightarrow R_i : \mathrm{SignWGA}[t_{\mathrm{WGA}}, r_{\mathrm{WGA}}, R_i, r_B, E_{k_i}[\alpha W_i(k,l)], \mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]], DigCertWGA$

Message 5 $R_i \rightarrow B : \mathrm{SignR}_i[r_B, k_i, E_{k_i}[\alpha W_i(k,l)], \mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]], OrderInfo, PaymentInfo], DigCertWGA$

Message 6 $B \rightarrow R_i : \mathrm{SignB}[E_{k_i}[v_i(k,l)], r_B]$

3.2.1.1 Protocol discussion. Message 1: The receiver R_i who wishes to join the broadcast initiates the protocol by sending Message 1 to broadcaster B. This message consists of a signed component and the digital certificate of R_i. The digital certificate $DigCertR_i$ contains the public key and identity of R_i, both of which can be recovered from the certificate by the broadcaster. The public key is then used to verify the signed component.

The signed component contains the following:

t_{Ri}	Timestamp containing a generation time and an expiration time.
r_{Ri}	Nonce to prevent replay attack. This has to be unique within the expiration time mentioned in the timestamp.
B	Specifies that the message is intended for B and not for anyone else.
$Rqst\_proof$	This is a message sent by the receiver R_i to the broadcaster asking to show the proof of ownership or distributorship of the digital content. Proof of ownership or distributorship is necessary to prove to the receivers that the digital content that is broadcast is legal and the broadcaster has the right to distribute it or owns it. This certificate may be obtained by the broadcaster from a central registration facility.
$\mathrm{SignR}_i[.]$	Signature done using R_i's private key.

By the end of the processing of Message 1, B will know the following:

(i) Identity of the receiver R_i.
(ii) R_i generated the message.
(iii) The message was intended for B.
(iv) The integrity and originality (not a replay) of the message.
(v) R_i is requesting proof of ownership or distributorship.

The broadcaster then constructs Message 2 and sends it to the receiver.

Message 2: Message 2 also consists of a signed portion and the digital certificate. The R_i extracts the identity and public key of B from the digital certificate $DigCertB$. The public key is then used to verify the signed portion.

The signed component contains the following:

t_B	Timestamp containing a generation time and an expiration time.

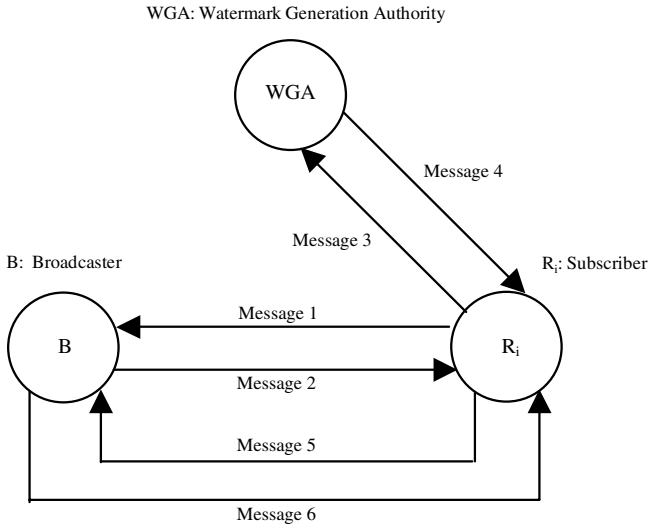

Fig. 3. Interactive protocol

r_B	Nonce to prevent replay attack. This has to be unique within the expiration time mentioned in the timestamp.
R_i	Specifies that the message is intended for R_i and not for anyone else.
r_{Ri}	Nonce that is included by B to validate the reply and to prove that the reply is for current request.
α	The scaling parameter used while blending.
$Proof\_msg$	The proof of ownership or distributorship.

>From the proof-of-ownership/distributorship field, R_i verifies the legality of the content. By the end of processing of Message 2, R_i is completely sure that B is a valid, authentic broadcaster and the material that is broadcast is legal, being rightfully owned or broadcast with permission.

Message 3: The receiver R_i then requests the watermark generation authority WGA create a permutation-variant watermark (i.e. $\sigma\{\text{watermark}\} \neq \text{watermark}$ where $\sigma\{.\}$ is the permutation operator) by sending Message 3. Thus for a permutation-variant watermark, the permuted watermark is different from the original watermark. The need for permutation variance is explained in Sect. 4, along with the need for the WGA. The message construct proves that the message is from R_i. The WGA verifies the signed component using the public key obtained from $DigCertR_i$. t_{RiWGA} and r_{RiWGA} are the timestamp and the nonce respectively. They are used to avoid the replays and delayed messages. WGA specifies that the message is intended for WGA. The nonce r_B is used to indicate that the message is for the current transaction. The key k_i is the public key in the Niederreiter public-key cryptographic system. Receiver R_i owns the corresponding private key. The Niederreiter cryptographic system is privacy homomorphic with respect to addition, and its requirement is presented later in Sect. 4. The factor α is the scaling parameter used in Eq. (1). This needs to be sent to the WGA to appropriately scale the generated watermark.

After verifying the signed component using the public key obtained from $DigCertR_i$, the WGA creates a robust invisible watermark $W_i(k,l)$, which is permutation variant (i.e. $\sigma\{W_i(k,l)\} \neq W_i(k,l)$). This watermark is for addressing the subscriber's false-implication concern. The watermark is then scaled by α and encrypted using public key k_i and Niederreiter's public-key cryptographic system. The WGA also generates a signed version of $E_{k_i}[\alpha W_i(k,l)]$ (the scaled encrypted watermark). In order to verify the signature, $DigCertWGA$ contains the corresponding public key.

Message 4: Message 4 is send to R_i by WGA. This message construct would enable the R_i to be sure that the received message is from WGA and is also not a delayed or replayed message. t_{WGA} is the timestamp, and r_{WGA} and r_B are nonces. The nonce r_B is to indicate that the message is for the current transaction with the broadcaster. $E_{k_i}[\alpha W_i(k,l)]$ is the scaled watermark, encrypted with public key k_i. The encryption algorithm used is the Niederreiter scheme. $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$ is the signed version of the $E_{k_i}[\alpha W_i(k,l)]$.

On receiving Message 4, R_i verifies $E_{k_i}[\alpha W_i(k,l)]$ against its signed version. Then Message 5 is constructed and transmitted to B.

Message 5: The nonce r_B is to indicate the message is for the current transaction. The key k_i is the public key in the Niederreiter scheme. The signed component of the message contains two further pieces of information:

$OrderInfo$	Order information, which is a description of the digital content and proof of ownership or distributorship.
$PaymentInfo$	Payment information (credit-card details etc.).

The broadcaster first verifies $E_{k_i}[\alpha W_i(k,l)]$ against $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$. The public key of WGA is obtained from $DigCertWGA$. Once verified, the broadcaster sends payment information to the acquirer bank and obtains the payment authorization. Once authorization is obtained the broadcaster constructs $E_{k_i}[v_i(k,l)]$ (encrypted unmasking frame). The construction of $E_{k_i}[v_i(k,l)]$ is explained in Sect. 3.3.1.

