中国科学: 信息科学 2022 年 第 52 卷 第 7 期: 1272–1286
SCIENTIA SINICA Informationis
论文

云与用户之间交互式抗屏摄密文水印协议

董晓娟, 张卫明*, 方涵, 俞能海

中国科学技术大学中国科学院电磁空间信息重点实验室, 合肥 230026
* 通信作者. E-mail: zhangwm@ustc.edu.cn

收稿日期: 2021–06–25; 修回日期: 2021–08–12; 接受日期: 2021–11–29; 网络出版日期: 2022–07–13

国家自然科学基金 (批准号: 62072421, 62002334, 62121002)、国家重点研发计划项目 (批准号: 2018YFB0804100)、安徽省自然科学基金 (批准号: 2008085QF296) 和中国科学技术大学探索类基金 (批准号: YD3480002001) 资助项目

摘要 现有的云存储服务普遍推出了浏览文件的功能, 这为用户通过拍摄屏幕上的文件来窃取信息带来极大的便利, 同时也加重了信息泄露的风险. 因此, 用户通过拍照窃取信息对云存储服务来说是一个潜在的安全性问题. 现有的办法是云端在推送图像至用户端前, 先嵌入与用户身份相关的抗屏摄水印, 那么云端能从拍摄的图像检测到水印, 发现用户身份. 然而云端和用户端都有含用户水印的图像, 我们无法确认被泄露的图像是从云端流出的还是从用户端流出的. 为了消除泄露源的二义性, 使用密文水印, 云端将用户的密文水印嵌入到明文图像, 生成含用户水印的密文图像. 只有用户端能将之解密, 获得相应的明文图像. 一旦该图像被偷拍泄露, 我们可断定该图像一定是从用户端流出的. 但是目前的抗屏摄水印算法不支持上述功能, 仅支持明文水印的嵌入. 为此, 本文提出一个云与用户之间的抗屏摄密文水印协议, 旨在保护版权用户对其图像的版权, 防止使用图像的用户通过偷拍窃取图像. 通过理论分析和仿真实验, 本文验证了所提出协议在实施过程中的安全性和有效性.

关键词 云存储, 密文水印协议, 抗屏摄水印, 版权保护, 泄露追踪

1 引言

个人云存储是一种在线存储模式, 通过互联网为个人提供存储、共享、管理、加密等服务, 已经成为人们工作和生活中的重要工具. 《2020~2021 年中国个人网盘专题调研报告》预计 2021 年中国个人云盘用户将超过 4 亿人, 并预测未来使用人数会继续增加. 为吸引更多用户, 提高市场占有率, 云盘服务商在提供存储服务的同时也在不断地开发其他增值服务, 如在线观看图片、在线点播视频、在线编辑文档、设链分享等. 但正是这些功能的开发, 滋生了云储存服务中的版权侵权问题 [1].

云存储服务提供的在线浏览功能方便用户查看文件, 用户虽然不能下载屏幕上的文件, 但能通过手机或照相机对着屏幕拍照来获取文件, 进而可复制、传播, 造成信息的泄露, 如图 1 所示. 为了从拍

引用格式: 董晓娟, 张卫明, 方涵, 等. 云与用户之间交互式抗屏摄密文水印协议. 中国科学: 信息科学, 2022, 52: 1272–1286, doi: 10.1360/SSI-2021-0213
Dong X J, Zhang W M, Fang H, et al. Interactive screen-shooting resilient ciphertext watermarking protocol between the cloud and the user (in Chinese). Sci Sin Inform, 2022, 52: 1272–1286, doi: 10.1360/SSI-2021-0213

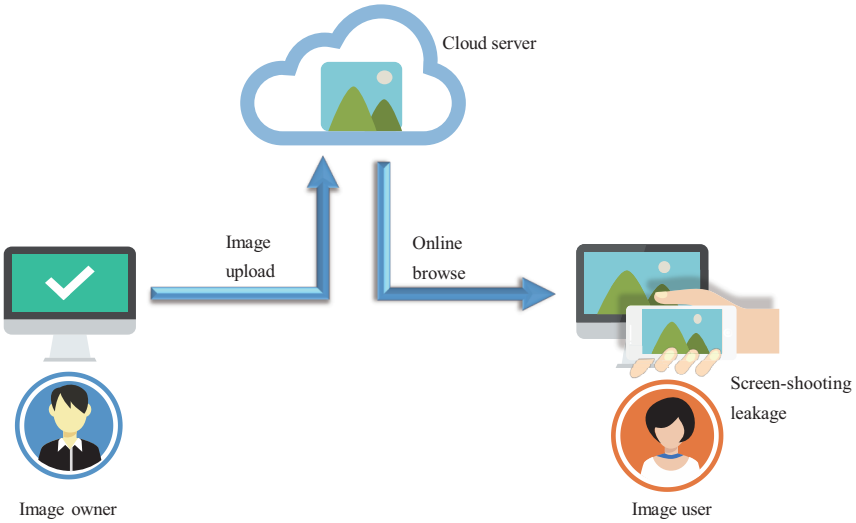

图 1 (网络版彩图) 云存储服务中浏览功能带来的屏摄泄露

Figure 1 (Color online) Screen-shooting leakage caused by the online browse in cloud storage services

