Abstract—Multiparty multilevel digital rights management (DRM) architecture involving several levels of distributors in between an owner and a consumer has been suggested as an alternative business model to the traditional two-party (buyer-seller) DRM architecture for digital content delivery. In the two-party DRM architecture, cryptographic techniques are used for secure delivery of the content, and watermarking techniques are used for protecting the rights of the seller and the buyer. The cryptographic protocols used in the two-party case for secure content delivery can be directly applied to the multiparty multilevel case. However, the watermarking protocols used in the two-party case may not directly carry over to the multiparty multilevel case, as it needs to address the simultaneous security concerns of multiple parties such as the owner, multiple levels of distributors, and consumers. Towards this, in this paper, we propose a joint digital watermarking scheme using Chinese remainder theorem for the multiparty multilevel DRM architecture. In the proposed scheme, watermark information is jointly created by all the parties involved; then a watermark signal is generated out of it and embedded into the content. This scheme takes care of the security concerns of all parties involved. Further, in the event of finding an illegal copy of the content, the violator(s) can be traced back.

Index Terms—Chinese remainder theorem, digital rights management, watermarking.

I. INTRODUCTION

The ease with which digital contents can be obtained, replicated, and distributed without any loss of quality has resulted in widespread illegal replication and distribution of digital content. Hence, to prevent this and protect intellectual property rights, digital rights management (DRM) technologies have been developed. DRM uses cryptographic and digital watermarking techniques to prevent consumers from unauthorized copying of digital content, to control the use of digital content, and to enable the development of digital distribution platforms on which innovative business models can be implemented. In DRM, encryption is used to prevent unauthorized access to content, and watermarking is used to establish and prove ownership rights and to trace copyright violators by embedding the seller’s and buyer’s information into the digital content.

The traditional two-party DRM architecture involving a seller and a buyer is not adequate to satisfactorily address the requirements of the present-day business models for content delivery. Hence, multiparty multilevel DRM architecture (MPML-DRM-A) has been used as an alternative to the traditional two-party (buyer-seller) DRM architecture by many authors [12], [22]. The term “multiparty” refers to the multiple parties such as the owner, distributors, subdistributors, and consumers; the term “multilevel” refers to the multiple levels of distributors/subdistributors involved in the distribution chain of content.

In a multiparty multilevel DRM architecture, if each party embeds its watermark signal separately into the digital content, the quality of digital content will deteriorate with each watermarking. Therefore, how to protect the rights of the owner, distributors, and consumer through watermarking is a very important issue in this architecture. In this paper, we propose a joint digital watermarking mechanism for MPML-DRM-A using Chinese remainder theorem (CRT) [20]. Our approach is to embed into the content only one watermark signal generated from watermark information jointly generated by all the parties involved. We generate this joint watermark information using CRT. This was motivated by an application of CRT in secure broadcasting, effected by means of a secure lock by Chiou et al. [4]. The authors implemented this lock by using CRT. Analogously, we lock the identities of all the entities using CRT as a watermark that is embedded into the content. Our scheme thus takes care of the security concerns of all the parties involved. Further, in the event of finding an illegal copy of the content, the traitors can be traced.

The rest of this paper is organized as follows. In Section II, we discuss the preliminaries required. Section III includes our joint watermarking mechanism and its security and complexity analysis. A discussion on the implementation of the proposed scheme is given in Section IV. In Section V, we compare the proposed approach with the extensions of two-party solutions. This paper concludes with some remarks and future directions for research in Section VI.

II. PRELIMINARIES

In this section, we briefly discuss the preliminaries required for this paper.

A. Multiparty Multilevel DRM Architecture

For more innovative and scalable business models that have the flexibility of packaging multiple content together in a regional and culturally sensitive manner, it is necessary to have a more flexible and hierarchical distribution network. Hence, a...
multiparty multilevel architecture involving multiple levels of distributors and subdistributors in addition to the owner and consumers is necessary. A local distributor can better explore a potentially unknown market to the owner and make strategies according to the market. In addition, distributors can help in handling different price structures of media in different locations. Currently, this much flexibility is often lacking. Apple’s iTunes music store lets customers search a catalog of tracks. With one click, users can purchase the songs and download them. iTunes uses Apple’s FairPlay DRM system, which limits and controls its usage. To the best of our knowledge, there is no such mechanism like a joint watermarking or traitor tracing mechanism in iTunes, which is using only copy-protection software with cryptographic mechanisms to stick to certain devices on which the content can play. Hence, we adopt an MPML-DRM-A given in Fig. 1 as our content distribution model. The owner and distributors maintain their own content servers (CSs). To ensure the security of the content, the content is stored in an encrypted manner on the content servers. The license server issues redistribution licenses to distributors and a usage license to the consumer. A license grants the receiver specific permissions, constraints, and content decryption keys. A consumer is allowed to get the content from any of the content servers. We intend to build a joint watermarking mechanism into this architecture to take care of all the copyright issues.

B. Structure of Licenses

Licenses are created by the owner and distributors for other distributors and consumers. The license contains the following entries: identity of the license-issuing party, identity of the content(s), permissions, constraints, and keys required for taking appropriate action. There are two types of licenses in this architecture: redistribution license (RL) and usage license (UL).

Redistribution licenses are created by the owner or a distributor for another distributor lower in the distribution chain. The redistribution licenses contain secret keys of the party that generates the license for a particular content. Permissions include permission for content redistribution and permission to issue redistribution licenses. Constraints associated with permissions can be based on time, count, and location. Enforcement of a redistribution license is done with the help of a license.
server, which tracks redistribution and usage licenses issued by the owner/distributor.

