applied sciences

MDPI

Article

Blockchain and Cloud to Overcome the Problems of Buyer and Seller Watermarking Protocols

Franco Frattolillo

Department of Engineering, University of Sannio, Corso Garibaldi 107, 82100 Benevento, Italy; frattolillo@unisannio.it

Abstract: Copyright protection of digital content has become a problem not only for web content providers but also for ordinary web users who like to publish their digital contents on social or user generated content platforms. Among the possible solutions to such a problem, digital watermarking, in conjunction with watermarking protocols, appears to be a valid alternative to current DRM (digital rights management) systems. In fact, watermarking based solutions insert perceptually invisible copyright information into the copies of contents published or distributed on the web in order to track them. Such insertions are carried out according to the watermarking protocols, which have evolved over the years from the classic "buyer and seller" paradigm into a simpler and versatile "buyer friendly" and "mediated" approach. However, such an approach cannot exploit the new technologies that characterize the current Internet. This paper presents a new watermarking protocol able to adapt the "buyer friendly" and "mediated" approach to the use of innovative technologies such as cloud platforms and blockchain. In this way, (1) content providers and common web users can take advantage of the computing and storage resources made available by cloud platforms; (2) the involvement of trusted third parties in the protocols can be reduced by using blockchain without complicating the protection scheme. In fact, these two goals make the protocol particularly suited for the current Internet.

Keywords: watermarking protocols; digital copyright protection; cloud computing; blockchain

check for updates

Citation: Frattolillo, F. Blockchain and Cloud to Overcome the Problems of Buyer and Seller Watermarking Protocols. *Appl. Sci.* **2021**, *11*, 12028. https://doi.org/10.3390/app112412028

Academic Editors: David Megías, Wojciech Mazurczyk and Minoru Kuribayashi

Received: 1 November 2021
Accepted: 15 December 2021
Published: 17 December 2021

1. Introduction and Motivations

One of the relevant problems of the current Internet is the copyright protection of digital contents distributed or published on the web. In fact, content providers expose economic losses caused by the failure to adequately protect their digital contents, which, if left unprotected, can be easily duplicated, modified, and re-distributed without reducing their perceptual quality, thus damaging their legitimate owners or revealing their private information. Such a problem also affects common web users, who are becoming producers of multimedia digital contents published on social networks or user generated content platforms.

Digital watermarking [1,2] promises to solve the problem of copyright protection. It enables the insertion of an hidden "fingerprint" [3], in the form of a "watermark", into any copy of content sold to a buyer. In this way, if the watermark is generated so as to specify copyright information and to identify the buyer, it can personalize each copy of content sold. Furthermore, if the watermarked copy is released according to the interaction scheme defined by a "watermarking protocol" [4,5], the embedded watermark can be used as a proof of ownership to establish who has initially obtained the copy and then illegally shared it on the Internet.

Even though digital watermarking and watermarking protocols can effectively support digital copyright protection, both are affected by documented problems. In particular, watermark insertion is a specialized and burdensome activity that requires huge computational resources, since it has to be carried out "on-the-fly", when content is sold or published [6,7]. Such a condition represents an obstacle to both content providers and common web users wanting to exploit digital watermarking, since they usually lack the needed computing and storage resources in addition to specific security competence [5,8].

Watermarking protocols, for their part, are characterized by two closely related problems: the presence of trusted third parties (TTPs) needed to enable the protocols, also called "watermark certification authorities" (WCAs) [9,10], and the difficulty of the actions that have to be performed by the entities participating in the protocols [5,11].

The former problem is caused by the collusive behaviours that the entities involved in the protocols could have towards TTPs [2,3], which reduce the security level achieved by the protocols. In fact, a possible solution consists in designing protocols that do not employ TTPs [12–16]. However, the absence of TTPs often entails a complex involvement of buyers, who have to carry out difficult actions to participate in the protocols [5], and this consequence just aggravates the latter problem.

To solve the problems reported above, this paper proposes a new watermarking protocol built around the experiences documented in [5,8,11,17]. The protocol follows the "buyer friendly" and "mediated" approach by which it is possible to restrict the role of TTPs without complicating the involvement of buyers in the interaction scheme. Moreover, the protocol is expressly designed to exploit both blockchain technology [18,19] and cloud computing platforms [20]. In particular, the blockchain technology is introduced to simplify the interaction scheme among the entities participating in the protocol without reducing security, whereas the ability to employ cloud computing platforms enables content providers to outsource a burdensome activity such as the on-the-fly insertion of personalized watermarks.

The result is an innovative watermarking protocol that, unlike previous experiences documented in the literature, is characterized by an original protection scheme able to achieve a right balance between conflicting objectives: protocol security, limited role of the involved TTP, easy participation of buyers in the protocol, ability to resort to cloud computing to implement on-the-fly and personalized watermark insertions into the digital contents to protect.

The outline of the paper is as follows. The next Section reports on related work. The main goals of the protocol are specified in Section 3. In Section 4 the proposed protocol is described in detail, whereas a security analysis of the protocol is provided in Section 5. A first prototype implementation of a platform able to run the proposed protocol is described in Section 6. The main performance aspects of the described platform are reported in Section 7. Section 8 concludes the paper.

2. Related Work

The first generation of the "buyer and seller" watermarking protocols is based on the presence of TTPs acting as WCAs. Although with some differences, the protocols exploit public-key infrastructures (PKIs) based on encryption schemes that are "privacy-homomorphic" with respect to the adopted watermarking algorithms [21]. This enables WCAs to encrypt watermarks with the buyers' public keys. Then, the encrypted watermarks can be directly embedded into the contents encrypted with the same buyers' public keys. As a consequence, buyers can obtain the requested contents protected by personalized watermarks by simply decrypting them with their private keys [9,10]. This makes it possible to solve the "customer's rights problem" [10], since sellers cannot know the final version of the protected contents, which are known only by buyers.

Although rather secure, the protocols belonging to the first generation are affected by a number of problems, such as the "unbinding problem" [10], conspiracy problems caused by collusive WCAs [2], "ambiguity problems" caused by double watermark insertions [22], and lack of "multiple negotiation mechanisms" [5].

A first evolution of the first-generation protocols is represented by the experience documented in [23], which does not employ a WCA, and by the protocols proposed in [24,25], which use off-line WCAs. In fact, all the protocols try to mainly solve conspiracy problems by eliminating TTPs or restricting their role in the protection schemes.

The protocol described in [26] follows a different approach to avoid TTPs. Such an approach also addresses scalability problems by trying to relieve sellers of the burden of having to embed watermarks in the contents to protect. It is known as "client-side embedding", and uses symmetric ciphers and "partial encryption" to enable both seller and buyer to take charge of watermark insertion: the former additively distorts selected transform coefficients of the content with a noise sequence, which is then partially removed by the latter by means of a specific decryption key received by the seller, thus leaving only the watermark [27].

Although innovative, the protocol proposed in [26] is affected by specific problems, such as the "customer's rights problem" and collusion problems, which are all due to the fact that the seller has access to the decryption keys that have to be used to partially remove the noise sequences so as to leave client-specific watermarks in the contents.

To solve the problems affecting the scheme described in [26], the protocols proposed in [15,16] prevent sellers from accessing the decryption keys used by buyers. In particular, the protocol in [15] manages such keys so as to leave a personalized binary fingerprint in each decryption that is unknown to the seller. In addition, the protocol in [16] provides the scheme described in [15] with collusion resistance facilities by adopting two different solutions to generate the fingerprinting codes to be embedded in the decryption keys sent to buyers: the former uses near orthogonal independent Gaussian fingerprints, whereas the latter generates fingerprints according to a Tardos code [28].

Even though the last generation of protocols based on "client-side embedding" is characterized by scalability, collusion resistance, and absence of TTPs, it does not relieve buyers of the burden of performing complex security actions, such as the removal of noise sequences from the purchased contents.

Different approaches are followed by the protocols described in [12,13], which mainly pursue efficiency by means of maximally simplified collusion-resistant interaction schemes which do without TTPs. However, the quest for simplification makes the protocols impractical in the web context, since it forces buyers to carry out complex security actions, such as encryptions, watermark generations, interactive zero-knowledge proofs, and group signatures.

The watermarking protocols reported above can be considered as some of the most relevant examples of how the "buyer and seller" scheme has undergone a constant evolution over time. However, in the literature, there are no significant examples of watermarking protocols able to employ both the blockchain technology and cloud resources. On the contrary, a number of DRM systems exploit blockchain and digital watermarking technologies to implement copyright management services, such as to keep track of content modifications, copyright transfers or other transaction trails related to the managed digital contents [29,30].

The project documented in [31] represents a copyright management system that only involves buyers and sellers, which can directly interact in the purchase transactions without the need of TTPs. The system employs an improved ElGamal encryption algorithm to efficiently protect the keys used in the interaction among buyers and sellers, and exploits a public blockchain to securely record copyright and transaction data. It also resorts to smart contracts and InterPlanetary File System (IPFS): the former are used to manage copyright information and the purchase transactions, whereas the latter stores watermarked contents. Even though the system cleverly implements all the phases needed to digitally protect contents, its authors do not provide any evidence that the system is secure and not affected by the classic "customer's rights problem" or the "unbinding problem".

BMCProtector [32] is an application to protect music copyright based on a public blockchain and smart contracts. In particular, it uses smart contracts registered in the blockchain to share the copyright parameters of the music owners and to automatically pay royalties to copyright owners. Moreover, it uses IPFS as an external storage for watermarked audio files, and exploits the AES algorithm to encrypt these files. Music distribution and usage are managed by an off-chain, classic access control mechanism characterizing DRM systems. Preliminary tests show that BMCProtector can achieve a transaction rate equivalent to that one exhibited by commercial and proprietary DRM systems. However, BMCProtector employs a TTP in the form of a Key Protection Center needed to manage the keys that music owners use to decrypt the audio files without disclosing them to buyers. Furthermore, as with [31], the security of BMCProtector is not proved.