Message 6: The nonce r_B indicates that the message is for current transaction. The term $[v_i(k,l)]$ is the unmasking frame for the receiver R_i, and $E_{k_i}[v_i(k,l)]$ is the unmasking frame encrypted using public key k_i. The signing is intended to preserve the authenticity and integrity of the message.

At the end of the protocol, if no complaints from the receiver R_i arise, the broadcaster requests payment from the acquirer bank. Messages 1 and 2 are intended for mutual authentication and proving the ownership or distributorship. Messages 3, 4, 5 and 6 are mainly for addressing the subscriber's concern. Message 6 is partly for copyright-violation detection.

3.3 Copyright-violation detection and subscriber's concerns

Digital watermarks [7,13] are information embedded in the original data with the intention of addressing copyright and related issues, such as copyright-violation deterrence, copyright-violation detection, data integrity (data tamper proofing) [18] and copy protection. The embedded watermarks may be either visible [16,22] or invisible (audible or inaudible in case of audio) [6,8,9,32,28]. They cannot be removed or detected by anyone other than the creator.

In our scheme, we make use of two robust invisible watermarks to provide for copyright-violation detection and subscriber's false-implication concerns. The unmasking frame carries both these watermarks to the subscriber set-top box, and they get embedded in the video during the unmasking process. Next, we explain the unmasking-frame construction and the unmasking process in detail.

3.3.1 Unmasking frame construction

After receiving Message 5 the broadcaster verifies the validity of the $E_{k_i}[\alpha W_i(k,l)]$ using $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$. Then the identity of the receiver (from $DigCertR_i$), nonce r_B and the present time information are used to produce a robust invisible watermark $W_{bi}(k,l)$ specifically for receiver R_i for this transaction. The watermark $W_{bi}(k,l)$ is meant to solve the broadcaster's concern of the copyright protection. The robust invisible watermark $W_{bi}(k,l)$ should be kept secret by the broadcaster. The broadcaster then creates an unmasking frame $v_{bi}(k,l)$ for the subscriber R_i using the following expression:

$$\forall k,l: \quad v_{bi}(k,l) = \beta v(k,l) - \alpha W_{bi}(k,l). \tag{6}$$

The frame $v_{bi}(k,l)$ is then encrypted using the public key k_i (Niederreiter's scheme) of the receiver to obtain $E_{k_i}[v_{bi}(k,l)]$. A random permutation σ is generated and is used to permute the elements of the encrypted watermark $E_{k_i}[\alpha W_i(k,l)]$ received from R_i. Let $\sigma\{.\}$ be the permutation operator. Therefore,

$$\sigma\{E_{k_i}[\alpha\, W_i(k,l)]\} = E_{k_i}[\sigma\{\alpha\ W_i(k,l)\}]. \tag{7}$$

This is true as $E_{k_i}[\alpha W_i(k,l)]$ is of the form $\{E_{k_i}[\alpha W_i(b_1)], E_{k_i}[\alpha W_i(b_2)], E_{k_i}[\alpha W_i(b_3)], \ldots\}$, where $b_1, b_2, b_3, \ldots$ are blocks consisting of pixels. In our case the blocks consists of 32 bits as discussed in Sect. 3.6. The encryption is applied to each block, and the blocks are permuted. Therefore, we can commute permutation and encryption.

Let $\sigma\{\alpha W_i(k,l)\} = W_i^{\sigma}(k,l)$. The broadcaster then subtracts the encrypted permuted watermark $E_{k_i}[W_i^{\sigma}(k,l)]$ from $E_{k_i}[v_{bi}(k,l)]$ to obtain $E_{k_i}[v_i(k,l)]$:

$$\forall k,l:\ E_{k_i}[v_i(k,l)] = E_{k_i}[v_{bi}(k,l)] - E_{k_i}[W_i^{\sigma}(k,l)]; \tag{8}$$

$$\forall k,l:\ E_{k_i}[v_i(k,l)] = E_{k_i}[v_{bi}(k,l) - W_i^{\sigma}(k,l)] \tag{9}$$

Here we assume the encryption system to be privacy homomorphic with respect to addition and subtraction. Privacy homomorphism with respect to addition and subtraction would mean that $E_{k_i}[X_1] \pm E_{k_i}[X_2] = E_{k_i}[X_1 \pm X_2]$. An example of such a system is Niederreiter's public-key cryptographic system based on coding theory [25]. We briefly explain the Niederreiter system in Sect. 3.6. Message 6 containing $E_{k_i}[v_i(k,l)]$ is then generated and transmitted to receiver R_i. The unmasking frame $v_i(k,l)$ is the access-control data for R_i. The broadcaster then stores $DigCertR_i$, obtained from the receiver, $E_{k_i}[\alpha W_i(k,l)]$, $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$, $W_{bi}(k,l)$ and σ in a look-up table called *SubTable*.

It is in the interest of the broadcaster that the unmasking frame construction is performed as defined in Eqs. (7) to (10). The proper generation of $v_{bi}(k,l)$ using Eq. (7) causes the unmasked watermarked video to be watermarked with $W_{bi}(k,l)$ when unmasked by the subscriber R_i. The watermark $W_{bi}(k,l)$ is meant for tracing the legal recipient of the video. Whereas Eq. (9) causes the unmasked watermarked video to be watermarked with $\sigma\{W_i(k,l)\}$ when the unmasking is done by the subscriber R_i. The watermark $\sigma\{W_i(k,l)\}$ is meant for addressing the subscriber's concern. Since processing of Eq. (9) is done in the encrypted domain, the broadcaster does not get to know what $W_i(k,l)$ is. Therefore, the broadcaster cannot create another video with watermark $\sigma\{W_i(k,l)\}$. At the same time, the permutation function is known only to the broadcaster, therefore the receiver does not know what $\sigma\{W_i(k,l)\}$ is. And therefore the receiver cannot remove $\sigma\{W_i(k,l)\}$ from the unmasked watermarked video.