Usage licenses are created by the owners and distributors for consumers to use the content. Consumers need to obtain the usage license of the owner as well as that of the distributor from whom the content was downloaded. Usage licenses contain the keys for opening the content and the permissions and constraints for using the content. Enforcement of a usage license is done with the help of a trusted DRM agent at the consumer’s machine.

C. Related Work and CRT

There exist several joint watermarking mechanisms [19], [15], [8] for the two party (buyer-seller) DRM architecture. However, as multiparty multilevel DRM architecture is a recent business model, there has not been any work on joint watermarking for this architecture yet. Our CRT-based joint watermarking scheme seems to be the first in this direction. The existence and uniqueness of $I$ as the (CRT) solution of a set of congruences corresponding to each party in the distribution chain.

III. PROPOSED WATERMARKING SCHEME

In this section, we describe our joint watermarking protocol based on CRT for MPML-DRM-A. The proposed watermarking protocol involves the following entities: an owner $O$, $k$ levels of distributors $D_1,\ldots,D_k$ (there can be no distributor also), a consumer $C$, and a license server $L$. We generate the joint watermark information $I$ as the (CRT) solution of a set of congruences corresponding to each party in the distribution chain.

The watermark signal $W$ is generated from this joint watermark information using a watermark generation algorithm and then embedded into the content using a robust embedding algorithm. The watermark signal is detected using the corresponding watermark detection algorithm.

A. Generation of Individual Watermark Information

Each party $i$ (owner or distributor or consumer) involved in the content delivery generates its individual watermark information $r_i$ using its private key $d_i$ as its digital signature $r_i = \text{Sig}(Y|d_i)$, where $Y = H(H(X)||I_X)$.

B. Generation of Joint Watermark Information

Let $r_0, r_1, \ldots, r_k$ and $r_{k+1}$ be the individual watermark information of the parties $O, D_1,\ldots,D_k$, and $C$, respectively, computed as digital signatures, as described in the previous section. Let $n_0, n_1,\ldots,n_{k+1}$ be relatively prime integers assigned to these parties, respectively. Then their joint watermark information $I$ is the solution of the following set of $k+2$ congruences:

$$I \equiv r_i \mod n_i, \quad i = 0, 1, \ldots, k+1,$$

The existence and uniqueness of $I$ are guaranteed by CRT.

C. Joint Watermarking Protocol

Recall the notations given in Section II-D. Let content reach a consumer $C$ from the owner $O$ through $k$ distributors $D_1,\ldots,D_k$. We now describe the watermarking protocol below. The steps performed by the owner, distributors, and the consumer are separately described.

We begin with the interactive protocol and the computations performed by the owner with the license server.

1) $O$ sends Cert$_O$ to the license server $L$. 

D. Notations

We use the following notations throughout this paper.

1) The entities involved are an owner $O$, $k$ distributors $D_1,\ldots,D_k$, a consumer $C$, and a license server $L$.
2) $X$ denotes the content and $I_X$ a unique copy number of the content $X$.
3) $E_{\text{pub}}(\cdot|K)$, $D_{\text{pub}}(\cdot|K)$, $\text{Sig}(\cdot|K)$, and $\text{Ver}(\cdot|K)$ denote the encryption, decryption, digital signature generation, and digital signature verification algorithms (with key $K$) respectively, corresponding to a standard public key cryptosystem.
4) For $0 \leq i \leq k+1$, $(c_i, d_i)$ and Cert$_i$ denote the public–private-key pairs and the public-key certificate assigned to the owner, the $k$ distributors, and the consumer, respectively.
5) For $i = 0, \ldots, k$, CST$_i$ denotes the content server of the owner and the $k$ distributors $D_1,\ldots,D_k$, respectively.
6) $E_{\text{sym}}(\cdot|K)$, $D_{\text{sym}}(\cdot|K)$ denote the encryption and decryption algorithms corresponding to a standard symmetric-key cryptosystem like AES or DES.
7) $E_{\text{sym}}(x_1,\ldots,x_p|K) = (E_{\text{sym}}(x_1|K),\ldots,E_{\text{sym}}(x_p|K))$, $D_{\text{sym}}(y_1,\ldots,y_p|K) = (D_{\text{sym}}(y_1|K),\ldots,D_{\text{sym}}(y_p|K))$.
8) $H(\cdot)$ denotes any standard hash function such as SHA1 or MD5.
9) PRNG(\cdot) denotes a binary pseudorandom number generator.
10) Let $\| \$ denote the concatenation operator, for $0 \leq i \leq k+1$; let $Y = H(H(X)||I_X)$ and $r_i = \text{Sig}(Y|d_i)$.
11) $W_{\text{sym}}(\cdot|K)$ denotes any standard watermark signal generation algorithm (with key $K$) from watermark information.
12) $W_{\text{embedding}}(\cdot|K)$ denotes any standard robust watermark embedding algorithm and $W_{\text{detect}}(\cdot|K)$ the corresponding watermark detection algorithm (with key $K$).
13) $K_X$ denotes the key used for embedding the watermark signal in the content $X$ and $K'_X$ the key used for detecting the same watermark signal.
14) $I$ denotes the joint watermark information and $W$ the joint watermark signal.
15) $J$ denotes a judge who is called for arbitration in case of a dispute.
16) For $i = 0,\ldots,k$, UL$_i$ and RdL$_i$ denote the usage and redistribution licenses of $O, D_1,\ldots,D_k$, respectively.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on January 17, 2010 at 22:53 from IEEE Xplore. Restrictions apply.
2) \( L \) verifies \( \text{Cert}_O \), extracts the public key \( e_0 \) of \( O \) from \( \text{Cert}_O \), generates a random session key \( K_0 \), and sends \( K' = E_{\text{pub}}(K_0) \) to \( O \).