摄文件中定位到偷拍用户, 云端在推送文件到用户端前, 将用户的身份信息当作水印按照抗屏摄的方式 [2~5] 嵌入到文件中, 这样可从拍摄后的文件中检测到水印, 发现用户身份. 但从泄露的文件中检测到用户水印, 用户仍可能否认偷拍并泄露该文件, 因为云端嵌入用户水印到文件, 这是因为云端也能获得含用户水印的文件, 被泄露的文件可能是从用户端流出, 也可能是从云端流出. 为了消除泄露源的二义性, 云端不能获得含用户身份水印的文件, 但仍然要负责水印的嵌入, 即云端负责操作但不能获得操作的结果, 而密文计算技术正好满足这个要求. 云端在用户浏览文件前嵌入用户的密文水印, 在将含用户身份水印的密文文件推送到用户端, 用户端解密后获得含用户水印的明文文件, 然后将之展示在屏幕上. 云端考虑到屏摄的过程会破坏嵌入的水印信息, 这就需要嵌入密文水印能抵抗屏摄攻击, 即需要一种抗屏摄的密文水印. 总结上述抗屏摄密文水印的 4 个特点如下: (1) 能抵抗屏摄攻击; (2) 接收方加密水印, 生成密文水印; (3) 发送方嵌入密文水印, 生成含水印的密文文件, 但不能解密密文水印以及含水印的密文文件; (4) 接收方解密含水印的密文文件, 获得相应的含水印明文文件.

本文将抗屏摄水印方案 [2] 嵌入明文水印转换为嵌入密文水印. 文献 [2] 利用水印比特控制一对数值 A 和 B 的大小关系. 如果水印比特是 0, 调整 A 和 B 的数值, 使得 A 大于 B; 如果水印比特是 1, 调整 A 和 B 的数值, 使得 A 小于 B. 而密文水印的明文值是未知的, 本文先假设密文水印的明文值为 0 或 1, 接着调整 A 和 B 的数值, 利用不经传输协议 [6] 传输调整后的 A 和 B 与密文水印, 获得密文数值对, 并将水印值与这对数值的大小关系绑定. 只有密文水印中的明文值等于假设的水印值时, 这两个密文数值才可被解密, 其明文数值的大小关系印证了水印值.

本文将提出的抗屏摄的密文水印应用到云存储服务中版权保护和泄露追踪的场景, 设计一个云 – 用户之间的交互式抗屏摄密文水印协议, 以图像为数据对象, 主要由涉及到云端 (cloud server, CS)、图像拥有者 (image owner, IO) 和图像使用者 (image user, IU) 参与, IO 和 IU 都是云服务的用户. IO 上传图像至 CS; 经过 IO 同意, IU 从 CS 下载图像但不能分享该图像给其他未经 IO 许可的用户. CS 和 IU 是潜在的版权侵权主体, IO 关注与 CS 和 IU 的行为, 具体的关注点有: (1) 图像的版权; (2) CS 存储的图像不会丢失、泄露; (3) 分享给 IU 的图像未经允许不会被传播发送给其他用户. 为此, IO 在上传图像至 CS 之前, 嵌入版权水印, 证明对图像的所有权; 嵌入 CS 的水印, 防止 CS 泄露该图像; IO 在

表 1 水印与载体数据之间的组合用途

Table 1 Application of the combination of watermark and carrier data

	Watermark	Encrypted watermark
Carrier	Confirm copyright	Confirm copyright, trace the source of leakage
Encrypted carrier	Confirm copyright, ensure the confidentiality of the carrier	Confirm copyright, trace the source of leakage, ensure the confidentiality of the carrier

分享 CS 的图像给 IU 之前, 嵌入 IU 的水印, 防止 IU 散播图像. IO 在图像中嵌入 CS 或 IU 的抗屏摄密文水印, 以消除泄露源的二义性, 也能防止 IU 通过屏摄窃取图像. 图像在传输过程中是被加密的以确保图像内容的保密性. 本文中的云 – 用户之间的交互式抗屏摄密文水印协议的贡献点如下:

(1) IO 实现抗屏摄密文水印的嵌入, 防止他人通过屏摄获取图像.