3.3.2 Unmasking process

The unmasking is carried out in the spatial domain for compressed-domain and spatial-domain processing. To view the channel without any obscurity the receiver R_i first extracts $E_{k_i}[v_i(k,l)]$ from Message 6 and then decrypts it using the R_i's private key k_i' (Niederreiter's scheme) to obtain $v_i(k,l)$:

$$D_{k_i'}[E_{k_i}[v_i(k,l)]] = v_i(k,l) = v_{bi}(k,l) - W_i^{\sigma}(k,l). \tag{10}$$

The frame $v_i(k,l)$ is then used to unmask the obscured video. Equation (12) defines the unmasking process:

$$\begin{aligned}
x_n^{w_i}(k,l) &= (x_n^m(k,l) - v_i(k,l))\ (1/\alpha) \quad \forall k,l \qquad (11)\\
x_n^{w_i}(k,l) &= (x_n^m(k,l) - v_{bi}(k,l) + W_i^{\sigma}(k,l))\ (1/\alpha)\\
x_n^{w_i}(k,l) &= (x_n^m(k,l) - \beta\, v(k,l) + \alpha\, W_{bi}(k,l)\\
&\quad +\sigma\{\alpha\, W_i(k,l)\})\ (1/\alpha)\\
x_n^{w_i}(k,l) &= \begin{cases} (\alpha\, x_n(k,l) + \beta\, v(k,l)\\ -\beta\, v(k,l) + \alpha\, W_{bi}(k,l)\\ +\sigma\{\alpha\, W_i(k,l)\})(1/\alpha) \end{cases}\\
x_n^{w_i}(k,l) &= x_n(k,l) + W_{bi}(k,l) + \sigma\{W_i(k,l)\} \ \ \forall k,l \qquad (12)
\end{aligned}$$

where $x_n^{w_i}(k,l)$ is the watermarked frame for receiver R_i. Notice that $x_n^{w_i}(k,l)$ contains the robust invisible watermark $W_{bi}(k,l)$ and the robust invisible watermark $\sigma\{W_i(k,l)\}$ (the permuted $W_i(k,l)$). For the compressed-domain case, after unmasking, Eq. (13) contains $x_n'(k,l)$ instead of $x_n(k,l)$ to reflect the fact that MPEG-2 is a lossy compression technique.

The subscriber has to carry out the unmasking to view the video unobscured and in the process the unmasked video gets watermarked with $W_{bi}(k,l)$ and $\sigma\{W_i(k,l)\}$. The watermark $W_{bi}(k,l)$ is for copyright-violation detection while $\sigma\{W_i(k,l)\}$ is for addressing subscriber's concerns.

3.4 Watermark construction

We construct a watermark frame [12] of dimension K pixels by L pixels, same as that of the video frames. We consider the watermark frame as a one-dimensional signal acquired by raster-scanning (scanning left to right and then top to bottom). Assume that the information to be embedded consists of bits having values $\{-1, 1\}$. Let us create a sequence a_j out of it (the watermark information to be embedded).

Let

$$a_j, \quad a_j \in \{-1, 1\} \quad j = 0, 1, \ldots, N, \tag{13}$$

be a sequence of bits, which is then spread using the chip rate C_r to obtain the spread sequence b_i. The C_r and N are selected in such a way that $C_r \times N = K \times L$, the frame dimension.

$$\forall j: \quad b_i = a_j, \quad jC_r \leq i < (j+1)C_r. \tag{14}$$

The spreading provides redundancy and improves the robustness to geometrical attacks such as cropping. The spread sequence is then multiplied with a pseudorandom noise sequence p_i, where $p_i \in \{-1, 1\}$. It is then amplified by a scaling factor κ (a positive number selected in such a way that the watermark still remains invisible in the watermarked frames and is also detectable) to get the watermark.

$$\forall i: \quad w_i = \kappa\, b_i p_i. \tag{15}$$

The watermark w_i could be arranged as a frame (dimension $K \times L$), which is the watermark frame.

3.5 Watermark detection

Detection of the hidden information a_j is done by employing the correlation receiver [12]. The correlation receiver does not require the original unwatermarked video signal for the detection. To detect a_j, we multiply the watermarked video x_i^w by the same pseudorandom noise sequence p_i that was used for watermark construction followed by a summation over the window of the embedded information, yielding the correlation s_j. $\mathrm{sign}(s_j)$ is a_j.

$$\begin{aligned} s_j &= \sum_{i=jC_r}^{(j+1)C_r-1} p_i x_i^w = \sum_{i=jC_r}^{(j+1)C_r-1} p_i x_i + \sum_{i=jC_r}^{(j+1)C_r-1} p_i w_i \\ &= \sum_{i=jC_r}^{(j+1)C_r-1} p_i x_i + \sum_{i=jC_r}^{(j+1)C_r-1} p_i^2 \kappa\, b_i, \end{aligned} \tag{16}$$

where x_i is the original unwatermarked video. The first term in Eq. (17) is zero if p_i and x_i are uncorrelated. However, this is not always the case for real data. So, to obtain a better result, we first prefilter the watermarked video x_i^w and remove most of the unwatermarked video content. But if we have the original unwatermarked video, we just need to subtract the original unwatermarked video from the watermarked video x_j^w. Assuming that the first term in Eq. (17) is almost zero,

$$\begin{aligned} s_j &= \sum_{i=jC_r}^{(j+1)C_r-1} p_i^2 \kappa\, b_i = \sum_{i=jC_r}^{(j+1)C_r-1} p_i^2 \kappa\, a_j \\ &= \kappa\, a_j \sum_{i=jC_r}^{(j+1)C_r-1} p_i^2 = a_j \kappa\, \sigma_p^2 \end{aligned} \tag{17}$$

Since κ and σ_p^2 are positive, we have

$$\mathrm{sign}(s_j) = \mathrm{sign}(a_j\ \kappa\ \sigma_p^2) = a_j \tag{18}$$

Next, we discuss the encryption system, which supports privacy homomorphism with respect to addition and subtraction.

3.6 Public-key systems based on codes

The decoding problem for general linear codes is NP-complete and is used in the design of the public-key cryptographic systems based on coding theory. The public-key cryptographic system based on codes introduced by Niederreiter is privacy homomorphic with respect to addition and subtraction. Privacy homomorphism with respect to addition and subtraction means that $E_{k_i}[X_1] \pm E_{k_i}[X_2] = E_{k_i}[X_1 \pm X_2]$.

Niederreiter's system [25] uses a linear $[n, k, d]$ code C over F_q, the finite field of order q, where n is the length of C, k is the dimension of C and d is the minimum hamming distance. The hamming weight $\omega(x)$ is the number of non-zero coordinates of x.

Private key: The private key consists of three matrices H, M and P where H is an $(n-k) \times n$ parity-check matrix of C, M is an arbitrary $(n-k) \times (n-k)$ invertible matrix, and P is an arbitrary $n \times n$ permutation matrix.

Public key: The public key is an $(n-k) \times n$ matrix H' defined by $H' = MHP$.

Encryption: The admissible plain texts are column vectors with weight $\omega(x) \leq t := \lfloor (d-1)/2 \rfloor$. Given a plain text x, which is a column vector), the corresponding cipher is $y = H'x$.

Decryption: Given a cipher y, a column vector, first compute $y' = M^{-1}y = HPx$. Let $x' = Px$; x' can be viewed as an error vector. Then the decoding algorithm of C is applied to the syndrome $y' = Hx'$ to yield the error vector x'. The plain text x is recovered by $x = P^{-1}x'$.