DRMChain [33] is a blockchain based DRM system that implements trusted content protection and conditional traceability of protected content violations. It employs IPFS to provide a flexible storage of protected contents, and implements a number of services, such as an efficient privacy preserving user authentication, content watermarking and encryption, and a license and violation tracing system based on a conditional identity management. Although DRMChain does not represent a proof-of-concept and its performance has been evaluated, security has been proved only for the single implemented services. As a consequence, the proposed system lacks a global security analysis concerning the behavior of DRMChain towards the classic security problems of watermarking protocols.

The examples of the DRM systems reported above show how they mainly attempt to combat the unauthorized use of protected digital contents without payment. On the contrary, most of them cannot prove to be able to counter the illegal sharing of contents legitimately bought by web users [34]. In fact, once contents are downloaded and tampered, the above DRM systems cannot legally prove the ownership of the contents as well as identify those responsible for copyright infringement.

3. Goals of the Protocol and How to Achieve Them

The considerations reported above and the experiences conducted in the last few years suggest that watermarking protocols should meet the following requirements.

First of all, protocols should enable buyers to easily participate in the purchase transactions of contents [5]. In fact, buyers are to be considered common web users not provided with specific security competence. Therefore, they cannot usually generate watermarks, fingerprints, or security tokens, and can perform the only actions that can be executed by common web browsers.

Protocols should enable content providers with different abilities and resources to easily take part in the protection scheme of distributed contents. In fact, content providers may be represented by very different web entities in the current web context. More precisely, a content provider can be a specialized entity provided with both competence and resources to apply an on-the-fly protection to the distributed contents. However, it can be also a common web user who only wants to protect his/her contents published on social platforms. Such a user, differently from specialized content providers, lacks both security competence and resources to protect his/her contents. As a consequence, protocols should enable content providers to eventually exploit “security delegates” to embed personalized watermarks in the distributed or published contents [11].

Protocols should not involve TTPs in the protection scheme, since TTPs could give rise to collusion problems with the other parties involved in the protocols. However, such a goal should be achieved without complicating the interaction schemes or forcing buyers to carry out complex security actions to complete the purchase transactions, thus making protocols impractical for the current Internet.

A starting point to achieve the goals reported above can be represented by watermarking protocols able to combine a “buyer friendly” approach with the capacity to employ means and technologies to respectively: (1) support content providers in embedding personalized watermarks in the distributed contents; (2) simplify the protection scheme by restricting the role played by TTPs without forcing buyers to carry out complex security actions. In this regard, a possible solution can be found in enabling content providers to exploit cloud computing platforms to outsource the burdensome activity represented by the on-the-fly insertion of personalized watermarks. In fact, content providers can opt to pay cloud platforms as a function of their needs without having to defray the costs of adequate, private computing and storage resources. In addition, even ordinary web users can easily become content providers, since they can take advantage of cloud computing platforms to protect their contents without having resources and specific expertise.

Moreover, such a solution can adopt the blockchain technology to limit the key role played by TTPs, thus supporting a decentralized approach for the protocol. In fact, a blockchain is a distributed ledger that can save data identifying the transactions among the entities involved in the protocol in cryptographic, hashed and signed blocks which are then timestamped and linked in a chain by means of a distributed consensus mechanism. The process makes the blocks resilient to modifications. Thus, they can be considered trusted for the transactions managed by the protocol, and can be also verified in a decentralized way by employing a consensus algorithm run by multiple web nodes to establish whether a transaction is valid or not. In addition, a blockchain also supports "smart contracts", which can be used to replace TTPs in performing the centralized actions needed to enable the protocol. More precisely, the smart contracts define preset rules, in the form of code, which are automatically executed when the terms of an agreement are reached between distinct web entities. Such terms are specified as trigger events under specific conditions: when the conditions are met, the code of the smart contracts is run without control from central authorities [35].

4. Watermarking Protocol

The proposed watermarking protocol exploits the blockchain technology and makes it possible to employ cloud computing platforms in a "buyer friendly" and "mediated" context. In particular, the protocol uses smart contracts within a blockchain to automatically verify and validate the security tokens exchanged during protocol transactions without the involvement of a TTP, whose role appears to be restricted to the initial phase of the protocol. If the tokens are correct, they can be locked in the public ledger managed by the blockchain so as to enable the protocol to run without a centralized control. However, the protocol employs a TTP to generate the security tokens needed for the transactions, such as one-time public and private key pairs and encrypted "nonces", so as to keep the protection scheme simple and secure.

Furthermore, the capacity to exploit a cloud computing platform to implement the watermark insertion algorithm enables even ordinary web users, namely users who lack specific expertise in the area of security transactions, to be involved as content providers. In fact, they can take part in the protocol even if they are not provided with the computing and storage resources needed to protect their digital contents, and this can be considered an actual plus also for specialized content providers, which can decide whether to use cloud computing platforms or not.

4.1. Basics of the Protocol

The watermarking protocol exploits the following security facilities: blind and readable watermarking scheme [1], public key infrastructure (PKI), homomorphic cryptosystem [21], and encrypted and signed tokens intended as simple or complex units of information exchanged during the protocol execution [5].

As regards watermark insertion, both content and watermark are assumed to be represented in the block-wise form $X = \{x_1, x_2, \dots x_l\}$ and $W = \{w_1, w_2, \dots w_l\}$, respectively. Therefore, watermark insertion, specified by the symbol $\oplus$, can be calculated as:

$$X \oplus W = \{x_1 \oplus w_1, x_2 \oplus w_2, \dots x_l \oplus w_l\} = \bar{X}$$

since it assumes linear watermarks [1].

The encryption $\mathbb{E}$ is assumed to be a block-wise function [9,10]. Therefore, the encryption of a content $X=\{x_1, x_2 \ldots x_l\}$ under the key k can be calculated as:

$$\mathbb{E}_k(X) = \mathbb{E}_k(x_1, x_2 \ldots x_l) = (\mathbb{E}_k(x_1), \mathbb{E}_k(x_2) \ldots \mathbb{E}_k(x_l))$$

Moreover, $\mathbb{E}$ is "homomorphic" with respect to the watermark insertion [21]. More precisely, $\mathbb{E}$ is homomorphic with respect to an operation $\odot$ if

$$\mathbb{E}_k(m_1 \odot m_2) = \mathbb{E}_k(m_1) \odot \mathbb{E}_k(m_2)$$

for any two plain messages m_1 and m_2. Therefore, the following expression makes it possible to insert any linear watermark directly into the encrypted domain [9,10]

$$\mathbb{E}_k(X \oplus W) = \mathbb{E}_k(X) \oplus \mathbb{E}_k(W) = \mathbb{E}_k(\bar{X})$$

thus being possible to directly work on encrypted data.

Finally, $\mathbb{E}$ is also assumed to be "commutative". As a consequence, it satisfies the following properties [36–39]:

$$\mathbb{E}_{k_1}(\mathbb{E}_{k_2}(m)) = \mathbb{E}_{k_2}(\mathbb{E}_{k_1}(m)) \text{ and } \mathbb{D}_{k_2}(\mathbb{E}_{k_1}(\mathbb{E}_{k_2}(m))) = \mathbb{E}_{k_1}(m)$$

for any two keys k_1 and k_2 and any message m.

4.2. Entities and Subprotocols

The proposed protocol involves the following entities, each one playing a specific role: (1) the content provider $\mathcal{CP}$ releases only encrypted and watermarked contents; (2) the buyer $\mathcal{B}$ decrypts the received content to obtain it in the final, protected form; (3) a blockchain $\mathcal{BC}$ automatically runs smart contracts to validate the purchases of protected digital contents occurring between buyers and content providers; (4) a cloud computing platform $\mathcal{CL}$ can be delegated by the content provider as an infrastructural service provider [20] to implement watermark insertion; (5) a "registration authority" $\mathcal{RA}$ guarantees the security tokens employed in the purchase transactions; (6) a judge $\mathcal{J}$ participates in the dispute resolution protocol to determine if a buyer is guilty of having released pirated copies.

The protocol consists of two subprotocols: the *protection protocol* and the *identification and arbitration protocol*. The symbols used to describe the protocol are specified in Table 1, whereas Table 2 describes the content of the most complex security tokens.

4.3. Protection Protocol

Table 3 shows the protection protocol that is activated when $\mathcal{B}$ visits the $\mathcal{CP}$'s web site to choose the content X. After the choice of X, $\mathcal{B}$ sends the purchase request to $\mathcal{CP}$ in the message m_1 together with B_{id}, which represents the information needed to identify $\mathcal{B}$ according to one of the "negotiation mechanisms" made available by $\mathcal{CP}$ [5]. In fact, $\mathcal{CP}$ provides multiple negotiation mechanisms, each based on the concept of "multilateral security" applied to web transactions [40]. Therefore, $\mathcal{B}$ can be unambiguously identified, for example, by using an anonymous digital certificate or a personal digital certificate or a credit card.

When $\mathcal{CP}$ receives the request, it generates: (1) $X_{d,t}$, which is a complex information consisting of two parts: the former is a string that unambiguously identifies the content X, whereas the latter is a timestamp referred to the ongoing transaction; (2) a one-time key pair ($pk^X_{\mathcal{CP}}$, $sk^X_{\mathcal{CP}}$) to be used only in the current transaction [41]; (3) the watermark W, which is a fingerprinting binary string that concatenates an anti-collusion code and a redundant code needed to address the problem of bit errors resulting from the watermark extraction process [3]. Then, $\mathcal{CP}$ sends $\mathcal{RA}$ the message m_2 including $X_{d,t}$, $pk^X_{\mathcal{CP}}$, and B_{id}.

Table 1. The symbols used to describe the proposed protocol.