(2) 嵌入版权水印以验证 IO 对图像的版权, 嵌入 CS 的密文水印以追踪 CS 泄露图像的行为, 嵌入 IU 的密文水印以追踪 IU 泄露图像的行为.

(3) 水印的嵌入位置是模糊的, 防止 CS 或 IU 破坏嵌入的水印, 使得水印检测失效.

(4) 嵌入的水印可被抹除, 保证 IO 能从 CS 无损取回原图像.

2 相关工作

目前网络传输中的数字媒体有两个方面的安全需求, 存取安全和使用安全 [7]. 而密码技术和水印技术被认为分别实现这两种安全的有效手段. 研究学者根据不同需求设计出了水印与载体数据之间不同的组合方式, 如表 1 所示.

第 1 种组合是当明文水印作为版权信息嵌入明文载体中, 通过提取载体中的版权信息, 确定载体数据的版权人 [2~5, 8, 9]. 第 2 种组合是将明文水印嵌入明文载体, 然后加密再传输, 解密后版权信息仍存在于载体 [10~13] 中. 此处的加密主要是为了保密传输. 第 3 种组合是一方先加密载体, 然后将加密的载体交给另外一方, 由另一方在加密的载体中嵌入明文水印, 解密之后水印依然存在于载体中. 这里的加密用于保护载体数据的机密性, 可避免原始载体被非法使用, 典型的案例有密文域可逆信息隐藏 [14~16]. 第 4 种组合是接收方先加密水印, 发送方负责嵌入密文水印, 生成含水印的密文载体, 接收方解密后获得含水印的明文载体; 先加密水印后嵌入可防止发送方知晓水印信息, 进一步防止发送方通过伪造并泄露含接收方水印的明文载体来栽赃接收方; 同时, 接收方不能散播含水印的载体, 因为只有接收方能获得含水印的明文载体, 如果含水印的载体被泄露, 泄漏源一定是接收方而非发送方, 典型应用有买方 – 卖方模型 [17, 18]. 近年来, 买方 – 卖方模型被应用到云存储的场景 [19~21]. 版权人将作品存放在云端, 版权人可以先在作品中嵌入明文水印以保证版权, 然后嵌入云端的密文水印, 获得密文含云水印的作品, 将之上传到云端; 云解密后获得明文含云水印的作品, 将该作品存储在服务器. 一旦存储在云端的明文作品被泄露, 版权人从作品中提取出版权水印来确认版权, 检测出云端水印来确认作品是被哪个云端泄露的. 在版权人取回作品时, 可二次使用买方 – 卖方模型嵌入取回标识的水印 [19] 或抹除被嵌入的云水印 [20, 21]. 在版权人同意分享自己的作品给使用者时, 也可二次使用买方 – 卖方模型抹除被嵌入的云水印同时嵌入使用者的水印 [21], 生成含使用者水印的密文图像. 使用者解密后获得含水印的明文图像. 然而, 若使用者在用户端浏览版权人的图像时可通过屏摄获取图像, 而屏摄的过程会破坏嵌入的常规水印信息 [2], 使得水印检测失效, 使用者便可私自分发版权人的图像而不会被检测到. 因此, 在提供浏览功能的云存储场景中使用数字水印技术需考虑抗屏摄攻击.

目前明文域的抗屏摄水印方案有文献 [2~5]. 文献 [2] 发现传统的水印方法对屏摄过程没有抗干扰能力, 屏摄过程带来的主要失真有镜头失真、光照失真、摩尔纹失真. 文献 [2] 根据这些失真特性提出了一种基于大小关系的水印方案, 对局部特征区域进行离散余弦变换 (discrete cosine transform, DCT), 然后在选择的一对 DCT 系数嵌入一比特水印信息. 相继, 文献 [3] 提出了一种基于深度学习的抗屏摄水印方案. 文献 [4] 设计了一种水印同步方法, 水印信息被嵌入到离散傅立叶变换 (discrete Fourier transform, DFT) 域中. 文献 [5] 提出了一种基于特征同步的水印方案, 水印信息被嵌入到 DFT 域. 结合加密技术的抗屏摄水印方案有文献 [13], 文献 [13] 的方案属于第 2 种加密与水印的组合, 可用于保密传输和版权验证, 但无法用于泄露追踪, 因发送方和接收方都有明文含水印的图像. 而先加密水印后嵌入, 也就是在载体中嵌入密文水印可消除泄露源的二义性. 为了使用密文水印同时保证水印具有抗屏摄的能力, 本文将文献 [2] 中嵌入明文水印的方法转变为可嵌入密文水印, 实现密文数据的比较. 目前, 在不知道数值的情况下比较数值的大小其他方案有同态加密方案 [22], 混淆电路 (百万富翁问题) [23] 和安全计算协议 [24], 其主要思想是先加密数值或者加入掩蔽因子, 再进行特定的计算, 根据计算结果反推数值的大小关系. 本文先假设水印值, 根据假设值调整明文数值对 A 和 B 之间的大小关系, 用不经意传输协议 [6] 传输密文水印和调整后的 A 和 B, 获得密文 A 和 B. 当密文水印的明文值等于假设值时, 密文 A 和 B 可解密, 同时 A 和 B 的大小关系代表水印值, 这使得水印值与一对数值的大小关系对应. 调整数值大小的操作在明文中进行, 解密的密文同时显露了明文数值的大小关系, 避免了对密文数据比较大小.