We can see from the encryption equation $y = H'x$ that $E_{k_i}[x_1] \pm E_{k_i}[x_2] = H'x_1 \pm H'x_2 = H'(x_1 \pm x_2) = E_{k_i}[x_1 \pm x_2]$. Therefore, the scheme possesses privacy homomorphism with respect to addition and subtraction. The McEliece public-key cryptography system based on codes can also be used in our protocol but only with certain conditions to provide more security. Encryption in Niederreiter's cryptography system is claimed to be 10 times faster than McEliece cryptography system and 48 times faster than the RSA cryptography system with comparable security levels. But the code-based systems require large key size for the same security compared to that of RSA [24]. The suggested parameters are $n = 1024$, $t = 37$, $k = 654$. This will require that the block size for encryption to be 4 pixels (32 bits). Since we are dealing with video data the key size is relatively small compared to the video data size. Also, in the real-time video-broadcasting environment, the encryption time is more important than the key size.

4 Analysis

In this section, we analyze the security of the proposed scheme, which consists of mask blending, the interactive watermarking protocol and the watermarks.

4.1 Security of mask blending

Given a composite masked video frame, finding the original video frame and the mask frame is the problem. There is only

one exact solution. However, there may be many acceptable solutions, as there are many video frames that appear similar to the original exact frame. We make use of the just-noticeable-distortion (JND) value definition by Chou and Li [5] for calculating the acceptable number of solutions. We only consider the visibility threshold due to the background luminance.

Therefore, the JND for each pixel value p is given by

$$f(p) = \begin{cases} 17\left(1 - (p/127)^{1/2}\right) + 3 & p \leq 127 \\ (3/128)\,(p - 127) + 3 & p > 127 \end{cases} \tag{19}$$

where $p \in [0..255]$.

Let the original image frame pixel values be Gaussian with mean μ_x and variance σ_x^2 and mask image frame pixel values be Gaussian with mean μ_v and variance σ_v^2. Then the masked video frame, which is obtained by blending the mask image frame into the video frame as given in Eq. (1), would have the mean $\mu_m = \alpha\mu_x + \beta\mu_v$ and the variance $\sigma_m^2 = (\alpha\sigma_x)^2 + (\beta\sigma_v)^2$.

Let

N be the total number of pixels in a frame of dimension $K \times L$, i.e., $N = K \times L$.
N_p be the number of pixels having pixel value p in the masked video frame.

Assuming Gaussian with mean μ_m and variance σ_m^2, the probability of a pixel having value p is

$$P(p) = \int_{p-0.5}^{p+0.5} \frac{1}{\sqrt{2\pi\sigma_m^2}} \exp\left\{-(p - \mu_m)^2 \Big/ 2\sigma_m^2\right\} dp. \tag{20}$$

Therefore, the number of pixels having value p in an ensemble of N pixels is

$$N_p = \text{round}\left[N \times P(p)\right] \quad \text{s.t.} \quad N = \sum_{p=0}^{255} N_p. \tag{21}$$

Given a masked video frame pixel value p, the number of compositing solutions is $(p+1)$. Therefore, the total number of compositing solutions for entire masked video frame of dimension $K \times L$ is

$$\prod_{p=0}^{255} (p+1)^{N_p}. \tag{22}$$

Since there is only one exact solution, the probability of finding the exact solution is

$$\frac{1}{\prod_{p=0}^{255} (p+1)^{N_p}}. \tag{23}$$

The total number of acceptable solutions is

$$\prod_{p=0}^{255} (f(p))^{N_p}. \tag{24}$$

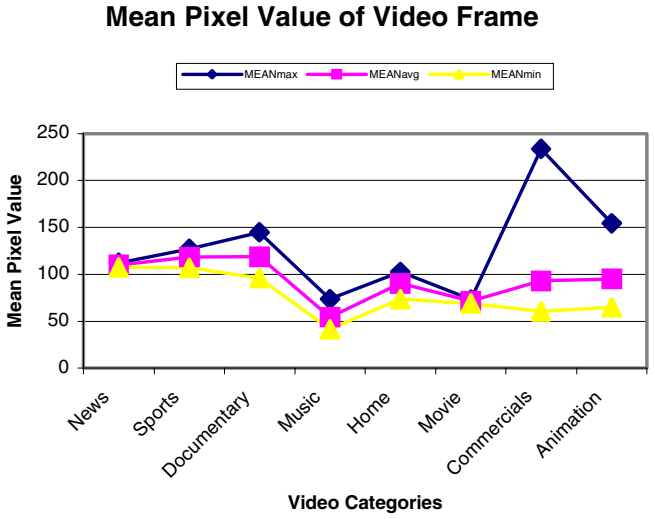

Fig. 4. Plot of mean pixel value

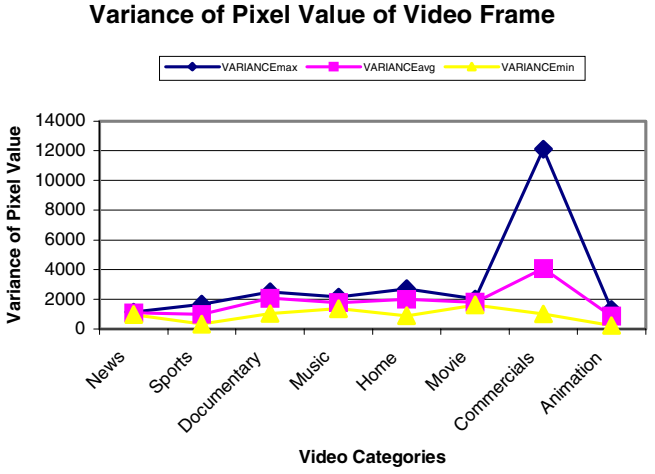

Fig. 5. Plot of variance of pixel value

Therefore, the probability of finding an acceptable solution is

$$\frac{\prod_{p=0}^{255} (f(p))^{N_p}}{\prod_{p=0}^{255} (p+1)^{N_p}}. \tag{25}$$

The luminance (Y) frame mean and variance for different MPEG-7 video categories have been experimentally determined. The MPEG-7 video set consists of 10 categories with 30 items. But our test covers only 8 categories that have 27 items of 12:50:47 hours' duration and 1,210,642 number of frames. The maximum of frame means ($MEANmax$), average of frame means ($MEANavg$) and minimum of frame means ($MEANmin$) for different video categories are plotted in Fig. 4. The average of frame means over all the video categories is computed to be 93.966. The maximum of frame variances ($VARIANCEmax$), average of frame variances (VARIANCEavg) and minimum of frame variances ($VARIANCEmin$) for different video categories are plotted in Fig. 5. The average of frame variances over all the video categories is computed to be 1826.263.