Symbol	Meaning
$\mathcal{B}$	buyer
$\mathcal{CP}$	content provider or seller
$\mathcal{RA}$	registration authority
$\mathcal{CL}$	cloud computing platform
$\mathcal{BC}$	blockchain
$\mathcal{J}$	judge
X	digital content purchased by $\mathcal{B}$
$X_{d,t}$	information used by $\mathcal{CP}$ to unambiguously identify X and the current transaction
N	nonce used to mark the watermarking transaction
W	watermark
$\bar{X}$	watermarked X
$\bar{\bar{X}}$	double watermarked X
B_{id}	information identifying $\mathcal{B}$
B_p	payment information related to $\mathcal{B}$
CP_p	information needed to pay $\mathcal{CP}$
$pk_{Ent.}$	public key of the entity $Ent.$
$sk_{Ent.}$	secret key of the entity $Ent.$
$pk^X_{Ent.}$	one time public key generated by the entity $Ent.$ in the transaction to watermark X
$sk^X_{Ent.}$	one time secret key generated by the entity $Ent.$ in the transaction to watermark X
$E_{key}(\ldots)$	token encrypted by using the key key and a public key cryptosystem
$H_{Ent.}$	digest calculated on the secret key of the entity $Ent.$ by applying the SHA-1 secure hash algorithm
$\mathbb{S}_{Ent.}(\ldots)$	token digitally signed by using the secret key of the entity $Ent.$ and the SHA-1 secure hash algorithm
$\mathbb{E}_{key}(\ldots)$	token encrypted by using the key key and a cryptosystem that is privacy homomorphic and commutative with respect to the watermark insertion
$\mathbb{D}_{key}(\ldots)$	decryption function corresponding to the encryption function $\mathbb{E}_{key}(\ldots)$

Table 2. Complex tokens used by the proposed protocol.

Complex Token	Content
$\mathbb{S}_{\mathcal{RA}}$	$\mathbb{S}_{\mathcal{RA}}(X_{d,t}, pk^X_{\mathcal{B}}, pk^X_{\mathcal{CP}}, \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), E_{\mathcal{RA}})$
$E_{\mathcal{RA}}$	$E_{pk^X_{\mathcal{RA}}}(X_{d,t}, pk^X_{\mathcal{B}}, sk^X_{\mathcal{B}}, pk^X_{\mathcal{CP}}, \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), N, B_{id}, H_{\mathcal{RA}})$

Table 3. Protection protocol.

$\mathcal{B} \rightarrow \mathcal{CP}$	:	$m_1 = \{$request for the content X, $B_{id}\}$
$\mathcal{CP}$	:	generates $X_{d,t}$, $(pk^X_{\mathcal{CP}}, sk^X_{\mathcal{CP}})$, and W
$\mathcal{CP} \rightarrow \mathcal{RA}$	:	$m_2 = \{X_{d,t}, pk^X_{\mathcal{CP}}, B_{id}\}$
$\mathcal{RA}$	:	generates $(pk^X_{\mathcal{B}}, sk^X_{\mathcal{B}})$ and N, and calculates $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$ and $E_{\mathcal{RA}}$
$\mathcal{RA} \rightarrow \mathcal{CP}$	:	$m_3 = \{X_{d,t}, pk^X_{\mathcal{B}}, \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}\}$
$\mathcal{RA} \rightarrow \mathcal{B}$	:	$m_4 = \{X_{d,t}, pk^X_{\mathcal{B}}, \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}\}$
$\mathcal{CP}$	:	calculates $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(W))$ and $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(X))$
$\mathcal{CP} \rightarrow \mathcal{CL}$	:	$m_5 = \{\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(W)), \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(X))\}$
$\mathcal{CL}$	:	calculates $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(\bar{X})) = \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(X \oplus W)) =$
		$= \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(X)) \oplus \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(W))$
$\mathcal{CL}$	:	calculates $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(\bar{\bar{X}})) = \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(\bar{X} \oplus N)) =$
		$= \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(\bar{X})) \oplus \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$
$\mathcal{CL} \rightarrow \mathcal{CP}$	:	$m_6 = \{\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(\bar{\bar{X}}))\}$
$\mathcal{CP} \rightarrow \mathcal{B}$	:	$m_7 = \{$request for payment$\}$
$\mathcal{CP} \rightarrow \mathcal{BC}$	:	$m_8 = \{X_{d,t}, pk^X_{\mathcal{CP}}, \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}, CP_p\}$
$\mathcal{B} \rightarrow \mathcal{BC}$	:	$m_9 = \{X_{d,t}, pk^X_{\mathcal{B}}, \mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}, B_p\}$
$\mathcal{BC}$	:	runs the smart contract

Table 3. *Cont.*

$\mathcal{BC}$	:	compares the tokens and verifies the signatures included in m_8 and m_9
$\mathcal{BC}$	:	creates a node in the blockchain by which to publish $X_{d,t}$, $pk_{\mathcal{B}}^X$, $pk_{\mathcal{CP}}^X$, $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, $E_{\mathcal{RA}}$, $\mathbb{S}_{\mathcal{RA}}$
$\mathcal{BC}$	:	implements the payment phase
$\mathcal{CP}$	:	generates $\mathbb{E}_{pk_{\mathcal{B}}^X}(\bar{\bar{X}}) = \mathbb{D}_{sk_{\mathcal{CP}}^X}(\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(\bar{\bar{X}})))$
$\mathcal{CP} \rightarrow \mathcal{B}$	:	$m_{10} = \{\mathbb{E}_{pk_{\mathcal{B}}^X}(\bar{\bar{X}})\}$
$\mathcal{CP}/\mathcal{CL}$	:	saves a new entry in its databases including $X_{d,t}$, $pk_{\mathcal{B}}^X$, $pk_{\mathcal{CP}}^X$, $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, $E_{\mathcal{RA}}$, $\mathbb{S}_{\mathcal{RA}}$ whose search key is W
$\mathcal{BC} \rightarrow \mathcal{RA}$	:	$m_{11} = \{X_{d,t}, pk_{\mathcal{B}}^X, pk_{\mathcal{CP}}^X, \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}\}$
$\mathcal{RA} \rightarrow \mathcal{B}$	:	$m_{12} = \{sk_{\mathcal{B}}^X\}$
$\mathcal{B}$	:	generates $\bar{\bar{X}} = \mathbb{D}_{sk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{B}}^X}(\bar{\bar{X}}))$

$\mathcal{RA}$ is a TTP. It is responsible for generating the one-time key pair ($pk_{\mathcal{B}}^X$, $sk_{\mathcal{B}}^X$) that makes it possible to create the double encrypted token $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, which will be part of the watermark to be used to protect the content X. In particular, N is a "nonce" doubly encrypted by employing a cryptosystem that is "privacy homomorphic" and "commutative" with respect to the subsequent watermark insertion, as reported in Section 4.1. $\mathcal{RA}$ also generates the encrypted token $E_{\mathcal{RA}}$, whose composition is shown in Table 2. In particular, $E_{\mathcal{RA}}$ is encrypted with the public key of $\mathcal{RA}$ and contains $X_{d,t}$, $pk_{\mathcal{B}}^X$, $sk_{\mathcal{B}}^X$, $pk_{\mathcal{CP}}^X$, $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, N, B_{id}, and $H_{\mathcal{RA}}$, which is the digest calculated on the secret key of $\mathcal{RA}$ by applying the SHA-1 secure hash algorithm. As a consequence, $E_{\mathcal{RA}}$ can be generated only by $\mathcal{RA}$.

$X_{d,t}$, $pk_{\mathcal{B}}^X$, and $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$ are sent by $\mathcal{RA}$ to $\mathcal{CP}$ and $\mathcal{B}$ in the messages m_3 and m_4 together with $E_{\mathcal{RA}}$ and the signature $\mathbb{S}_{RA}$, which, as shown in Table 2, is calculated on the following tokens: $X_{d,t}$, $pk_{\mathcal{B}}^X$, $pk_{\mathcal{CP}}^X$, $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, and $E_{\mathcal{RA}}$.

Upon receipt of the message m_3, $\mathcal{CP}$ checks the signature $\mathbb{S}_{RA}$ and starts the interaction with $\mathcal{CL}$ to protect X. It doubly encrypts X and W by using the one-time keys $pk_{\mathcal{CP}}^X$ and $pk_{\mathcal{B}}^X$, thus generating $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(X))$ and $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(W))$, which are inaccessible to $\mathcal{CL}$. Then, $\mathcal{CP}$ sends $\mathcal{CL}$ the message m_5, which includes $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(X))$ and $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(W))$.

When it receives the message m_5, $\mathcal{CL}$ can embed the watermark W directly into the encrypted domain due to the homomorphism of the encryption scheme:

$$\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(\bar{X})) = \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(X \oplus W)) = \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(X)) \oplus \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(W))$$

Likewise, $\mathcal{CL}$ repeats the operation reported above to carry out the second watermark insertion, thus embedding $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$ into X in the encrypted domain:

$$\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(\bar{\bar{X}})) = \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(\bar{X} \oplus N)) = \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(\bar{X})) \oplus \mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$$

After receipt of the message m_6 from $\mathcal{CL}$, $\mathcal{CP}$ can demand for payment by sending the message m_7 to $\mathcal{B}$.

At this point, $\mathcal{CP}$ and $\mathcal{B}$ send the messages m_8 and m_9 to run the smart contract in the blockchain infrastructure $\mathcal{BC}$. In particular, both messages m_8 and m_9 contain $X_{d,t}$, $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$. In addition, m_8 includes $pk_{\mathcal{CP}}^X$ and CP_p, whereas m_9 includes $pk_{\mathcal{B}}^X$ and B_p. In this way, the miners of $\mathcal{BC}$ have all the elements to execute the smart contract, whose main task consists in comparing the tokens and verifying the signature $\mathbb{S}_{\mathcal{RA}}$ contained in the received messages. Moreover, the smart contract checks whether $pk_{\mathcal{B}}^X$ and $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$, generated by $\mathcal{RA}$, have been already used in a previous purchase transaction or not. This means to check whether $pk_{\mathcal{B}}^X$ and $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$ have been already published in a block of the blockchain or not. If all the checks are successfully passed, the code of the smart contract validates the purchase transaction, and the tokens identifying the ongoing transaction, such as $X_{d,t}$, $pk_{\mathcal{B}}^{X}$, $pk_{\mathcal{CP}}^{X}$, $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$ are published in a block of $\mathcal{BC}$. Then, the smart contract implements the payment phase by employing CP_p and B_p, and ends by sending the message m_{11} to $\mathcal{RA}$.