3) \( O \) decrpts \( K_0 = D_{\text{pub}}(K' | d_0) \), computes its watermark information \( r_0 = \text{Sig}(H(H(X) || I_X) | d_0) \), and generates its usage license \( U_{L_0} \) and redistribution license \( R_{L_0} \). It encrypts them using \( K_0 \) as \( Y = E_{\text{sym}}(r_0, U_{L_0}, R_{L_0}, r_0, H(X), I_X | K_0) \).

4) \( O \) sends \( Y \) to \( L \).

5) \( L \) does \( D_{\text{sym}}(Y | K_0) = (r_0, U_{L_0}, R_{L_0}, r_0, H(X), I_X) \). It verifies \( r_0 \) and checks the licences \( U_{L_0} \) and \( R_{L_0} \). If they are correct, \( L \) adds to its database (\( \text{Cert}_O, U_{L_0}, R_{L_0}, r_0, H(X), I_X \)) and notifies \( O \).

6) \( O \) generates a unique watermark signal \( W_{\text{own}} \) as a function of \( I_X \) and embeds into the content \( X \) to get \( X' \). It then encrypts \( X' \) and uploads on its content server \( CS_O \).

We now describe the interactive protocol and the computations performed by a distributor \( D_L \) with \( L \).

1) \( D_L \) downloads (encrypted) content from the content server \( CS_{\text{consumer-1}} \) and sends the request for the redistribution licence of \( D_{\text{consumer-1}} \) along with its public-key certificate \( \text{Cert}_L \) to \( L \).

2) \( L \) verifies the public-key certificate \( \text{Cert}_L \) and extracts the public key \( e_L \) of \( D_L \) from \( \text{Cert}_L \). It generates a random session key \( K'_L \) encrpts it using the public key of \( D_L \) as \( K' = E_{\text{pub}}(K'_L) \) and then encrypts the licence and other parameters using the session key as \( Y = E_{\text{sym}}(\text{RdL}_{L-1}, H(X), I_X | K'_L) \) and sends \( Y, K' \) to \( D_L \).

3) \( D_L \) decrypts \( K' \) and \( Y \) as \( K_L = D_{\text{pub}}(K' | d_L) \) and \( D_{\text{sym}}(Y | K_L) = (\text{RdL}_{L-1}, H(X), I_X) \). It computes its watermark information \( r_L = \text{Sig}(H(H(X) || I_X) | d_L) \) and generates its usage license \( U_{L_0} \) and redistribution license \( R_{L_0} \), encrypts to get \( Y = E_{\text{sym}}(r_L, U_{L_0}, R_{L_0}, r_L, I_X | K_L) \), and sends \( Y \) to \( L \).

4) \( L \) decrypts \( Y \) as \( D_{\text{sym}}(Y | K_L) = (r_L, U_{L_0}, R_{L_0}, r_L, n_L) \). It then verifies \( r_L, U_{L_0}, R_{L_0}, \) and \( n_L \). If they are correctly generated, it adds to its database (\( \text{Cert}_L, U_{L_0}, R_{L_0}, r_L, n_L \)) and notifies \( D_L \).

5) \( D_L \) uploads (encrypted) content on its content server \( CS_L \).

In the final stage, a consumer \( C \) downloads the content from the content server of the distributor \( D_L \). The consumer’s machine has a separate module for storing its secret keys and carrying out sensitive/secure computations. In the following protocol, \( C \) denotes this module. Further, the consumer has installed a DRM agent of the owner in its machine. Although DRM agent is an entity of the owner, it is hard for it to collude with the owner against the distributors and consumers. The DRM agent is provided only with the joint watermark information. The options in front of the DRM agent are whether to embed the joint watermark or not, and the DRM agent will do the embedding for the sake of the owner. Further, since it does not have access to the watermark information of the other entities (including that of the consumer), it cannot act against them. We assume that there is cryptographic key \( K_{\text{drm}} \) associated with the DRM agent. The licence server can find this key in the usage license \( U_{L_0} \) of the owner. Formally, the steps are as follows.

1) \( C \) downloads (encrypted) content anonymously from the content server \( CS_k \) of the distributor \( D_k \) and sends the public-key certificate \( \text{Cert}_{k+1} \) to \( L \) and requests for starting a session.

2) \( L \) verifies \( \text{Cert}_{k+1} \), extracts the public key \( e_{k+1} \) from \( \text{Cert}_{k+1} \), generates a random session key \( K_{k+1} \) encrypts using the public key of \( C \) as \( K' = E_{\text{pub}}(K_{k+1}) \) and sends \( K' \) to \( C \).

3) \( C \) decrypts \( K' \) as \( K_{k+1} = D_{\text{pub}}(K' | d_{k+1}) \) and then sends to \( L \), the request for the usage licence of the distributor, after encrypting both the identity of \( D_k \) and the identifier for the content \( X \) with \( K_{k+1} \).

4) \( L \) decrypts the identity of \( D_k \) and the identifier for the content \( X \) with \( K_{k+1} \) and then identifies the content information, the owner, and all the distributors associated with it in its database. It then encrypts the usage licence of the distributor and other parameters to get \( Y_1 = E_{\text{sym}}(H(X), I_X | K_{k+1}), Y_2 = E_{\text{sym}}(U_{L_k} | K_{\text{drm}}) \) and sends \( (Y_1, Y_2) \) to \( C \).