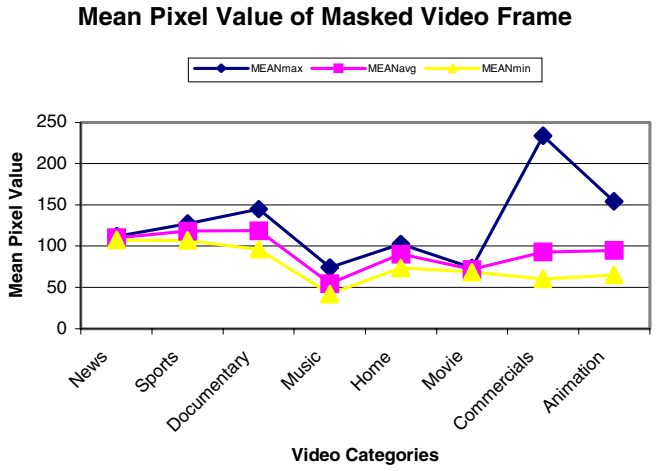

Fig. 6. Mean pixel value of masked video frame

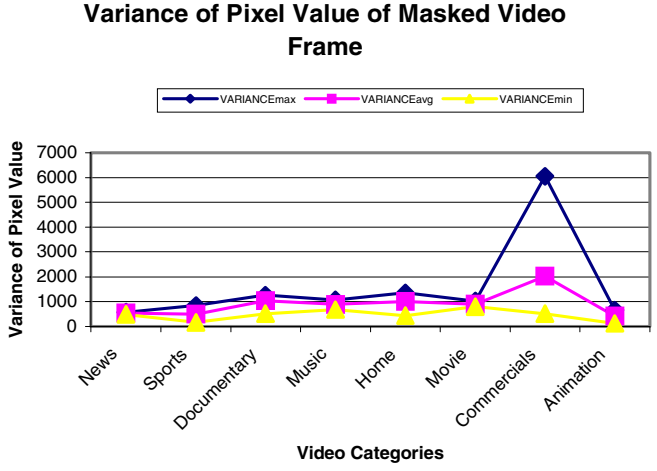

Fig. 7. Variance of pixel value of masked video frame

The masked video frame mean and variance are computed from the experimentally determined frame mean and variance assuming both the original video frame and the mask frame belongs to the same category and $\alpha = \beta = 0.5$. These are plotted in Figs. 6 and 7. For the computations, we use the expressions $\mu_m = \alpha\mu_x + \beta\mu_v$ and $\sigma_m^2 = (\alpha\sigma_x)^2 + (\beta\sigma_v)^2$ for mean and variance respectively, where μ_x is the original video frame mean, μ_v is the mask frame mean, σ_x^2 the original video frame variance and σ_v^2 the mask frame variance. The use of $\alpha = \beta = 0.5$ and the assumption that the mask frame is of same category as that of the video frame results in frame means of the original video and those of the masked video being the same, but the variances differ. The average of frame means and variances over all the video categories is computed to be 93.966 and 913.132 respectively for the masked video.

The computed masked video frame mean and variance are then used to find the total number of compositing solutions and total number of acceptable solutions, and hence the probability of an acceptable solution for various categories of video. The total number of compositing solutions and the total number of acceptable solutions for the video categories are plotted in Figs. 8 and 9. The plotted values on the graph are logarithmic to base 10 as the numbers are extremely large. $NSOLmax$, $NSOLavg$ and $NSOLmin$ are the total numbers of compositing solutions computed using $(MEANmax, VARIANCEmax)$, $(MEANavg, VARIANCEavg)$ and $(MEANmin, VARIANCEmin)$ as parameters for Gaussian distribution respectively. $NACPTmax$, $NACPTavg$ and $NACPTmin$ are the total number of acceptable solutions computed using $(MEANmax, VARIANCEmax)$, $(MEANavg, VARIANCEavg)$ and $(MEANmin, VARIANCEmin)$ as parameters for Gaussian distribution respectively. It can be observed from the plot of the number of acceptable solutions that $NACPTmin$ is larger than $NACPTavg$ and $NACPTmax$. This is due to the interplay of the functions $f(p)$ and N_p. The function $f(p)$ consists of a nonlinear part and a linear part with a discontinuity at 127. The function $f(p)$ gives us the tolerance allowed to each pixel. From Eq. (20) for $f(p)$ it can be seen that the pixels values near zero enjoy more tolerance. A Gaussian distribution with a lower mean (assuming constant variance) points to more pixels having zero or near to zero values, resulting in a larger number of acceptable solutions.

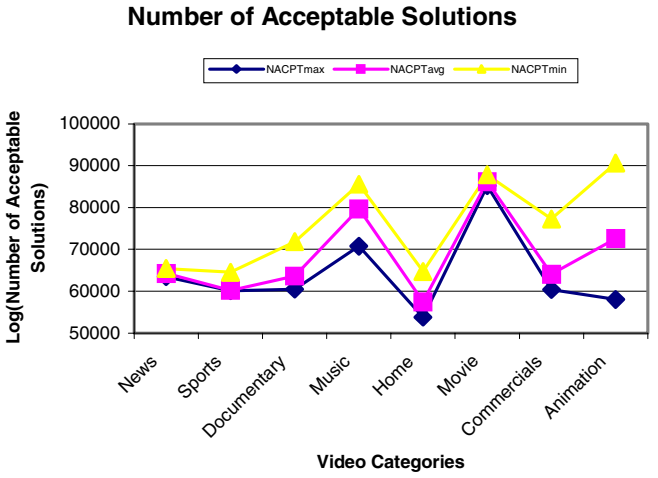

Fig. 8. Plot of number of compositing solutions

The probability of finding an acceptable solution is plotted in Fig. 10. $Prob\_Acpt\_max$, $Prob\_Acpt\_avg$ and $Prob\_Acpt\_min$ are computed using $(NSOLmax, NACPTmax)$, $(NSOLavg, NACPTavg)$ and $(NSOLmin, NACPTmin)$ respectively. The plotted values on the graph are logarithmic to base 10. The plots show that the probability of finding an acceptable solution is too low.

Using the average of frame means 93.966 and variances 913.132 over all the video categories for the masked video the total number of compositing and acceptable solutions are 10^{197635}, 10^{73845} and the probability of finding an acceptable solution is $1/10^{123790}$.

Therefore, it is hard to break this masking through brute-force techniques, as it requires an enormous amount of computing power coupled with human interaction to identify the correct or acceptable original video frame.

However, the broadcaster must make sure that the mask frame should be neither made known (through either applying the mask to a blank frame or some other means) nor easily guessed by the receivers (subscribers and non-subscribers). To make the proposed method more robust against attacks, one needs to design a porous mask and robust invisible watermark,