If the payment phase is successful, $\mathcal{CP}$ can use its private key $sk_{\mathcal{CP}}^{X}$ to decrypt $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(\bar{\bar{X}}))$, thus generating $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\bar{\bar{X}}) = \mathbb{D}_{sk_{\mathcal{CP}}^{X}}(\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(\bar{\bar{X}})))$. Then, $\mathcal{CP}$ sends $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\bar{\bar{X}})$ to $\mathcal{B}$ in the message m_{10}, and stores a new entry in its databases, which can be also managed by $\mathcal{CL}$ on behalf of $\mathcal{CP}$. In particular, the entry includes $X_{d,t}$, $pk_{\mathcal{B}}^{X}$, $pk_{\mathcal{CP}}^{X}$, $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$. It can be retrieved by $\mathcal{CP}$, or by $\mathcal{CL}$, by using the watermark W as search key. In fact, all the tokens saved by $\mathcal{CP}$ are needed to prove that $\mathcal{B}$ is the legitimate owner of the protected content $\bar{\bar{X}}$ sold by $\mathcal{CP}$ through a transaction registered by a node published in the blockchain $\mathcal{BC}$.

The message m_{11} allows $\mathcal{RA}$ to send the secret key $sk_{\mathcal{B}}^{X}$ to $\mathcal{B}$ in the message m_{12}. $\mathcal{B}$ can thus use the received key to decrypt $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\bar{\bar{X}})$ and obtain $\bar{\bar{X}} = \mathbb{D}_{sk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\bar{\bar{X}})$, which represents the final version of the purchased protected content.

4.4. Identification and Arbitration Protocol

Table 4 shows the protocol that is run to identify, with undeniable evidence, the legitimate copyright owner of the protected content $\bar{\bar{X}}$ that was illegally shared in the form of pirated copy [5]. The protocol is run by $\mathcal{CP}$ when such a copy is found in the market.

Table 4. Identification and arbitration protocol.

$\mathcal{CP}$	:	finds X' in the market
$\mathcal{CP}/\mathcal{CL}$	:	extracts W' and N', and searches databases for a possible match on W'
$\mathcal{CP} \rightarrow \mathcal{J}$	:	$m_1 = \{N', X_{d,t}, pk_{\mathcal{B}}^{X}, pk_{\mathcal{CP}}^{X}, \mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}\}$
$\mathcal{J}$	:	searches $\mathcal{BC}$ for a node containing the tokens included into m_1 except N'
$\mathcal{J}$	:	retrieves the tokens published in the node of $\mathcal{BC}$, which are $X_{d,t}$, $pk_{\mathcal{B}}^{X}$, $pk_{\mathcal{CP}}^{X}$, $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N))$, $E_{\mathcal{RA}}$, $\mathbb{S}_{\mathcal{RA}}$
$\mathcal{J}$	:	verifies that the tokens retrieved from $\mathcal{BC}$ match those ones received from $\mathcal{CP}$
$\mathcal{J} \rightarrow \mathcal{RA}$	:	$m_2 = \{X_{d,t}, pk_{\mathcal{B}}^{X}, pk_{\mathcal{CP}}^{X}, \mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N)), E_{\mathcal{RA}}, \mathbb{S}_{\mathcal{RA}}\}$
$\mathcal{RA}$	:	verifies the received tokens, decrypts $E_{\mathcal{RA}}$, and obtains N
$\mathcal{RA} \rightarrow \mathcal{J}$	:	$m_3 = \{B_{id}, N\}$
$\mathcal{J}$	:	compares N' with N and adjudicates

The protocol starts with the extraction of the watermarks W' and N' from the pirated copy of $\bar{\bar{X}}$, denoted as X'. The extraction of W', which can be also carried out by $\mathcal{CL}$ on behalf of $\mathcal{CP}$, makes it possible to search databases for a possible match. If it is found, $\mathcal{CP}$ can retrieve the associated tokens corresponding to the purchase transaction of $\bar{\bar{X}}$: $X_{d,t}$, $pk_{\mathcal{B}}^{X}$, $pk_{\mathcal{CP}}^{X}$, $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$. The tokens can thus be sent to $\mathcal{J}$ in the message m_1 together with N'.

The message m_1 enables $\mathcal{J}$ to search the blockchain $\mathcal{BC}$ for a node that publishes the tokens received from $\mathcal{CP}$. If a node is found, $\mathcal{J}$ can retrieve the published tokens, which are reported in Table 3. Then, $\mathcal{J}$ compares the tokens published by $\mathcal{BC}$ with those ones received by $\mathcal{CP}$, and, if they match, $\mathcal{J}$ can send $\mathcal{RA}$ the message m_2, which includes $X_{d,t}$, $pk_{\mathcal{B}}^{X}$, $pk_{\mathcal{CP}}^{X}$, $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$.

$\mathcal{RA}$ decrypts $E_{\mathcal{RA}}$ and compares the extracted tokens with those ones received from $\mathcal{J}$. If the tokens match, $\mathcal{RA}$ also decrypts $\mathbb{E}_{pk_{\mathcal{B}}^{X}}(\mathbb{E}_{pk_{\mathcal{CP}}^{X}}(N))$ and obtains N. Then, $\mathcal{RA}$ sends $\mathcal{J}$ the message m_3, which includes B_{id} and N.

The message m_3 enables $\mathcal{J}$ to compare N' and N. If $N' = N$, $\mathcal{J}$ can close the case adjudicating the buyer identified by B_{id} to be a traitor. Otherwise, the protocol ends without exposing any identity.

5. Security Analysis

The conducted analysis is based on the following ideal behaviour of the proposed watermarking protocol: $\mathcal{CP}$ sells X to $\mathcal{B}$; $\mathcal{B}$ gets the protected content $\bar{\bar{X}}$ from $\mathcal{CP}$; $\mathcal{CL}$ embeds watermarks into the contents sold or published by $\mathcal{CP}$; $\mathcal{BC}$ is a ledger that registers the purchase transactions of digital contents sold or published by $\mathcal{CP}$; $\mathcal{RA}$ generates data needed to protect X; $\mathcal{J}$ decides whether $\mathcal{B}$ is guilty of releasing pirated copies.

The analysis assumes that:

- $\mathcal{J}$ and $\mathcal{RA}$ cannot be corrupted since they are TTPs.
- $\mathcal{CP}$ and $\mathcal{B}$ can be corrupted only "statically", i.e., the corrupt entities are decided before the execution of the protocol and cannot be modified throughout the execution [42].
- $\mathcal{BC}$ and $\mathcal{CL}$ are characterized by an "honest-but-curious" behaviour [42]. They have to follow the rules of the protocol, even though they can try their best to get information from the executed actions. As a consequence, $\mathcal{BC}$ has to correctly manage the blockchain, whereas $\mathcal{CL}$ has to correctly apply the received watermarks. In fact, these assumptions are reasonable, since $\mathcal{BC}$ has only to run smart contracts whose code is accepted in advance and cannot be modified during the life of the blockchain [19]. Similarly, $\mathcal{CL}$ acts like a delegate that supplies security web services on behalf of $\mathcal{CP}$. Therefore, if $\mathcal{CL}$ cannot ensure high security standards, its reputation can be compromised as well as its business possibilities, which ultimately depend on the capability to actually protect the contents distributed by $\mathcal{CP}$.
- Uncorrupt buyers and content providers do not release pirated copies.

With these assumptions, the protocol ensures that:

- If $\mathcal{CP}$ and $\mathcal{B}$ are uncorrupt, $\mathcal{B}$ receives a unique and personalised protected content $\bar{\bar{X}}$. Furthermore, pirated copies of $\bar{\bar{X}}$ found on the web can be traced back to the original buyer and purchase transaction.
- If $\mathcal{CP}$ is corrupt, $\mathcal{B}$ receives a content $\bar{\bar{X}}$ that prevents the identification of traitors. In fact, such a corruption is useless and deleterious just for $\mathcal{CP}$.
- If $\mathcal{B}$ is corrupt, $\mathcal{CP}$ aborts the purchase transaction and does not release any content.

5.1. Basic Requirements

The watermark insertion technique adopted to protect digital contents has to resist both nonmalevolent manipulations and intentional attacks [43]. Well-known examples of such a technique can be found in [1,27,44–46].

The digital encryption implementing the PKI has to ensure indistinguishably under chosen plaintext attack (IND-CPA). Therefore, any knowledge about a plaintext message m cannot be get from the corresponding ciphertext c.

Finally, the well-known SSL/TLS protocol is used to protect communications among the web entities involved in the protocol.

5.2. Analysis

The security analysis has been developed on the basis of a simplified proof scheme derived from what is reported in [5,8,11,17]. According to such a scheme, a watermarking protocol consists of entities involved in the protocol and of tokens exchanged among such entities.

Entities can be classified into two categories on the basis of the role they play during the protocol execution:"honest" entities and "corrupt" entities. As reported in Section 5, such a role is determined "statically", i.e., honest and corrupt entities are decided before the execution of the protocol and cannot be modified throughout the execution. More precisely, honest entities correctly generate tokens without altering them during protocol executions, follow the protocol without performing cheating actions, and reveal possible mismatches and attacks. On the contrary, corrupt entities try to attack the protocol particularly by generating corrupt tokens or maliciously altering them during protocol executions.

The description of the protocol behaviour in terms of entities and tokens suggests that a security proof can be obtained (1) by building the paths followed by the tokens through the entities involved in the protocol, and (2) by proving that every malicious intervention on the exchanged tokens made by the corrupt entities is always revealed by the honest entities. This is because any attack on the protocol has the effect of maliciously creating, changing, or eliminating tokens. As a consequence, a watermarking protocol can be proved to be secure if honest entities can always reveal attacks and malicious interventions on the tokens during every execution of the protocol [5,11].