5) \( C \) decrypts \( Y_1 \) as \( D_{\text{sym}}(Y_1 | K_{k+1}) = (H(X), I_X) \), and the DRM agent decrypts \( Y_2 \) as \( D_{\text{sym}}(Y_2 | K_{\text{drm}}) = U_{L_k} \).

6) \( C \) computes the watermark information of the consumer as \( r_{k+1} = \text{Sig}(H(H(X) || I_X) | d_{k+1}) \), generates a random number \( n_{k+1} \) coprime to \( n_0, \ldots, n_k \), digitally signs it as \( \text{Sig}(n_{k+1}) = \text{Sig}(n_{k+1} | d_{k+1}) \), encrypts \( r_{k+1} \) and \( n_{k+1} \) as \( Y = E_{\text{sym}}(r_{k+1}, n_{k+1} | K_{k+1}), \) and sends \((Y, \text{Sig}(n_{k+1})) \) to \( L \).

7) \( L \) computes \( D_{\text{sym}}(Y | K_{k+1}) = (r_{k+1}, n_{k+1}) \) and checks that, for all \( 0 \leq i \leq k, n_{k+1} \) is coprime to \( n_i \). If it does not hold, \( L \) requests \( C \) to resend \( n_{k+1} \) and \( \text{Sig}(n_{k+1}) \). It then verifies signatures \( r_{k+1}, \text{Sig}(n_{k+1}) \). If all verifications pass through, it adds to its database the entry \( (\text{Cert}_{k+1}, n_{k+1}, r_{k+1}, \text{Sig}(n_{k+1})) \).

8) \( L \) computes the joint watermark information \( I \) as the CRT solution of the following equations:

\[
I \equiv r_i \mod n_i, \quad \text{where} \quad 0 < i, \ldots, k + 1
\]

encrypts and sends \( Y = E_{\text{sym}}(U_{L_0}, I | K_{\text{drm}}) \) to \( C \).

The DRM agent decrypts \( D_{\text{sym}}(Y | K_{\text{drm}}) = (U_{L_0}, I) \), opens the content using the keys in \( U_{L_0} \) and \( U_{L_k} \) to get \( X' \), computes a watermark signal \( W \) from the watermark information \( I \) using the watermark signal generation algorithm \( W_{\text{gen}}(\cdot) \), and then embeds into the content \( X' \) using the watermark signal embedding algorithm \( W_{\text{emb}}(\cdot | K_X) \).

D. Watermarking Detection and Traitor Tracking

We assumed that the watermark signal \( W \) is generated and then embedded using a well-known robust watermarking algorithm. Suppose that the owner \( O \) found an illegal copy \( Y \) of his content \( X \). Let \( J \) denotes a judge for arbitration. The traitor tracing protocol is as follows.

1) \( O \) checks whether its watermark signal \( W_{\text{own}} \) is present in the content \( Y \). If it is not present, END the protocol; else proceed.

2) \( O \) presents \((Y, W_{\text{gen}}(H(X), I_X)) \) to \( J \).

3) \( J \) checks whether \( W_{\text{own}} \) is present in \( Y \). If it is not present, END the protocol; else proceed.

4) \( J \) gets the joint watermark information \( I \) from the license server \( L \) and computes the watermark signal \( W = W_{\text{gen}}(I) \). It obtains \( K_X \) and checks whether \( W \) is present in \( Y \) using the detection algorithm \( W_{\text{det}}(\cdot | K_X) \). If \( W \) cannot be detected in \( Y \), END the protocol; else proceed.
TABLE I

<table>
<thead>
<tr>
<th>Computational Complexity</th>
<th>O</th>
<th>D</th>
<th>C</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetric-key Encryption</td>
<td>6</td>
<td>4</td>
<td>3k+5</td>
<td>2k+5</td>
</tr>
<tr>
<td>Symmetric-key Decryption</td>
<td>0</td>
<td>3</td>
<td>4k+10</td>
<td>2k+2</td>
</tr>
<tr>
<td>Public-key Encryption</td>
<td>0</td>
<td>0</td>
<td>k+1</td>
<td>0</td>
</tr>
<tr>
<td>Public-key Decryption</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Digital Signature Generation</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Digital Signature Verification</td>
<td>0</td>
<td>0</td>
<td>2k+5</td>
<td>2k+5</td>
</tr>
</tbody>
</table>

5) $J$ gets $(n_{k+1}, \text{SIG}(n_{k+1}), \text{Cert}_{k+1})$ from $L$.
6) $J$ computes $r_{k+1}$ from the equation $I \equiv r_{k+1} \mod n_{k+1}$.
7) $J$ checks whether $r_{k+1}$ is a valid watermark information of the consumer $C$ by verifying whether $r_{k+1}$ is a valid signature of $C$ and $n_{k+1}$ is a random number generated by $C$ by verifying the signature $\text{SIG}(n_{k+1})$. If both verifications pass through, $J$ concludes that $C$ was the consumer associated with that content and hence was the traitor.

Note that, in the proposed solution, the distributors are not able to identify an illegal copy distributed without the help of the license server. Only the owner has this possibility, thanks to the unique watermark $W_{\text{own}}$ of the owner embedded into the content by the owner.

E. Complexity of the Protocol

In this section, we analyze the complexity of the proposed scheme. Most of the encryption operations used are symmetric-key cryptography based to minimize the costly public-key cryptographic operations. The public-key certificates $\text{Cert}_{0}, \ldots, \text{Cert}_{k+1}$, the parameters $\eta_0, \ldots, \eta_k$ in the CRT equations, and the watermark detection key $K'_{X}$ are publicly available. We now analyze the communication, computational, and storage complexity of the proposed scheme.