综上, 表 2 [25~27] 给出了本文协议与相关协议 [19~21] 在支持图像的可展示性、支持图像无损取回、第三方的可信度、具体的水印算法, 以及加密基础方面的不同.

从表 2 看出, 文献 [19~21] 支持存储明文图像, 使得图像具有可展示性, 这也提供了通过偷拍窃取图像的可能, 而拍摄的过程会破坏嵌入的水印. 本文使用了抗屏摄水印算法 [2] 在支持存储明文图像同时防止图像被偷拍.

3 预备知识

本节介绍本文协议涉及到的不经意传输协议 [6] 和抗屏摄水印算法 [2].

3.1 不经意传输协议

在 Tzeng [6] 提出的不经意传输方案中, 发送者 Alice 有 2 个消息 m_1, m_2, 接收者 Bob 从中选择 m_α, $\alpha \in \{1,2\}$, 并满足以下特性:

- 正确性. 如果 Alice 和 Bob 均遵守协议, 那么 Bob 只能获得 1 个消息 m_α, $\alpha \in \{1,2\}$.
- 接收者的隐私. Alice 不会知道 Bob 获得的是哪一个消息.
- 发送者隐私. Bob 不会知道另一个消息.

设群的 $\mathbb{G}$ 阶是素数 q, 生成元 $g, h \in \mathbb{G}$, 且离散对数 $\log_g h$ 是无人知晓的. 文献 [6] 中不经意传输协议的交互过程如表 3 所示. 因为 $c_\alpha = (c_{\alpha,1}, c_{\alpha,2}) = (g^{k_\alpha}, m_\alpha(y/h^\alpha)^{k_\alpha})$, 所以 $c_{\alpha,2}/(c_{\alpha,1})^r = m_\alpha(y/h^\alpha)^{k_\alpha}/(g^{k_\alpha})^r = m_\alpha$.

3.2 抗屏摄的水印算法

文献 [2] 提出的水印算法能抵抗屏摄过程对水印信号的干扰, 使得屏摄图像中的水印仍然能够被提取. 水印是由 0 和 1 构成的比特串. 设置初始水印长度为 39 bit, 利用纠错编码扩充到 64 bit, 接着

表 2　本协议与其他协议的功能比较

Table 2　Comparison of the functions of our protocol with other protocols

Protocol	Exhibition	Lossless retrieval	Semi-trusted third party	Embedding position of watermark hidden	Watermarking algorithm	Screen resistance	Encryption algorithm
Ref. [19]	√	×	×	×	Quantization watermark	×	Paillier [25]
Ref. [20]	√	√	×	×	Quantization watermark	×	Paillier [25]
Ref. [21]	√	√	√	√	Multiplicative spread spectrum watermark	×	Restained Paillier [26]
This paper	√	√	×	√	Watermark based on size relation	√	ElGamal [27]

表 3　不经意传输协议 [6] 的交互过程

Table 3　Interaction process of the oblivious transmission scheme [6]

Alice input: m_1, m_2		Bob input $\alpha \in \{1,2\}$
		Select $r \in \mathbb{Z}_q^*$
Select $k_1, k_2 \in \mathbb{Z}_q^*$	$\xleftarrow{y}$	Compute $y = g^r h^\alpha$
Compute $c_1 = (g^{k_1}, m_1(y/h)^{k_1})$, $c_2 = (g^{k_2}, m_2(y/h^2)^{k_2})$	$\xrightarrow{c_1, c_2}$	Compute $m_\alpha = c_{\alpha,2}/(c_{\alpha,1})^r$

重复嵌入 5 次. 下面简单介绍该算法的水印嵌入和水印提取过程 [28].