On the basis of the consideration reported above, the security analysis can be conducted by examining the behaviour of the proposed watermarking protocol in the two cases of interest that can be derived from the assumptions characterizing the protocol: (1) $\mathcal{CP}$ is corrupt and tries to cheat $\mathcal{B}$; (2) $\mathcal{B}$ is corrupt and attempts to cheat $\mathcal{CP}$. In both cases, $\mathcal{BC}$ and $\mathcal{CL}$ behave like honest-but-curious entities.

5.2.1. $\mathcal{CP}$ Is Corrupt

Suppose that $\mathcal{CP}$ wants to cheat $\mathcal{B}$. Preliminarily, it is assumed that B_{id} cannot be altered since the current analysis does not address the problem of fake identities. Moreover, it is useless to consider the possible changes to $X_{d,t}$, W and the key pair $(pk^X_{\mathcal{CP}}, sk^X_{\mathcal{CP}})$, since they are still generated by $\mathcal{CP}$. Therefore, the only tokens that $\mathcal{CP}$ can try to alter to cheat $\mathcal{B}$ are $pk^X_{\mathcal{B}}$, $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$, $E_{\mathcal{RA}}$, and $\mathbb{S}_{\mathcal{RA}}$.

The first token to consider is the key $pk^X_{\mathcal{B}}$. Its path among the entities involved in the protocol is shown in Figure 1a, where the corrupt entity is draw as a circle. In particular, $pk^X_{\mathcal{B}}$ is generated by the trusted entity $\mathcal{RA}$ and is initially received by $\mathcal{CP}$ and $\mathcal{B}$ in the messages m_3 and m_4. Then, it is sent to $\mathcal{BC}$ by $\mathcal{B}$ in the message m_9. Finally, $pk^X_{\mathcal{B}}$ is returned to $\mathcal{RA}$ in the message m_{11} (see Table 3).

Suppose that $\mathcal{CP}$ wants to alter the key $pk^X_{\mathcal{B}}$ used to protect X, thus generating the corrupt key $pk^{X^c}_{\mathcal{B}}$. Such an action succeeds only if the check carried out by $\mathcal{BC}$ after the reception of the messages m_8 and m_9 is passed (see also Table 3 and Figure 1a). In fact, the tokens contained in the two messages enable $\mathcal{BC}$ to verify the signature $\mathbb{S}_{\mathcal{RA}}$, which is calculated on $X_{d,t}$, $pk^X_{\mathcal{B}}$, $pk^X_{\mathcal{CP}}$, $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$, and $E_{\mathcal{RA}}$. Since $\mathcal{BC}$ receives $pk^X_{\mathcal{B}}$ only from $\mathcal{B}$, who in turn receives this key from $\mathcal{RA}$, $\mathcal{CP}$ can replace $pk^X_{\mathcal{B}}$ with $pk^{X^c}_{\mathcal{B}}$ only if it can generate the corrupt tokens $\mathbb{S}^c_{\mathcal{RA}}$ and $E^c_{\mathcal{RA}}$, which are to be both consistent with $pk^{X^c}_{\mathcal{B}}$. However, assuming that the key pair $(pk_{\mathcal{RA}}, sk_{\mathcal{RA}})$ is not impaired, $\mathcal{CP}$ cannot generate these tokens.

Furthermore, as reported in Section 4.3, the protocol prevents the reuse of the tokens previously published in the blocks of the blockchain, which are just $X_{d,t}$, $pk^X_{\mathcal{B}}$, $pk^X_{\mathcal{CP}}$, $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$. In fact, any transaction can successfully complete only if the tokens included in the messages m_8, m_9, and m_{11} are consistent, have not been already published in the blocks of the blockchain, and have been directly generated or initially received by $\mathcal{RA}$ in the message m_2.

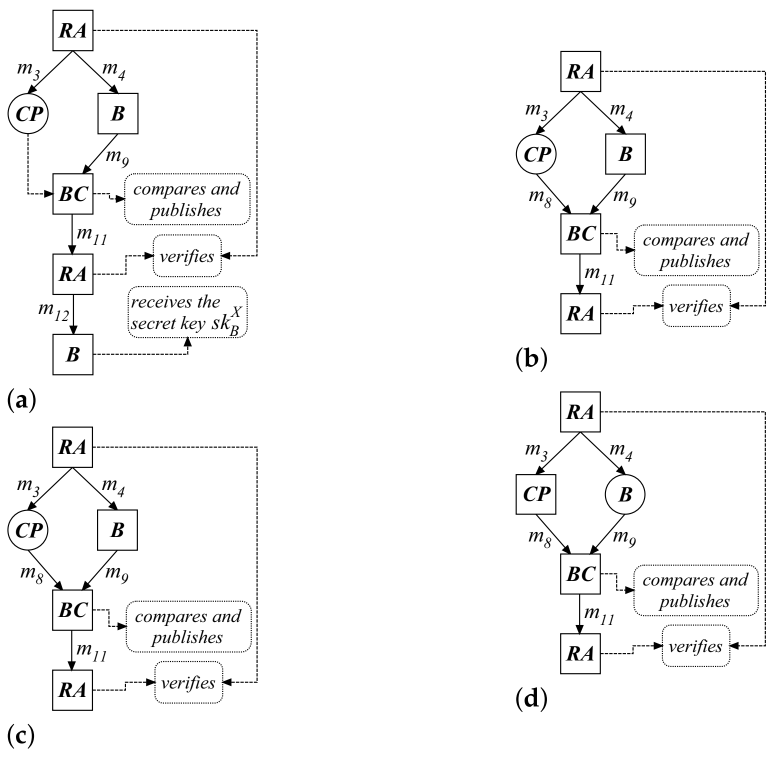

Figure 1. The paths of $pk_{\mathcal{B}}^X$ (**a**), $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$ (**b**), and $\mathbb{S}_{\mathcal{RA}}$ and $E_{\mathcal{RA}}$ (**c**) when $\mathcal{CP}$ is corrupt. The path of all tokens when $\mathcal{B}$ is corrupt (**d**).

More precisely, the protocol is made secure by three main properties:

1. It is impossible to reuse the tokens generated in a successfully completed purchase transaction since, in this hypothesis, they are already published in a block of the blockchain.
2. $\mathcal{CP}$ cannot force the use of specific or arbitrary tokens during a purchase transaction, since only the tokens initially generated by $\mathcal{RA}$ and sent to $\mathcal{B}$ in the message m_4 can be used during the ongoing transaction. These tokens are then sent by $\mathcal{B}$ to $\mathcal{BC}$ in the message m_9, and forwarded to $\mathcal{RA}$ in the message m_{11}, along a path over which $\mathcal{CP}$ has no control (see Figure 1a).
3. A purchase transaction can complete only if all the tokens published in the corresponding block of the blockchain are consistent. This is another reason why $\mathcal{CP}$ cannot force the use of specific or arbitrary tokens, since it cannot generate $\mathbb{S}_{\mathcal{RA}}$ and $E_{\mathcal{RA}}$ corresponding to arbitrary tokens.

The same reasoning can be applied to $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$. More precisely, the properties reported above prevent $\mathcal{CP}$ from forcing the use of a specific or arbitrary watermark during purchase transactions. In fact, the check carried out by $\mathcal{BC}$ and $\mathcal{RA}$ is passed only if the tokens contained in the messages m_8, m_9, and m_{11} are consistent, come from $\mathcal{RA}$, and have not been already published in the blocks of the blockchain (see Table 3 and Figure 1b). In particular, since the token $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$ sent by $\mathcal{B}$ to $\mathcal{BC}$ is the one directly received by $\mathcal{B}$ from $\mathcal{RA}$, a transaction can complete only if the token $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N))$ is just the one generated by $\mathcal{RA}$ at the start of the transaction. Moreover, the above-mentioned properties do not allow $\mathcal{CP}$ to generate or obtain the generation of the tokens $\mathbb{S}_{\mathcal{RA}}^c$ and $E_{\mathcal{RA}}^c$ corresponding to a corrupt $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N^c))$. As a consequence, any insertion of a corrupt watermark into X generates an incorrectly protected content that cannot be traced back to any buyer, since $\mathcal{CP}$ cannot obtain the generation of a valid block in the blockchain able to contain tokens consistent with $\mathbb{E}_{pk_{\mathcal{B}}^X}(\mathbb{E}_{pk_{\mathcal{CP}}^X}(N^c))$.

The consideration reported above can be extended to the tokens $\mathbb{S}_{\mathcal{RA}}$ and $E_{\mathcal{RA}}$, whose path is shown in Figure 1c. In fact, if the keys $(pk_{\mathcal{RA}}, sk_{\mathcal{RA}})$ are not impaired, these tokens cannot be arbitrarily generated by $\mathcal{CP}$. Moreover, as with the other tokens, $\mathbb{S}_{\mathcal{RA}}$ and $E_{\mathcal{RA}}$ can pass the check carried out by $\mathcal{BC}$ and $\mathcal{RA}$ only if they are coherently contained in the messages m_8, m_9, and m_{11}, come from $\mathcal{RA}$, and have not been already published in the blocks of the blockchain (see Table 3 and Figure 1c). As a consequence, any reuse is not allowed.

The proof scheme used for the tokens $pk^X_{\mathcal{B}}$, $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$ demonstrates that $\mathcal{CP}$ cannot alter them without being detected. In fact, the checks carried out by $\mathcal{BC}$ and $\mathcal{RA}$ can be passed only if the tokens sent by $\mathcal{CP}$ in the message m_8 during a purchase transaction: (1) are generated by $\mathcal{RA}$ for the transaction; (2) are consistent with those ones included in the messages m_9 and m_{11}; (3) have not yet been used in previous purchase transactions. Therefore, $\mathcal{CP}$ cannot cheat $\mathcal{B}$ by generating or reusing corrupt keys or tokens, since both $\mathcal{BC}$ and $\mathcal{RA}$ can always check the received keys and tokens and decide to abort the protocol execution or not. As a consequence, the proposed protocol is secure against attacks from $\mathcal{CP}$ since it either can reveal such attacks or ends with correctly protected contents.