Assume that there are $k$ distributors. The license server $L$ exchanges a total of $2k+5$ messages. $L$ needs to verify $2k+5$ digital signatures, compute $2k+2$ public-key encryption, $3k+5$ symmetric-key encryptions, and $3k+8$ symmetric-key decryptions. It also generates $k+2$ random numbers, performs one CRT computation (if $n_i$'s are $t$ bit numbers, the complexity for this computation is $O(kt^2)$ [20]), and verifies $k+1$ usage licenses and redistributions licenses each. The license server needs to store $k+2$ digital certificates, $k+1$ usage and redistribution licenses each, $k+3$ digital signatures, $k+2$ prime numbers, hash of the content $h(X)$, identifier for that content $t_{iX}$, and the joint watermark information $I$. Thus, the overall complexity of $L$ is linear in the number of distributors $k$. Practically, $k$ is a small number.

The major computations of owner, distributors, consumer, and license server are summarized in Table I.

F. Security Analysis

We will now analyze the security of our protocol. The soundness and completeness of the protocol rely on the security and robustness of the underlying cryptographic and watermarking primitives and the trustworthiness of the license server and the DRM agent.

1) **Traitor Tracing:** If the owner finds an illegal copy of the content, he can identify all the distributors and the consumer involved in the distribution of the content using the protocol given in Section III-D.

2) **Security Against False Framing:** The scheme offers protection for parties who were not associated with content against wrong identification or false framing as follows. Let $n$ and $(e, d)$ be the parameters of a party. The judge computes $r$ from the equation $I \equiv r \mod n$ and checks whether $r$ is a valid signature of that party by checking whether $\text{Ver}(r, H(H(X) || I_{X}) || I_{X}) = 1$ holds. If the party was not involved, this verification will fail, as its success corresponds to the existential forgery of the signature $\text{Sig}(H(H(X) || I_{X}) || I_{X}) = 1$, which is not possible as the underlying digital signature scheme is secure.

3) **Rights of Consumer and Distributors:** Since the watermark signal is generated and then embedded by the DRM agent, the owners or distributors cannot create copies of the original content containing the consumer’s watermark. Further, since the watermark signal is formed from the joint watermark information, the owner or distributor will not be able to frame false allegations against a lower level distributor or consumer regarding illegal distribution of a content.

4) **Binding of Watermark to Content:** The individual watermark information $r_{iX}$ and hence the joint watermark information $I$, are generated as a function of the content $(H(X))$ and the identifier $(I_{X})$. Thus, the watermark signal $W$ is bound to the content.

5) **Proof of Ownership and Distributorship:** In case of a dispute, the owner or distributor can settle the dispute using the protocol given in Section III-D with the help of a judge.

6) **Collusion Attack:** The term “collusion attack” in the watermarking literature usually refers to a coalition of users that compare their watermarked contents in order to gain information about the watermarking process and/or remove the watermark. These types of attacks are not specific to our proposal and depend on the strength and robustness of the specific watermarking algorithm used. In our case, the individual watermark information $r_{iX}$ is generated by the parties themselves as their digital signature. The license server verifies $r_{iX}$ and stores them in the database. This prevents collusion attacks in the generation of $I$.

7) **Embedding of Correct Watermark Signal:** The license server verifies the individual watermark information $r_{iX}$ and stores it in the database. The redistribution license of the $i$th party is accepted by the license server only if $r_{iX}$ was correctly generated. In the final stage, the license server verifies the watermark information $r_{k+1}$ of the consumer and generates the joint watermark information $I$. The watermark signal $W$ is generated from $I$ and then embedded into the content by the DRM agent. The DRM agent is the owner’s entity residing in a consumer’s machine and performing actions on content according to the usage licenses. Since the DRM agent is a trusted entity representing the owner, these steps will be carried out correctly. If not, the owner will not be able to trace the traitors if he finds illegal copies in the future, and will not be able to trace the distributors involved in the distribution of his content. Thus, the watermark signal will be correctly embedded into the content.
8) Privacy/Anonymity of Consumers: The scheme protects the privacy concerns of a consumer. The consumer downloads the content anonymously and generates a random number \( n_i \) towards generating the joint watermark information \( I \). Thus, \( I \) does not reveal the identity of the consumer. While interacting with the license server, the consumer maintains privacy by sending only encrypted information about the content it downloaded. Although \( I \) does not reveal the identity of the consumer, if a need arises, the consumer can be identified. Further, the watermark signal embedding key \( K_X \) and the detection key \( K'_X \) depend only on the content and are common for all the consumers using the same content \( X \). This choice also ensures the privacy of the consumers.

IV. IMPLEMENTATION OF THE PROPOSAL

In this section, we discuss an implementation of the proposed scheme.

A. Secure Delivery of the Content

The owner and the distributors encrypt the content \( X \) before uploading on their content servers. The set of components of \( X \) to be encrypted is divided into two mutually disjoint sets of components \( X_O \) and \( X_D \). Now the owner (corresponding to role as owner) encrypts \( X_O \) and the distributors/owner (owner corresponding to role as a distributor) encrypt \( X_D \). For security reasons, the set \( X_O \) should contain the major components of \( X \). The encryption of \( X_O \) and \( X_D \) is performed using symmetric key encryption algorithms. Whenever a distributor obtains the content from a higher level distributor/owner, he decrypts the encryption (by that distributor/owner) of \( X_D \) using the key in the redistribution license and then reencrypts \( X_D \) using his key and uploads the resultant content on his content server. A consumer downloads the content from the content server of any distributor/owner. It then obtains the usage licenses by the owner and the distributor from the license server. The usage license of the owner contains the key for decrypting \( X_O \) and the usage license of the distributor/owner contains the key for decrypting \( X_D \).