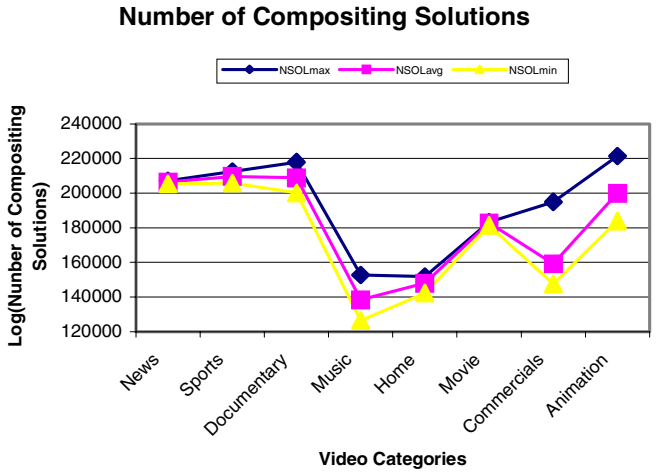

Fig. 9. Plot of number of acceptable solutions

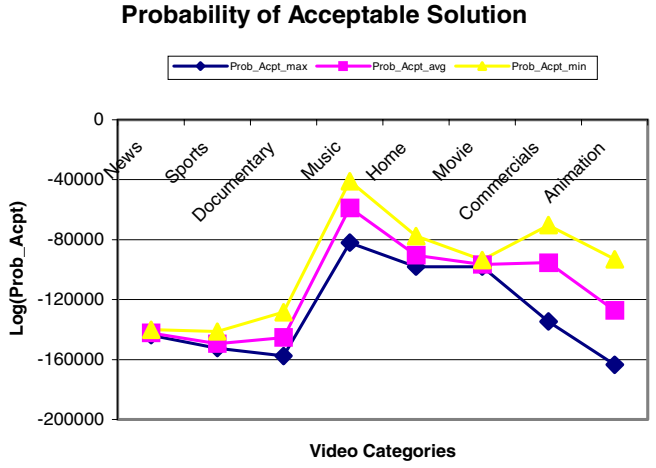

Fig. 10. Probability of finding an acceptable solution

which means we mask and watermark only some random pixel locations (substantially large in number in order to have the opacity effect) or change the mask often or use multiple masks and use them interchangeably.

4.2 Security of the protocol

The parameters, timestamps and nonces are required for preventing the replay attack. This requires the nonce to be unique within the expiration time mentioned in the timestamp. The nonces are also used to track the messages belonging to a particular transaction. Since Messages 1 through 6 are just signed messages, the identity of the recipient has to be included in the signed messages in order to indicate to whom this message is intended. $Proof\_msg$ is intended to help in verifying that the broadcaster either owns or has permission to broadcast the content. The digital certificates contain the public key of the user, the identity of the user, the identity of the signing authority, the signing algorithm, and the period of validity of the certificate. The certifying authority signs all the certificates. The digital certificates vouch for the authenticity and integrity of the public keys of the user. The public key and the identity of the user can be extracted from it. The term $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$ is needed to authenticate $E_{k_i}[\alpha W_i(k,l)]$. The order information ($OrderInfo$) is needed to indicate that the payment ($PaymentInfo$) is meant for this order.

The security of the protocol lies in the private key k_i' (Niederreiter's scheme) of the subscriber and the private keys of the WGA, subscriber and broadcaster used while signing. In theory, one could use any secure public-key encryption algorithm that exhibits privacy homomorphism with respect to addition, instead of Niederreiter's scheme. Whichever public-key cryptography system is selected, it should be hard for the opponent to break the encryption scheme. The WGA should be trusted by broadcaster as well as the subscribers, and must create a permutation-variant watermark $W_i(k,l)$ (i.e., $\sigma\{W_i(k,l)\} \neq W_i(k,l)$, where $\sigma\{.\}$ is the permutation operator) for the subscriber. The watermark $W_i(k,l)$ is handled in the encrypted domain, and only the subscriber gets to know the watermark $W_i(k,l)$, which the subscriber must keep secret. The broadcaster must keep the permutation σ and watermark $W_{bi}(k,l)$ secret.

The mutual-authentication scheme used is similar to the X.509 standard, whose security with respect to authentication, replay attacks and other attacks has been widely studied [29]. Essentially, the security of the authentication lies in the private keys used for signing the Messages 1 through 6. The respective private keys are kept secret by each party (B, R_i and WGA). The digital certificate of each party contains the corresponding public key for verification of the signed messages. For this purpose, one may use the RSA public-key cryptography system.

Suppose an eavesdropper observes the communication (Messages 1 through 6). Since the messages are just signed anyone including the eavesdropper can verify the signature and obtain the message contents. However, the eavesdropper will not be able to create the signed message from the obtained message content, as the private key used for signing the message contents will not be known. However, the eavesdropper would be able to get $E_{k_i}[v_i(k,l)]$, $E_{k_i}[\alpha W_i(k,l)]$, $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$ and k_i (the public key used in the Niederreiter scheme) from the signed messages after applying the verification algorithm. The term $E_{k_i}[v_i(k,l)]$ is of no use to eavesdropper as it is not able to be decrypted, since the corresponding private key k_i' is not known. Suppose the eavesdropper uses the eavesdropped message $E_{k_i}[\alpha W_i(k,l)]$, $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$ and key k_i for a transaction with the broadcaster. At the end of the protocol, the last message would contain $E_{k_i}[v_{i\_\mathrm{new}}(k,l)]$. Once again the eavesdropper cannot decrypt this since the corresponding private key k_i' will not be known. Therefore, the security of the protocol rests on the private keys of the WGA, subscriber and broadcaster used while signing, as well as the private key k_i' of the subscriber.

4.3 Robustness of the watermark

The security of the watermarks has been well studied [13,14]. We, in our scheme, make use of the robust invisible watermark proposed by Hartung and Girod [12], which is a watermark based on spread spectrum. Therefore, the security of the watermark in our scheme is similar to that of their scheme. There are many remedies and counter attacks presented in [14] to

make the spread-spectrum watermarks more resistant against attacks.

Suppose n subscribers collude to make an unwatermarked video by averaging the corresponding frames of each subscriber's video, or they collude by averaging the unmasking frames to make an unmasking frame that does not carry watermark (more precisely, the inverse watermark) information. Boneh and Shaw [14,35] have shown, how to construct watermark signals to defeat this kind of averaging collusion attacks. In the event of collusion, Boneh and Shaw's schemes would point out the colluding parties. There is another kind of collusion where the colluding parties can assemble a new video by randomly selecting frames from each of their watermarked videos. But since watermarking is done frame-wise, this would reveal the colluding parties. The subscribers can possibly assemble an unmasking frame strip by strip, by switching between different unmasking frames, or can create an unmasked video strip by strip from each corresponding frame, by switching between different unmasked watermarked video. In either of these cases, to defeat this kind of collusion, the watermark signal/information bits have to be pseudorandomly distributed to the pixels by using a pseudorandom noise sequence p_i [14], as described in Sect. 3.4.

The copy attack copies the watermark present in watermarked data to other data where there is no watermark [19]. The attack consists of the following: first it estimates the watermark in the watermarked data, then the estimated watermark is adapted and inserted into the data that does not contain a watermark. While this attack is mainly intended for defeating the authentication or identification purpose of watermarking, it makes use of estimation techniques that assumes at least knowledge of the statistics of the watermark [19]. The watermark creators, the broadcaster and WGA should keep the statistics of the watermarks a secret. In order to make the estimation of watermark using Wiener estimator more difficult, the power spectrum of watermark should be a scaled version of the signal power spectrum [14].