5.2.2. $\mathcal{B}$ Is Corrupt

Suppose that $\mathcal{B}$ wants to cheat $\mathcal{CP}$. Excluding problems of fake identities, $\mathcal{B}$ can attempt to alter only the tokens received from $\mathcal{RA}$ in the message m_4, which are: $X_{d,t}$, $pk^X_{\mathcal{B}}$, $\mathbb{E}_{pk^X_{\mathcal{B}}}(\mathbb{E}_{pk^X_{\mathcal{CP}}}(N))$, $\mathbb{S}_{\mathcal{RA}}$, and $E_{\mathcal{RA}}$. However, as shown in Figure 1d, the same tokens are also received by $\mathcal{CP}$ in the message m_3.

Moreover, the tokens contained in the messages m_3 and m_4 are then received by $\mathcal{BC}$ in the messages m_8 and m_9. They can be thus compared and verified by both $\mathcal{BC}$ and $\mathcal{RA}$. Therefore, only if all the received tokens match, the ongoing transaction can complete. This also means that only the tokens initially generated by $\mathcal{RA}$, and correctly forwarded by $\mathcal{CP}$, can be accepted by $\mathcal{BC}$ in the comparison with the tokens received from $\mathcal{B}$. In fact, such a condition prevents $\mathcal{B}$ from cheating $\mathcal{CP}$ by corrupting the tokens generated by $\mathcal{RA}$ during a purchase transaction, since any alteration of the tokens is always disclosed by $\mathcal{BC}$, and causes the protocol to abort without releasing any protected content.

6. Prototype Platform

This Section describes a prototype platform able to implement the proposed watermarking protocol. Figure 2 shows its architecture and focuses on the entities involved in the protocol and on the interactions among them.

$\mathcal{CL}$ represents the cloud computing platform exploited as an “Infrastructure as a Service” (IaaS) [20,47]. It interfaces with the other entities of the protocol through the *master node*, which also implements all the services needed to dynamically manage the needed computing power.

$\mathcal{B}$ is a user provided with a common web browser that can display HTML5 pages, install plug-ins, and run JavaScripts. As shown in Figure 2, $\mathcal{B}$ initially contacts the *Transaction Manager*, which publishes the content catalogue of $\mathcal{CP}$ and receives the purchase requests from buyers. Then, the *Transaction Manager* interacts with *Nprocess*, which generates the tokens that link $\mathcal{B}$ and $\mathcal{CP}$ to the chosen content and the current transaction. The tokens are also communicated to *Pprocess*, which manages the next protection phase.

The *Transaction Manager* also involves $\mathcal{RA}$, which generates its security tokens and returns them to $\mathcal{B}$ and $\mathcal{CP}$. The tokens are received by the *Transaction Manager* and forwarded to *Pprocess*, which retrieves the information previously sent by *Nprocess* and contacts the *Handler* to start the protection process.

The *Handler* involves the cloud computing platform in the watermarking process by contacting the *Interface Service*, which receives all the data needed to watermark the chosen content.

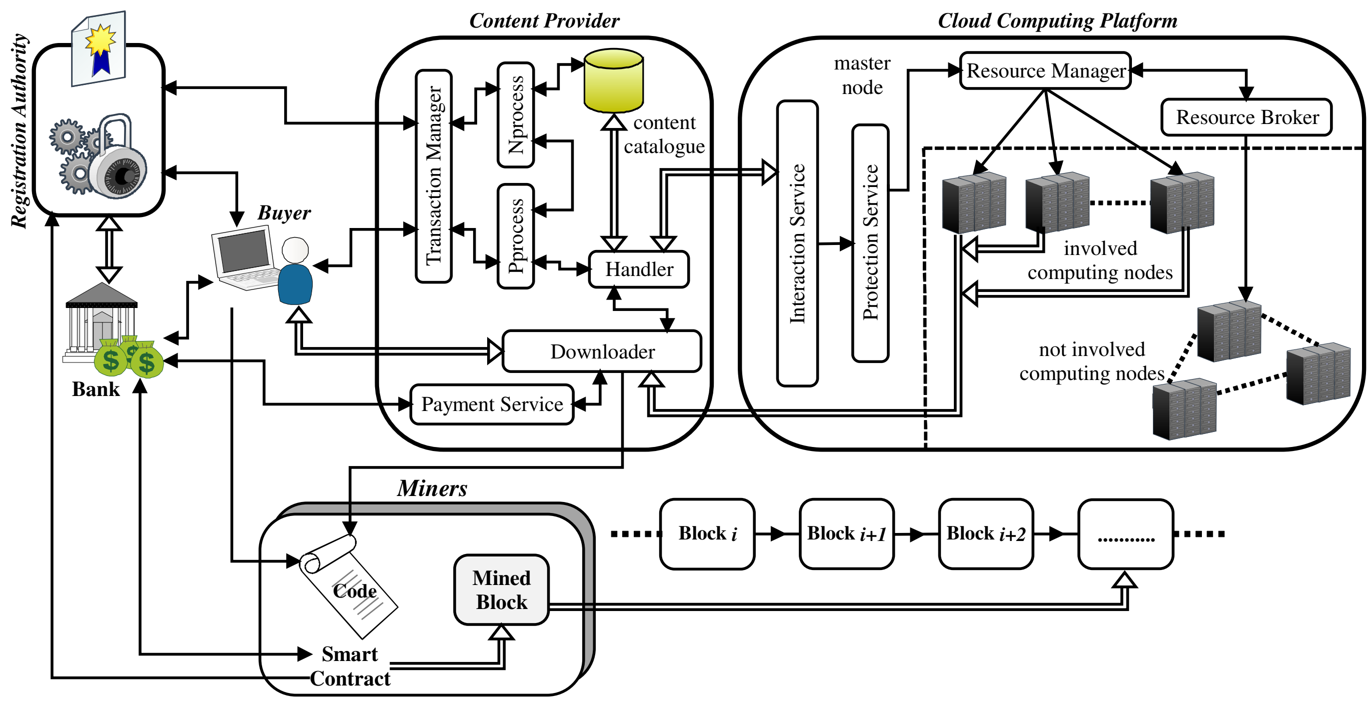

Figure 2. Prototype platform implementing the proposed watermarking protocol.

The *Interface Service* is the entry point of the cloud computing platform. It runs on the *master node* (see Figure 2), and creates a "job" that encapsulates the content to protect. The job is sent to the *Protection Service*, which manages the watermarking process by contacting the *Resource Manager*.

The *Resource Manager* manages the pool of computing nodes supplied by the cloud computing platform. When it receives a job, it chooses the less loaded computing node among those ones initially available within the platform. If all the available nodes are overloaded, the *Resource Manager* contacts the *Resource Broker*, which can demand further computing nodes from the cloud computing platform.

After the watermark insertion, the protected content reaches the *Downloader*, which runs on $\mathcal{CP}$. The *Downloader* activates $\mathcal{BC}$ and requires the payment by contacting the *Payment Service*.

$\mathcal{BC}$ is a "public" blockchain characterized by a fully decentralized architecture based on the "proof of work" consensus algorithm [48]. It runs smart contracts and, if the mining process is successful, the payment takes place. After the payment, the *Downloader* enables the protected content to be downloaded by $\mathcal{B}$, who can thus generate the final version of the watermarked content.

7. Implementation and Performance

In the platform presented in Section 6, the entities of the protocol are implemented as C++ programs according to the experiences documented in [5,8,17]. The programs run on Linux operating system and communicate via OpenSSL standard socket library. They use the NTL and GNU Multi Precision Arithmetic libraries to implement the encryption/decryption and watermark insertion procedures. The cloud computing platform runs the release "Pike" of the OpenStack operating system, and executes the necessary processes interacting through the functions belonging to the API of OpenStack. The blockchain is implemented in Ethereum, whereas the smart contract is coded in Solidity.

The watermark insertion is carried out according to the spread-spectrum technique described in [1] and represented by the formula:

$$\bar{x} = x + \alpha(2b-1)s$$

where x is a host signal feature obtained by calculating the cosine discrete transform, $\bar{x}$ is the corresponding watermarked feature, $b \in \{0, 1\}$ is the bit to embed, s is the component of a spreading sequence, and α is a scaling factor able to control the watermark strength. Such a formula can be directly translated into the encrypted domain due to the homomorphism of the adopted cryptosystem, thus obtaining:

$$E[\bar{x}] = E[x] \cdot E[b]^{2\alpha s} \cdot E[\alpha s]^{-1}$$

Moreover, in the first prototype implementation, the commutative behaviour of the cryptosystem $\mathbb{E}$ is obtained by applying a variant of the well-known ElGamal algorithm [36–39,49,50], which is well-described in [17]. However, such a cryptosystem has been adopted only to prove the correctness and feasibility of the proposed protocol, since it is not IND-CPA secure. In fact, the cryptosystem can be replaced by the lifted-ElGamal proxy re-encryption scheme widely documented in [51–55]. It is proved IND-CPA secure and is also characterized by the following relevant feature: both the first-level ciphertext and the second-level ciphertext satisfy the property of additive homomorphism [51].

7.1. Performance

The performance of the platform described above has been evaluated by employing the following resources. The C++ programs implementing $\mathcal{B}$, $\mathcal{RA}$, $\mathcal{CP}$, and the master node of the cloud computing platform $\mathcal{CL}$, run on distinct notebooks. Moreover, to simplify the test phase, the computing nodes within the cloud platform are emulated by separate C++ programs running on a single, further notebook.

All the notebooks are equipped with a CPU Intel(R) Core(TM) i7-1165G7 with frequency up to 4.7 GHz, 32 GB of RAM, and an SS disk of 1TB. They are connected by a GBit Ethernet.

Due to the complexity of the implemented platform and in order to correctly compare the proposed protocol with other solutions documented in the literature, the performance of the protocol has been evaluated in terms of main contributions to the total computation cost.