B. Joint Watermark Information Generation

All the parties get their cryptographic credentials (public-private keys and digital certificate) from a key generation/certifying authority. Further, the owners and distributors collect their CRT parameter \( n_i \) along with a certificate of its ownership from another or the same key generation/certifying authority. SHA-1 is chosen as the one-way hash function \( H(\cdot) \) for use.

The parties generate their individual watermark information \( r_i \) as their elliptic curve digital signatures (ECDSA) [14] on \( H(H(X)||l_X) \), where \( X \) is the content and the license server finally generates the joint watermark information, as described in the Section III.

For security level of 80 bits, the ECDSA signature size is 320 bits. The size \( t \) of the numbers \( n_i \) can be chosen as any number greater than 320. This is to ensure that \( r_i \)'s do not get modulated out in the congruence relation \( I \equiv r_i \mod n_i \), so that \( r_i \)'s could be extracted out from \( I \) and verified. The authority generating \( n_i \)'s needs to ensure they are coprime to each other (another way to generate \( n_i \)'s is to allow \( O \) to generate a prime number \( n_i \) of size \( t_0 \), each \( D_i \) to generate a prime number \( n_i \) of size \( t_i \), and \( C \) to generate a prime number \( n_{k+1} \) of size \( t_{k+1} \) individually such that \( t_0 < t_1 < \ldots < t_k < t_{k+1} \). Since \( 0 \leq I \leq n_0 \ldots n_{k+1} \), the size of \( I \) is \( (k+2)t \) bits.

C. Watermark Signal Generation

The DRM agent obtains the joint watermark information \( I \) from the licence server as described in the protocols in Section III. Let \( s = H(I) \) and \( p_0 \) be a seed obtained from \( s \) (\( p_0 \) could be a few bits of \( s \)). A pseudorandom number sequence \( p_1, p_2, \ldots \) is generated using PRNG(\( \cdot \)) with \( p_0 \) as seed. If the watermark signal needed to be embedded is not larger than 160 bits, it is advisable to generate \( p_1, p_2, \ldots \) from \( s \) itself by interpreting 0 bits as \(-1 \) and 1 bit as \( 1 \), instead of using PRNG(\( \cdot \)) to prevent possible collisions in the generated watermark signal. The spread spectrum watermark signal \( w_1, w_2, \ldots \) is generated from \( p_1, p_2, \ldots \) as \( w_i = \alpha_i b_i p_i \), for all \( i \geq 1 \), where \( \alpha_i \geq 0 \) is a locally adjustable amplitude factor and \( b_i \) is the spread sequence (see [9] for details). \( w_i \) is then embedded into the content using the spread-spectrum watermarking techniques given in [9].

D. Watermark Signal Embedding

The watermark signal generation from \( I \) and embedding are carried out by the DRM agent in the consumer’s device. This may be implemented using one-time pads [7] or stream switching [13] or joint decryption and watermarking [18] or lookup-table-based ciphers [1]. For simplicity, we assume that a tamper-proof box—or, more specifically, a trusted platform module (TPM)—is available at the consumer’s machine. A trusted computing group has specified the components of a TPM, which is a tamper-resistant module and can be trusted to store security-sensitive data in ways testable by a remote party. TPM can enforce access-control policies associated with a resource in such a way that a user cannot bypass these policies while maintaining access to resource.

The watermark embedding key \( K_w \) is chosen as a function of the media. For example, if \( H(\cdot) \) is a hash function and \( X = \{x_1, x_2, \ldots, x_m\} \) is a representation of the media \( X \) as a vector of real components, then we may compute \( K_w \) as \( K_w = H(y_1 || y_2 || \ldots || y_m) \) or a key derived from the above hash using a key derivation function, where, \( \forall i, y_i = [\lfloor x_i \rfloor ] \) (absolute value of the integral part). We assume that this key is available in the usage license UL of the owner.

E. Watermark Signal Detection

To detect the watermark information in a content, first the database of watermark information is obtained from the license.
server. The pseudorandom number sequences are then generated as described in the Section IV-C. The watermark signal is then detected in the content using the watermark detection algorithm given in [9]. The watermark detection key $K'_X$ is either the same as the embedding key $K_X$ or, if it is different, it is obtained from the owner. The detection key $K'_X$ is available with the owner, license server (from UL), and DRM agent.

V. COMPARISON OF THE PROPOSAL

In this section, we compare our approach with the extensions of buyer-seller protocol to the MPML-DRM-A.

A. Extensions of Buyer-Seller Protocols

In this section, we compare our approach with the extensions of buyer-seller watermarking protocols [19] and [17] to the multiparty multilevel architecture. The buyer-seller watermarking protocol is a three-party protocol between a seller, a buyer, and a trusted watermark certification/generation authority (WCA/WGA). The seller sells copies of the content to a buyer, logging all transactions in a local database, whose entries facilitate tracing of content. The role of WCA/WGA is to ensure an honest generation of watermarks and send them along with a time-stamp and a digital signature.