While the inversion or ambiguity attacks are mainly intended for false ownership claims of watermarked data by producing a guessed false watermark and derived false original data, it causes confusion about the real owner of the data. Non-invertible watermarks can be designed to defeat this attack [14]. One of the ways of designing non-invertible watermarks is by using cryptographically secure timestamps provided by trusted third parties and encoded in the watermark [33]. Another way is by making the watermark dependent on the original data in a one-way fashion (for example, by using a hash function) [8,27,33].

Thus, spread-spectrum watermarks, such as Hartung and Girod's [12], can be designed to be robust against attacks. However, our proposed scheme supports the use of any additive robust invisible watermark. Therefore, the security depends on the robustness of the selected watermarking technique.

5 Discussion

We now discuss in detail some of the salient aspects of our scheme.

Copyright-violation detection. Suppose a legal recipient makes multiple copies of the unmasked watermarked $x_n^{w_i}(k,l)$ or the unmasking frame $v_i(k,l)$ and redistributes them to non-subscribers. The broadcaster can identify the subscriber who has redistributed the video by detecting the watermark $W_{bi}(k,l)$ present in the unauthorized copy found with the non-subscriber. For this purpose, the broadcaster selects one by one the watermarks already created, $W_{bi}(k,l)$'s, from the look-up table called *SubTable* and then correlates them with the copy found from the non-subscriber. The highest correlation value with a certain minimum threshold value is used to identify the watermark $W_{bi}(k,l)$ present in the copy of the video. If the correlation value is smaller than the minimum threshold, we declare that the watermark is not found. Once the watermark $W_{bi}(k,l)$ is identified, it could be used to obtain from *SubTable* the identity of the subscriber R_{i} who is the legal recipient.

The broadcaster can then initiate necessary legal steps and prove to the judge the existence of $W_{bi}(k,l)$ in the unauthorized copy. However, the existence of $W_{bi}(k,l)$ is not enough to prove the culpability of the subscriber. This is because the broadcaster has full control over $W_{bi}(k,l)$'s generation and insertion. The broadcaster can create, knowingly or unknowingly, another video with the same watermark. Therefore, the subscriber can dispute that the illegal copy has originated from the broadcaster. In the event of a dispute, the judge has to be also convinced of the existence of $\sigma\{W_i(k,l)\}$ in the unauthorized copy.

Dispute resolution: The broadcaster takes the *SubTable* entries for the subscriber R_i, $E_{k_i}[\alpha W_i(k,l)]$, $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$ and σ to the judge. The judge first verifies the validity of $E_{k_i}[\alpha W_i(k,l)]$ using $\mathrm{SignWGA}[E_{k_i}[\alpha W_i(k,l)]]$. The judge then proceeds to extract the watermark $W_i(k,l)$ from the message $E_{k_i}[\alpha W_i(k,l)]$. This can be done in two ways. First, the judge can request the corresponding private key k_i' from the subscriber and then decrypts and scales down by α $E_{k_i}[\alpha W_i(k,l)]$ to obtain $W_i(k,l)$. The judge then verifies the integrity of k_i' by inverse processing the watermark $W_i(k,l)$, which yields $E_{k_i}[\alpha W_i(k,l)]$ and then compares it with the *SubTable* entry $E_{k_i}[\alpha W_i(k,l)]$. Since k_i' is a private key, revealing it even to a judge may not be acceptable. Therefore, alternately the judge asks the subscriber to decrypt and scale down by α the *SubTable* entry $E_{k_i}[\alpha W_i(k,l)]$ and so obtain $W_i(k,l)$. The judge then inverse processes the watermark $W_i(k,l)$, which yields $E_{k_i}[\alpha W_i(k,l)]$ and compares it with the *SubTable* entry $E_{k_i}[\alpha W_i(k,l)]$. If they are equal, the judge takes the $W_i(k,l)$ as the valid one.

The judge then proceeds to construct the permuted watermark $\sigma\{W_i(k,l)\}$ and verifies the existence of the same watermark in the unauthorized copy. If the existence is proven then the judge declares that the subscriber has violated the copyright.

Subscriber's rights protection. Suppose the subscriber is innocent. The concern of such subscribers is that they be protected from being framed by the broadcasters. The broadcaster can successfully frame an innocent subscriber only when the broadcaster is able to make an illegal copy with $\sigma\{W_i(k,l)\}$. But, it is extremely difficult for the broadcaster to create another copy of the video with the watermark $\sigma\{W_i(k,l)\}$ as the broadcaster knows only the permutation function $\sigma\{.\}$ but not

$W_i(k,l)$. The broadcaster has access only to $E_{k_i}[\alpha W_i(k,l)]$, the encrypted, scaled version of $W_i(k,l)$. The public-key encryption used is assumed to be practically secure. Therefore, finding out $W_i(k,l)$ from $E_{k_i}[\alpha W_i(k,l)]$ is hard.

Need for the WGA. The WGA is necessary for the following reason: if the subscriber R_i were to generate the watermark $W_i(k,l)$, it may be possible to generate a watermark approximately invariant to permutation operation $\sigma\{.\}$ and hence possible remove the $\sigma\{W_i(k,l)\}$ from the unmasked watermarked video containing the watermark $\sigma\{W_i(k,l)\}$.

Need for privacy homomorphism with respect to addition. A public-key cryptography system with privacy homomorphism with respect to addition and subtraction will provide us with more control over the watermarking, with additive watermarking process in the encrypted domain, than with privacy homomorphism with respect to multiplication. It allows for easier and flexible modulation of the watermark signal over the video signal.

Other advantages. As far as the digital-video broadcasting pay channel is concerned, the subscriber can join the broadcast any time the subscriber wishes and remain in the broadcast as long as the mask remains valid (the mask could be changed periodically at certain time intervals for better security). Therefore, the method supports dynamic join and leave. Since the operations performed are simple point-by-point additions of frames for masking and unmasking, the method is less complex in terms of implementation. The masked video frames are created only once for any video, but the unmasking frames and the robust invisible watermark are computed whenever a new subscriber subscribes to the service. The unmasking frames are also computed whenever the mask frames are changed (for better security) but the robust invisible watermark need not be created again.

6 Conclusion

This paper proposes a novel integrated solution to address the broadcaster's copyright concern and subscriber's false-implication concern for a digital-video broadcast. The proposed scheme uses mask blending, watermarking techniques and interactive watermarking protocols to solve the challenging problem. The mask blending is intended to implement the confidentiality requirement against non-subscribers. The concern of broadcasters regarding copyright violation is addressed through the use of a mask-based, additive, robust, invisible watermarking technique. For this purpose, each subscriber receives a uniquely watermarked video. Broadcasting, however, requires a single copy to be transmitted. The proposed scheme efficiently meets both these requirements. The subscriber's false-implication concern is addressed through the use of an interactive watermarking protocol based on public-key cryptography. The protocol is carefully designed in such a way that neither the broadcaster nor the subscriber alone can perform the watermarking intended to mitigate the subscriber's concern. The proposed scheme supports dynamic join and leave, is not complex in terms of implementation and is also not taxing on the resources needed in terms of computing power and bandwidth. Our future directions are to extend the scheme to the audio stream of the broadcasts.