Let T_{pub}, T_{sim}, T_{sig}, T_{wat}, and T_{blk} denote the time costs for a public key encryption/decryption, a symmetric key encryption, a public-key digital signature, a watermark insertion, and a block creation, respectively. The total computation cost T_{total} required by the proposed protocol can be calculated by adding two contributions, denoted as T_1 and T_2 respectively. T_1 is equal to $3T_{pub} + 2T_{wat} + T_{blk}$, whereas T_2 is equal to $4T_{pub} + T_{sim} + T_{sig}$. However, the computation costs included in T_2 are related to operations on short bit strings, whereas the computation costs in T_1 are related to the multimedia digital contents to protect. This means that the contributions in T_2 are much smaller that those ones present in T_1. As a consequence, $T_{total} = T_1 + T_2 \approx T_1$, since $T_1 >> T_2$. In fact, the total computation cost of the proposed protocol turns out to be lower than that one calculated for the system described in [33], which is equal to $4T_h + 4T_{pub} + 2T_{ipfs} + 2T_{wat} + T_{blk}$, with T_h and T_{ipfs} denoting the computation cost of a one-way hash function and of the management of a multimedia digital content on IPFS, respectively. This can be considered a good result for the proposed protocol especially considering that the protocol is compared to an advanced system, such as that one proposed in [33], that outperforms many other relevant solutions documented in the literature and just reported in [33].

As to the watermark embedding into the encrypted domain, two sets of images have been used: images of 512×512 pixels and images of 1024×1024 pixels. The embedded fingerprints are 128 bits long, and the employed cryptosystem uses keys with 1024 bits. In these hypotheses, a watermark insertion in 512×512 images takes about 20–27 s depending on the watermarked image, whereas the same insertion in 1024×1024 images takes about 88–102 s. Therefore, to protect a 512×512 image, the proposed protocol implemented by the prototype platform takes about 55 s, whereas a 1024×1024 image requires about 150 s for the same operation.

Finally, as to the blockchain, the computational cost needed to perform each purchase transaction is about 90,000 gas units, whereas the computational cost per block has a limit of 8,000,000 gas units. Therefore, the number of transactions per block is about 88–90. Since the time required to mine a block of transactions in the blockchain is about 15–20 s, the rate at which the purchase transactions are committed by the blockchain is about 5–6 per second.

7.2. Discussion

Despite the prototype nature of the tested platform, the results reported above are encouraging and show that the proposed protocol is feasible in the current web context. In particular, the conducted tests show that the time costs required to execute the proposed protocol are strongly influenced by the times taken to encrypt and watermark multimedia digital contents. Such times can be also reduced, for instance, by using lighter encryption and watermarking algorithms, which could however turn out to be less secure, thus lowering the security level of the protocol. In any case, under the same levels of security of the adopted encryption and watermarking algorithms [12,13,56], the proposed protocol can achieve performances better, or at most, in line with those ones obtained by other similar solutions existing in the literature, such as those reported in [57]. In addition, it is worth noting that the adoption of a blockchain reduces the global performance of the proposed protocol by 10% at most. However, such a reduction cannot be predicted with certainty in real executions, since it mainly depends on a number of factors, such as the quality of the computing nodes involved in the mining processes and the time needed for propagating the purchase transactions, which are independent of the protocol [18,19,58]. On the other hand, the performance of the blockchain can be improved by increasing the number of transactions registered in each block and by reducing the time needed to mine a new block, or by adopting a more efficient consensus algorithm [58]. Nevertheless, experimenting such solutions does not qualify the effectiveness of the proposed protocol. In fact, the main goals of a watermarking protocol are robustness and security, together with a simplified protection scheme able to relieve content providers of the burden of implementing on-the-fly watermark insertions. In this regard, the ability to exploit cloud computing platforms together with an Ethereum, public and decentralized blockchain, characterized by immutability and lack of trusted third parties [18,19,48], represents a good compromise among conflicting objectives, such as security, simplicity, feasibility, and efficiency.

8. Conclusions

Cloud computing and blockchain technologies represent an actual plus in developing advanced watermarking protocols suited for the current Internet. They make it possible to achieve two main goals.

The former consists in enabling both professional content providers and common web users not provided with computing and storage resources to protect the contents they want to distribute or publish on the Internet. They can thus delegate a burdensome activity, such as watermark insertion, to cloud platforms according to their needs.

The latter consists in avoiding the participation of a TTP in the watermark insertion phase as well as a complex involvement of buyers in content purchase transactions. In fact, the use of smart contracts in the context of blockchain makes it possible to simplify the protection scheme without compromising the classic achievements characterizing “buyer friendly” and “mediated” protocols in terms of security and robustness [5,8,11,17]: (1) the content provider never releases unprotected contents, thus always keeping control on them; (2) pirated copies of $\bar{\bar{X}}$ can be always traced back to the corrupt buyer, since he/she is the only entity getting access to the final watermarked content $\bar{\bar{X}}$; (3) the specific problem described in [10] is definitively solved since the protocol prevents X from being released in a partially protected form; (4) the “identification and arbitration protocol” does not require the cooperation of a suspected buyer to make appropriate adjudications.

Moreover, it is worth noting that the malicious actions of $\mathcal{CP}$ are always disclosed and blocked by the protection protocol or, at most, they cause contents to be incorrectly protected, thus making it impossible to identify traitors. In fact, this just damages $\mathcal{CP}$. The same considerations also apply to $\mathcal{B}$, whose malicious actions are likewise disclosed and aborted by the protection protocol.

Finally, the performances achieved by the proposed protocol are better or, at most, equal to those ones achieved by similar copyright protection solutions documented in the literature, even though they are penalised by the consensus algorithm of the employed blockchain. However, such a drawback is widely counterbalanced by the positive aspects reported above. Moreover, next generation blockchains will be certainly characterized by improved consensus algorithms able to achieve better and better performances.

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Cox, I.; Miller, M.; Bloom, J.; Fridrich, J.; Kalker, T. *Digital Watermarking and Steganography*; Morgan Kaufmann: Burlington, MA, USA, 2007.
2. Barni, M.; Bartolini, F. Data Hiding for Fighting Piracy. *IEEE Signal Process. Mag.* **2004**, *21*, 28–39. [CrossRef]
3. Liu, K.J.R.; Trappe, W.; Wang, Z.J.; Wu, M.; Zhao, H. *Multimedia Fingerprinting Forensics for Traitor Tracing*; Hindawi Publishing Corporation: New York, NY, USA, 2005.
4. Gopalakrishnan, K.; Memon, N.; Vora, P.L. Protocols for watermark verification. *IEEE Multimed.* **2001**, *8*, 66–70. [CrossRef]
5. Frattolillo, F. A Buyer–Friendly and Mediated Watermarking Protocol for Web Context. *ACM Trans. Web* **2016**, *10*, 9. [CrossRef]
6. Frattolillo, F.; Landolfi, F. Designing a DRM System. In Proceedings of the The Fourth International Conference on Information Assurance and Security, Naples, Italy, 8–10 September 2008; IEEE Computer Society: Washington, DC, USA, 2008; pp. 221–226.
7. Frattolillo, F.; Landolfi, F.; Marulli, F. A Novel Approach to DRM Systems. In Proceedings of the 12th IEEE International Conference on Computational Science and Engineering, Vancouver, BC, Canada, 29–31 August 2009; IEEE Computer Society: Washington, DC, USA, 2009; pp. 492–497.
8. Frattolillo, F. A multiparty watermarking protocol for cloud environments. *J. Inf. Secur. Appl.* **2019**, *47*, 246–257. [CrossRef]
9. Memon, N.; Wong, P.W. A buyer-seller watermarking protocol. *IEEE Trans. Image Process.* **2001**, *10*, 643–649. [CrossRef]
10. Lei, C.L.; Yu, P.L.; Tsai, P.L.; Chan, M.H. An Efficient and Anonymous Buyer-Seller Watermarking Protocol. *IEEE Trans. Image Process.* **2004**, *13*, 1618–1626. [CrossRef]
11. Frattolillo, F. Watermarking protocols: An excursus to motivate a new approach. *Int. J. Inf. Secur.* **2018**, *17*, 587–601. [CrossRef]
12. Rial, A.; Deng, M.; Bianchi, T.; Piva, A.; Preneel, B. A Provably Secure Anonymous Buyer—Seller Watermarking Protocol. *IEEE Trans. Inf. Forensics Secur.* **2010**, *5*, 920–931. [CrossRef]
13. Rial, A.; Balasch, J.; Preneel, B. A Privacy-Preserving Buyer–Seller Watermarking Protocol Based on Priced Oblivious Transfer. *IEEE Trans. Inf. Forensics Secur.* **2011**, *6*, 202–212. [CrossRef]
14. Xu, Z.; Li, L.; Gao, H. Bandwidth Efficient Buyer-seller Watermarking Protocol. *Int. J. Inf. Comput. Secur.* **2012**, *5*, 1–10. [CrossRef]
15. Bianchi, T.; Piva, A. TTP-free asymmetric fingerprinting based on client side embedding. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 1557–1568. [CrossRef]
16. Bianchi, T.; Piva, A.; Shullani, D. Anticollusion solutions for asymmetric fingerprinting protocols based on client side embedding. *EURASIP J. Inf. Secur.* **2015**, *2015*, 6. [CrossRef]
17. Frattolillo, F. A Watermarking Protocol Based on Blockchain. *Appl. Sci.* **2020**, *10*, 7746. [CrossRef]
18. Casino, F.; Dasaklis, T.K.; Patsakis, C. A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telemat. Inform.* **2019**, *36*, 55–81. [CrossRef]
19. Aggarwal, S.; Chaudhary, R.; Aujla, G.S.; Kumar, N.; Choo, K.K.R.; Zomaya, A.Y. Blockchain for smart communities: Applications, challenges and opportunities. *J. Netw. Comput. Appl.* **2019**, *144*, 13–48. [CrossRef]
20. Chaudhary, S.; Somani, G.; Buyya, R. *Research Advances in Cloud Computing*; Springer: Berlin/Heidelberg, Germany, 2017.
21. Fontaine, C.; Galand, F. A Survey of Homomorphic Encryption for Nonspecialists. *EURASIP J. Inf. Secur.* **2007**, *2007*, 13801. [CrossRef]
22. Hartung, F.; Su, J.K.; Girod, B. Spread Spectrum Watermarking: Malicious Attacks and Counterattacks. In Proceedings of the Security and Watermarking of Multimedia Contents, San Jose, CA, USA, 23–29 January 1999; Delp, E.J., Wong, P.W., Eds.; SPIE: Bellingham WA, USA, 1999; Volume 3657, pp. 147–158.
23. Zhang, J.; Kou, W.; Fan, K. Secure buyer-seller watermarking protocol. *IEEE Proc. Inf. Secur.* **2006**, *153*, 15–18. [CrossRef]
24. Fan, C.I.; Chen, M.T.; Sun, W.Z. Buyer-Seller Watermarking Protocols with Off-line Trusted Parties. In Proceedings of the IEEE International Conference on Multimedia and Ubiquitous Engineering, Seoul, Korea, 22–24 May 2017; IEEE Computer Society: Washington, DC, USA, 2007; pp. 1035–1040.
25. Hu, Y.; Zhang, J. A Secure and Efficient Buyer-Seller Watermarking Protocol. *J. Multimed.* **2009**, *4*, 161–168. [CrossRef]