In [19], Memon et al. described a buyer-seller watermarking protocol using homomorphic public-key cryptosystems. Here the seller first embeds his watermark signal into the content and then embeds a transformation (permutation) of the watermark signal of the buyer into the already watermarked content and passes the resultant content to the buyer. The extension of this protocol to the MPML-DRM-A is as follows. The owner and $D_1$ execute a buyer-seller watermarking protocol. $D_1$ gets the watermarked content (two watermark signals embedded). Then $D_1$ gets the watermark signal of $D_2$ (in encrypted format), embeds a transformation of it into the content, and passes the resultant content to $D_2$. There are three watermark signals in the content now. It is easy to see that the security concerns of $O$, $D_1$, and $D_2$ are taken care of here ($D_1$ need not have to put an additional watermark signal before passing the resultant content to $D_2$, as the role of seller’s watermark signal is taken care of by the joint watermark signal between $O$ and $D_1$). Finally, the consumer passes his watermark signal (encrypted) to $D_k$, and $D_k$ embeds a transformation of it into the content received from $D_{k-1}$. $D_k$ then passes the resultant content to the consumer $C$. We can see that there will be $k+2$ watermark signals in the content the consumer is receiving. In general, with this approach, the number of watermark signals in the content increases linearly with the number of distributors and thus it is not scalable. With our joint watermarking mechanism, there will be only two watermark signals in the content the consumer receives, irrespective of the number of distributors involved. This can be even reduced to just one watermark signal by embedding only the joint watermark into the content and ignoring the separate watermark signal ($W_{own}$) embedding by the owner. However, the main disadvantage of the scheme of Memon et al. is the use of costly (homomorphic) public-key encryption mechanisms. Further, the scheme will not work if an intermediate entity behaves maliciously. If any party embeds a wrong watermark, the whole watermark will get corrupted, and it will not even possible to trace the malicious entity. So a trusted third party will be needed to ensure that each party performs its role correctly.

In [17], Katzenbeisser et al. proposed a buyer-seller watermarking protocol that avoids the use of homomorphic public-key encryption and uses a secure watermark embedding based on partial encryption. The two-party protocol of Katzenbeisser et al. may be described in simple terms as follows. Let WGA be a trusted watermark generation authority. $X$ denote content (represented as a vector of quantized real numbers that denote samples in the spatial/temporal domain or coefficients in a transform domain), $W$ denote a watermark signal, and $K$ denote an encryption key.

1) WGA generates $W$ and $K$.
2) WGA sends $K$ to $O$ and $W \oplus K$ to $C$.
3) $O$ computes $X \oplus K$ and sends to $C$.
4) $C$ computes $(X \oplus K) \oplus (W \oplus K) = X \oplus W$.

Thus the consumer $C$ gets the watermarked content $X \oplus W$. Note that the scheme works only if the mutual cancellation in Step 4) holds. In particular, the scheme does not work with any encryption of the content or any watermarking mechanism. A straightforward extension of the above protocol to our multiparty multilevel DRM architecture may be as follows. WGA chooses random keys $K_1, K_2, \ldots, K_k$. WGA sends $K$ to $O$, $K_i$ to $D_i$, for $1 \leq i \leq k$, and $W \oplus K_i$ to $C$, where $K_i$ is such that $K = K_1 \oplus \ldots \oplus K_k \oplus K_c$. $O$ computes $X \oplus K$ and uploads on its content server. $D_1$ downloads $X \oplus K$, computes $X \oplus K \oplus K_1$, and uploads on its content server. Finally, $D_k$ computes $X \oplus K \oplus K_1 \oplus \ldots \oplus K_k$ and uploads on its content server. The consumer $C$ downloads $X \oplus K \oplus K_1 \oplus \ldots \oplus K_k$, and obtains $W \oplus K_c$ from WGA. $C$ computes $X \oplus K \oplus K_1 \oplus \ldots \oplus K_k \oplus W \oplus K_c = X \oplus W$. Thus, the consumer $C$ gets the watermarked content $X \oplus W$.

The main disadvantage with this approach is that it does not work with any encryption of the content or any watermarking of the content. Our approach is applicable to any encryption of the content and any watermarking of the content. The computational and communication load on the WGA under this scheme is comparable to that on the license server $L$ in our scheme. WGA is required to perform many complex computations like: (PKC) decryption of session keys, generation of watermark sequence, generation of multiple keys for encrypting watermark, encryption of watermark (XOR), encryption of encrypted watermark (SKC), PKC encryption of watermarks and keys, and digital signature generation [17]. The identity of all the entities involved can be immediately derived in a unique manner from the watermark—in our case from the CRT equations—whereas this is not immediate in this case, as the key $K$ could be split up in several ways that point to different entities at the same time. Further, as in the previous case, the above extensions will succeed only if all the parties involved are honest. A trusted third party is needed to verify that each party is performing correct steps.

We conclude that the role license server $L$ in our architecture is analogous to the role of WCA and WGA in the buyer-seller watermarking protocols and its extensions. Thus, all these protocols use a trusted third party. Minimization of dependence on any of these trusted parties requires implementation of more complex interactive protocols and cryptographic primitives.

B. Experimental Study

In this section, we carry out a set of experiments to evaluate the performance of our proposal. In the first set of experiments,
we study the effect of multiple watermarking on images. In the second set of experiments, we fix the watermark strength and study the robustness of multiple watermarking on images.

1) Effect of Multiple Watermarking on Quality: In this section, we do a comparison of the naive extension of the buyer-seller watermarking (resulting in multiple watermarks on the content) with our approach. It is carried out using the spread-spectrum watermarking algorithm of Hartung and Girod [9] with parameters $\alpha = 5$ as the amplification factor and $cr = 2000$ as the chip-rate. The results of the experiments with the standard test images of Lena, Baboon, and Pepper are given in Table II and illustrated in Figs. 2–4. The four images in each figure are given in the following order: first row: original image and image with one watermark (left to right); second row: image with two and ten watermarks (left to right), respectively. The figures clearly show the deterioration of the quality of the images with multiple watermarking. We expect similar results with other watermarking algorithms.

The peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) index between the original and the watermarked images are given in Table II.

PSNR represents the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Let $PV_{\text{org}}(i, j)$ and $PV_{\text{wat}}(i, j)$ denote the pixel value of the original image and the watermarked image at $(i, j)$ ($m \times n$ grayscale images). Then PSNR is computed by the following equations:

$$\text{PSNR} = 20 \log \frac{255}{\sqrt{\text{MSE}}}$$

where

$$\text{MSE} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (PV_{\text{org}}(i, j) - PV_{\text{wat}}(i, j))^2.$$  

SSIM is a perceptual measure used for measuring the similarity between two images. It is designed to improve on methods like PSNR and MSE [24]. The SSIM metric is calculated on various windows of an image. The measure between two windows of size $N \times N$, $x$ and $y$, is

$$\text{SSIM}(x, y) = \frac{(2\mu_x \mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

where $\mu_x$ is the average of $x$, $\mu_y$ is the average of $y$, $\sigma_x^2$ is the variance of $x$, $\sigma_y^2$ is the variance of $y$, $\text{cov}_{xy}$ is the covariance of $x$ and $y$, $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$, $L = 2$ bits per pixel, $k_1 = 0.01$, and $k_2 = 0.03$.

The SSIM index lies between $-1$ and 1, and value 1 is only reachable in the case of two identical sets of data. Typically, it is calculated on window sizes of $8 \times 8$.  

<table>
<thead>
<tr>
<th>Watermark</th>
<th>Single WM PSNR</th>
<th>Single WM SSIM</th>
<th>Double WM PSNR</th>
<th>Double WM SSIM</th>
<th>Multiple(10) WM PSNR</th>
<th>Multiple(10) WM SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>37.1718</td>
<td>0.82156</td>
<td>31.1645</td>
<td>0.56598</td>
<td>17.9468</td>
<td>0.08309</td>
</tr>
<tr>
<td>Baboon</td>
<td>37.1701</td>
<td>0.93796</td>
<td>31.1495</td>
<td>0.81924</td>
<td>17.2268</td>
<td>0.27282</td>
</tr>
<tr>
<td>Pepper</td>
<td>37.2395</td>
<td>0.86246</td>
<td>31.2277</td>
<td>0.65612</td>
<td>17.5349</td>
<td>0.16106</td>
</tr>
</tbody>
</table>

Fig. 2. *Lena*: original and watermarked images.

Fig. 3. *Baboon*: original and watermarked images.

Fig. 4. *Pepper*: original and watermarked images.
PSNR and SSIM of the images with single, double, and multiple (ten) watermarking are given in Table II. The rapid decrease in the PSNR and SSIM along a row shows the deterioration of the quality of the content with multiple watermarking.

2) Robustness of Multiple Watermarking: In this section, we examine the probability of detection of the watermarks from a watermarked content. We computed the ratio of the total number of bits of a watermark correctly detected to the total number of bits in that watermark. We may interpret this ratio as the probability of detection of that watermark. Let $p$ denote the product of the ratios (of the number of bits detected to the total number of bits in the watermark) of all the watermarks embedded into the content. We may interpret $p$ as the probability of detection of all the watermarks in the content. Let $n_{0.60}$ denote the number of watermarks in a content where the ratio of number of bits detected to the total number (probability) is greater than 0.60 and $n_{0.75}$ denotes the number of watermarks where this ratio (probability) is greater than 0.75. We performed experiments with two watermarks and ten watermarks embedded on nine test images. The six additional test images used are given in Fig. 5. The watermarking algorithm and the chip-rate used are the same as those in the previous section. The parameter $\alpha$ and the number of bits in the watermark ($W_{kn}$) are varied in each case to get different watermark strengths. The results of the experiments are given in Table III.

We observed the following from our experiments.

1) If we fix $W_{kn}$ and increase $\alpha$ (this increases the watermark strength), the probability of detection of watermarks increases. This implies that higher probability of detection can be obtained at a cost on the quality of the content.

2) If we fix $\alpha$ and decrease $W_{kn}$ (this decreases watermark strength), the probability of detection of watermarks increases. This implies that higher probability of detection can be obtained at a cost on the security level of the watermark.

We conclude that with our approach, the watermarks (two) can be detected without compromise on the quality or on the security, whereas with multiple watermarking, either the quality of the content or the security of the watermark is to be compromised to detect all the watermarks embedded into the content.

VI. CONCLUSION

In this paper, we presented a joint watermark protocol for the MPML-DRM-A using Chinese remainder theorem. The proposed scheme ensures that only two watermark signals are embedded into the content compared to the embedding of multiple watermark signals into the content with the naive approach. Thus, this approach minimizes the possible degradation of the quality of a digital content due to embedding of watermark signals. Further, since the size of the watermark signal embedded into the content is independent of the number of distributors involved, the quality of the content used by all the consumers will be the same. The experiments performed in Section V clearly show the advantage of the joint watermark mechanism compared to the naive approach. The protocol takes care of the security concerns of the owner, distributors, and consumers. The identity of all the participants is carefully embedded into the content using the Chinese remainder theorem. The identity of all the participants can be determined from the watermark signal by reverse computing the CRT equations. In case the owner or distributors find an unauthorized copy, they can identify the traitors with the help of a judge.

As a future direction of research, the protocols may be improved to reduce the dependence on the license server. Further, in the proposed scheme, the individual watermark information is computed as digital signatures. The protocols can be made more secure by incorporating additional security measures.
more computationally efficient if these are replaced by any other easily verifiable watermark informations.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers for their valuable comments.

REFERENCES