Acknowledgements. We would like to sincerely thank our colleague Harald Niederreiter for the many useful discussions.

References

1. Anderson R, Manifavas C (1997) Chameleon: a new kind of stream cipher. In: Biham E (ed) Fast software encryption, 4th international workshop, Haifa, Israel, 20–22 January 1997. Lecture notes in computer science, vol 1267. Springer, Berlin Heidelberg New York
2. Briscoe B, Fairman I (1999) Nark: receiver based multicast key management and non-repudiation. British Telecom technical report, June 1999
3. Brown I, Perkins C, Crowcroft J (1999) Watercasting: distributed watermarking of multicast media. In: Rizzo L, Fdida S (eds) Networked group communications, first interational COST264 workshop, Pisa, Italy, 17–20 November 1999. Lecture notes in computer science, vol 1736. Springer, Berlin Heidelberg New York
4. Chu HH, Qiao L, Nahrstedt K (1999) A secure multicast protocol with copyright protection. In: Proceedings of SPIE symposium on electronic imaging, science and technology, San Jose, Calif., 23–29 January 1999. SPIE, Bellingham, Wash.
5. Chou C-H, Li Y-C (1995) A perceptually tuned subband image coder based on the measure of just-noticeable-distortion profile. IEEE Trans Circuits Syst Video Technol 5(6):467–476
6. Cox IJ, Killian J, Leighton T, Shamoon T (1997) Secured spread spectrum watermarking for multimedia. IEEE Trans Image Process 6(12):1673–1687
7. Cox IJ, Miller LM, Bloom JA (2001) Digital watermarking. Academic Press, San Diego, Calif.
8. Craver S, Memon N, Yeo B, Yeung MM (1998) Resolving rightful ownerships with invisible watermarking techniques: limitations, attacks and implications. IEEE J Selected Areas Commun 16(4):573–586
9. Dittman J, Stabenau M, Steinmetz R (1998) Robust MPEG video watermarking technologies. In: Proceedings of the sixth ACM international conference on multimedia, Bristol, U.K. ACM Press, New York
10. Emmanuel S, Kankanhalli MS (2001) Mask based interactive watermarking protocol for video. In: Proceedings of the SPIE's ITCOM 2001, international symposium on the convergence of IT and communications, Denver, August 2001. SPIE, Bellingham, Wash.
11. Emmanuel S (2002) Digital rights management for video broadcasting. PhD thesis, Department of Computer Science, School of Computing, National University of Singapore
12. Hartung F, Girod B (1998) Watermarking of uncompressed and compressed video. Signal Process 66(3):283–301
13. Hartung F, Kutter M (1999) Multimedia watermarking techniques. Proc IEEE 87(7):1079–1107
14. Hartung F, Su JK, Girod B (1999) Spread spectrum watermarking: malicious attacks and counterattacks. In: Proceedings of SPIE symposium on electronic imaging, science and technology, San Jose, Calif., 23–29 January 1999. SPIE, Bellingham, Wash.
15. Judge P, Ammar M (2000) WHIM: watermarking multicast video with a hierarchy of intermediaries. In: Proceedings of the 10th international workshop on operating system support for digital audio and video, Chapel Hill, N.C., 26–28 June 2000
16. Kankanhalli MS, Rajmohan, Ramakrishnan KR (1999) Adaptive visible watermarking of images. In: IEEE international conference on multimedia computing and systems, Florence, Italy,

7–11 June 1999, vol 1. IEEE Computer Society, Los Alamitos, Calif.
17. Kirovski D, Peinado M, Petitcolas FAP (2001) Digital rights management for digital cinema. In: International symposium on optical science and technology, security in imaging, theory and applications, San Diego, Calif., July 2001
18. Kundur D, Hatzinakos D (1999) Digital watermarking for telltale tamper proofing and authentication. Proc IEEE 87(7):1167–1180
19. Kutter M, Voloshynovskiy S, Herrigel A (2000) The watermark copy attack. In: Proceedings of the SPIE, electronic imaging 2000, San Jose, Calif. SPIE, Bellingham, Wash.
20. Linnartz JP, Depovere G, Kalker T (1997) Philips electronics response to call for proposals issued by the data hiding subgroup copy protecion technical working group
21. Macq BM, Quisquater JJ (1995) Cryptology for digital TV broadcasting. Proc IEEE 83(6):944–957
22. Meng J, Chang SF (1998) Embedding visible video watermarks in the compressed domain. In: International conference on image processing, Chicago, Ill., 4–7 October 1998, vol 1. IEEE Computer Society Press, Los Alimitos, Calif.
23. Memon N, Wong PW (1998) A buyer seller watermarking protocol. In: Proceedings of the IEEE second workshop on multimedia signal processing, Redondo Beach, Calif., 7–9 December 1998. IEEE, Piscataway, N.J.
24. Niederreiter H (2000) Error correcting codes and cryptography. In: Proceedings of the public-key cryptography and computational number theory conference, Warsaw, 11–15 September 2000. Walter de Guyter, Berlin
25. Niederreiter H (1986) Knapsack-type cryptosystems based on algebraic coding theory. Probl Control Inf Theory 15:159–166
26. Parviainen R, Parnes P (2001) Large scale distributed watermarking of multicast media through encryption. In: Proceedings of the CMS 2001, Darmstadt, Germany, 21–22 May 2001
27. Qiao L, Nahrstedt K (1998) Watermarking schemes and protocols for protecting rightful ownership and customer's rights. J Vis Commun Image Represent 9(3):194–210
28. Piva A, Barni M, Bartolini F, Cappellini V (1997) DCT-based watermark recovering without resorting to the uncorrupted original image. In: International conference on image processing, Washington, D.C., 26–29 October 1997, vol 1. IEEE Computer Society Press, Los Alimitos, Calif.
29. Stallings W (2000) Cryptography and network security principles and practice. Prentice Hall, Upper Saddle River, N.J.
30. de Strycker L, Termont P, Vandewege J, Haitsma J, Kalker A, Maes M, Depovere G (2000) Implementation of a real-time digital watermarking process for broadcast monitoring on a TriMedia VLIW processor. IEE Proc Vis Image Signal Process 147(4):371–376
31. Voyatzis G, Pitas I (1999) The use of watermarks in the protection of digital multimedia products. Proc IEEE 87(7):1197–1207
32. Wolfgang RB, Podilchuk CI, Delp EJ (1999) Perceptual watermarks for digital images and video. Proc IEEE 87(7):1108–1126
33. Wolfgang RB, Delp EJ (1997) A watermarking technique for digital imagery: further studies. In: Proceedings of the international conference on imaging science, systems and applications (CISST 97), Las Vegas, Nev., June 1997
34. Zeng W, Lei S (1999) Efficient frequency domain digital video scrambling for content access control. In: Proceedings of the seventh ACM international conference on multimedia, part 1, Orlando, Fla., November 1999. ACM Press, New York
35. Boneh D, Shaw J (1998) Collusion-secure fingerprinting for digital data. IEEE Trans Inf Theory 44:1897–1905