26. Katzenbeisser, S.; Lemma, A.; Celik, M.U.; van der Veen, M.; Maas, M. A Buyer—Seller Watermarking Protocol Based on Secure Embedding. *IEEE Trans. Inf. Forensics Secur.* **2008**, *3*, 783–786. [CrossRef]
27. Lemma, A.N.; Katzenbeisser, S.; Celik, M.U.; Van Der Veen, M. Secure Watermark Embedding Through Partial Encryption. In *Lecture Notes in Computer Science, Proceedings of the 5th International Workshop on Digital Watermarking, Jeju Island, Korea, 8–10 November 2006*; Shi, Y.Q., Jeon, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4283, pp. 433–445.
28. Tardos, G. Optimal probabilistic fingerprint codes. In Proceedings of the 35th Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, 9–11 June 2003; ACM: New York, NY, USA, 2003; pp. 116–125.
29. Meng, Z.; Morizumi, T.; Miyata, S.; Kinoshita, H. Design Scheme of Copyright Management System Based on Digital Watermarking and Blockchain. In Proceedings of the IEEE 42nd Annual Computer Software and Applications Conference; Tokyo, Japan, 23–27 July 2018; IEEE Computer Society: Washington, DC, USA, 2018; pp. 359–364.
30. Zhaofeng, M. Weihua, H.; Hongmin, G. A new blockchain-based trusted DRM scheme for built-in content protection. *EURASIP J. Image Video Process.* **2018**, *2018*, 91. [CrossRef]
31. Peng, W.; Yi, L.; Fang, L.; XinHua, D.; Ping, C. Secure and Traceable Copyright Management System Based on Blockchain. In Proceedings of the IEEE 5th International Conference on Computer and Communications, Chengdu, China, 6–9 December 2019; IEEE Computer Society: Washington, DC, USA, 2019; pp. 1243–1247.
32. Zhao, S.; O'Mahony, D. BMCProtector: A Blockchain and Smart Contract Based Application for Music Copyright Protection. In Proceedings of the International Conference on Blockchain Technology and Application, Xi'an, China, 10–12 December 2018; ACM: New York, NY, USA, 2018; pp. 1–5.
33. Ma, Z.; Jiang, M.; Gao, H.; Wang, Z. Blockchain for digital rights management. *Future Gener. Comput. Syst.* **2018**, *89*, 746–764. [CrossRef]
34. Zhang, Z.; Pei, Q.; Ma, J.; Yang, L. Security and Trust in Digital Rights Management: A Survey. *Int. J. Netw. Secur.* **2009**, *9*, 247–263.
35. Macrinici, D.; Cartofeanu, C.; Gao, S. Smart contract applications within blockchain technology: A systematic mapping study. *Telemat. Inform.* **2018**, *35*, 2337–2354. [CrossRef]
36. Zhao, W.; Varadharajan, V.; Mu, Y. A Secure Mental Poker Protocol over the Internet. In Proceedings of the Australasian Information Security Workshop Conference on ACSW Frontiers 2003, Adelaide, Australia, 1 February 2003; Australian Computer Society: Adelaide, Australia, 2003; pp. 105–109.
37. Choi, J.G.; Sakurai, K.; Park, J.H. Does It Need Trusted Third Party? Design of Buyer-Seller Watermarking Protocol without Trusted Third Party. In *Lecture Notes in Computer Science, Proceedings of the 1st International Conference on Applied Cryptography and Network Security, Kunming, China, 16–19 October 2003*; Zhou, J., Yung, M., Han, Y., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2846, pp. 265–279.
38. Wang, C.; Leung, H.F.; Cheung, S.C.; Wang, Y. Use of Cryptographic Technologies for Privacy Protection of Watermarks in Internet Retails of Digital Contents. In Proceedings of the 18th International Conference on Advanced Information Networking and Application, Fukuoka, Japan, 29–31 March 2004; IEEE Computer Society: Washington, DC, USA, 2004.
39. Jeng, F.J.; Huang, J.C.; Chen, T.H. An Improved Anonymous Buyer-Reseller Watermarking Protocol. *Int. J. Netw. Secur.* **2016**, *18*, 728–735.
40. Rannenberg, K. Multilateral Security. A Concept and Examples for Balanced Security. In Proceedings of the 9th ACM Workshop on New Security Paradigms, County Cork, Ireland, 18–21 September 2000; ACM: New York, NY, USA, 2001; pp. 151–162.
41. Williams, D.M.; Treharne, H.; Ho, A.T.S. On the Importance of One-time Key Pairs in Buyer-seller Watermarking Protocols. In Proceedings of the International Conference on Security and Cryptography, Athens, Greece, 26–28 July 2010; IEEE Computer Society: Washington, DC, USA, 2010; pp. 441–446.
42. Canetti, R. Security and Composition of Cryptographic Protocols: A Tutorial. *ACM SIGACT News* **2006**, *37*, 67–92. [CrossRef]
43. Petitcolas, F.A.P. Watermarking schemes evaluation. *IEEE Signal Process. Mag.* **2000**, *17*, 58–64. [CrossRef]
44. Chen, B.; Wornell, G. Quantization index modulation: A class of provably good methods for digital watermarking and information embedding. *IEEE Trans. Inf. Theory* **2001**, *47*, 1423–1443. [CrossRef]
45. Malvar, H.S.; Florêncio, D.A.F. Improved Spread Spectrum: A New Modulation Technique for Robust Watermarking. *IEEE Trans. Signal Process.* **2003**, *51*, 898–905. [CrossRef]
46. Piva, A.; Bianchi, T.; De Rosa, A. Secure Client-Side ST-DM Watermark Embedding. *IEEE Trans. Inf. Forensics Secur.* **2010**, *5*, 13–26. [CrossRef]
47. Rittinghouse, J.W.; Ransome, J.F. *Cloud Computing: Implementation, Management, and Security*; CRC Press: Boca Raton, FL, USA, 2009.
48. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, X.; Wang, H. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* **2018**, *14*, 352–375. [CrossRef]
49. Huang, K.; Tso, R. A Commutative Encryption Scheme based on ElGamal Encryption. In Proceedings of the International Conference on Information Security and Intelligent Control, Yunlin, Taiwan, 14–16 August 2012; IEEE Computer Society: Washington, DC, USA, 2012; pp. 156–159.
50. Huang, K.; Tso, R.; Chen, Y.C. One-time-commutative public key encryption. In Proceedings of the Computing Conference 2017, London, UK, 18–20 July 2017; IEEE Computer Society: Washington, DC, USA, 2017; pp. 814–818.
51. Yu, B.; Zhang, C.; Li, W. File matching based on secure authentication and proxy homomorphic re-encryption. In Proceedings of the 11th International Conference on Machine Learning and Computing, Zhuhai, China, 22–24 February 2019; ACM: New York, NY, USA, 2019; pp. 472–476.

52. Samanthula, B.K.; Elmehdwi, Y.; Howser, G.; Madria, S. A secure data sharing and query processing framework via federation of cloud computing. *Inf. Syst.* **2015**, *48*, 196–212. [CrossRef]
53. Gao, C.z.; Cheng, Q.; Li, X.; Xia, S.b. Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network. *Clust. Comput.* **2019**, *22*, 1655–1663. [CrossRef]
54. Shafagh, H.; Hithnawi, A.; Burkhalter, L.; Fischli, P.; Duquennoy, S. Secure sharing of partially homomorphic encrypted IOT data. In Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems, Delft, The Netherlands, 6–8 November 2017; ACM: New York, NY, USA, 2017; pp. 1–14.
55. Derler, D.; Ramacher, S.; Slamanig, D. Homomorphic proxy reauthenticators and applications to verifiable multi-user data aggregation. In Proceedings of the International Conference on Financial Cryptography and Data Security, Kota Kinabalu, Malaysia, 10–14 February 2020; Springer: Sliema, Malta, 2017; pp. 124–142.
56. Deng, M.; Bianchi, T.; Piva, A.; Preneel, B. An efficient buyer-seller watermarking protocol based on composite signal representation. In Proceedings of the 11th ACM Workshop on Multimedia and Security, Princeton, NJ, USA, 7–8 September 2009; ACM: New York, NY, USA, 2009; pp. 9–18.
57. Qureshi, A.; Megías, D. Blockchain-Based Multimedia Content Protection: Review and Open Challenges. *Appl. Sci.* **2021**, *11*, 1. [CrossRef]
58. Bamakan, S.M.H.; Motavali, A.; Bondarti, A.B. A survey of blockchain consensus algorithms performance evaluation criteria. *Expert Syst. Appl.* **2020**, *154*, 113385. [CrossRef]