水印嵌入阶段. 对载体图像中进行尺度不变特征变换 (scale-invariant feature transform, SIFT), 并筛选出若干个 SIFT 点, 增强这些点的特征强度. 从增强后点的周围区域中选出 $a \times b$ 个 8×8 的块. 对每个块进行 DCT 变换, 得到 DCT 系数矩阵, 取出 $(4,5)$ 和 $(5,4)$ 位置上的系数, 分别被标记为 C_1 和 C_2, 进行 1 bit 水印 w 的嵌入, 其方式为

$$\begin{cases} d \geqslant |q_2 - q_1| \cdot \dfrac{C_1 \cdot \max(q_1,q_2) + C_2 \cdot \min(q_1,q_2)}{2q_1q_2} + \dfrac{(q_1+q_2)\cdot r}{2}, \\ C_1 = \max(q_1,q_2) + \dfrac{d}{2},\ C_2 = \min(q_1,q_2) - \dfrac{d}{2}, \quad \text{if } w = 0, \\ C_1 = \min(q_1,q_2) - \dfrac{d}{2},\ C_2 = \max(q_1,q_2) + \dfrac{d}{2}, \quad \text{if } w = 1, \end{cases}$$

其中, q_1 和 q_2 是 JPEG 压缩表中 $(4,5)$ 和 $(5,4)$ 位置上的量化系数, r 表示水印的嵌入强度. 为了保留 JPEG 压缩后 C_1 与 C_2 之间的大小关系, 继续增加 C_1 与 C_2 之间的绝对差 d, 满足: 若 $C_1 \geqslant C_2$, 使得 $\lfloor \frac{C_1}{q_1} \rfloor \leqslant \lfloor \frac{C_2}{q_2} \rfloor$; 若 $C_1 < C_2$, 使得 $\lfloor \frac{C_1}{q_1} \rfloor > \lfloor \frac{C_2}{q_2} \rfloor$. 本文设置 $r \geqslant 1$, $q_1 = 29$, $q_2 = 40$. 嵌入每个水印比特后, 逆 DCT 变换每个块, 得到含水印的图像.

水印的提取阶段. 先对含水印图像进行透视畸变的校正, 确保水印在一定区域内, 将每区域分成 $a \times b$ 个 8×8 的块; 每个块进行 DCT 变换, 取出系数 C_1 和 C_2, 比较 C_1 和 C_2 的大小提取水印. 如果 $C_1 \geqslant C_2$, $w = 0$; 否则, $w = 1$.

提取出 5 个水印序列, 当 5 个水印序列之间的差距不大于阈值 th 时, 说明图像定位准确, 水印被正确提取. 否则, 提取水印失败. 水印序列之间的最小差距的阈值 th 被看做一个假设检验问题. 根据纽曼皮尔逊 (Neyman-Pearson) 准则, 文献 [2] 中测试不同距离下的阈值. 在虚警率设定为 $\alpha = 10^{-2}$ 时, th 设置为 6.

表 4 协议中的英文符号及其含义

Table 4 English symbols and their meanings

Symbol	Meaning
X, Y	X is the original image, and Y is a copy of image X
IO	Image owner
IU	Image user
CS, J	CS stands for the cloud server, and J stands for the judge
$\mathrm{pk}_{o/c/u}, \mathrm{sk}_{o/c/u}$	Public and private keys of IO/CS/IU
$W_{o/c/u}$	The plaintext watermark of IO/CS/IU is a string of length L composed of {1,2}
$\mathrm{EW}_{o/c/u}$	The ciphertext watermark of IO/CS/IU is the encrypted $W_{o/c/u}$ under the public key
K_{oc}	The key of scrambling watermark in image upload phase
K_{ou}	The key of scrambling watermark in image sharing phase
L, n	L is the watermark length, and n is the number of random watermarks
W_s	The random watermark is a string with length L composed of {1,2}
EW_s	The ciphertext watermark is the encrypted W_s by IO
EW_{oc}	Mix EW_o, EW_c, and EW_s, and scramble the generated ciphertext watermark with K_{oc}
EW_{ou}	Mix EW_o, EW_u, and EW_s, and scramble the generated ciphertext watermark with K_{ou}

4 云 – 用户之间的交互式抗屏摄密文水印协议

本协议分为初始化、上传、取回和图像泄露追踪 4 个阶段, 涉及 4 个参与者, 分别是: 图像拥有者 (IO)、云服务器 (CS)、图像使用者 (IU) 和一个法官 (J). 在上传图像前, IO 将 CS 的密文水印嵌入图像中. 在图像分享阶段前, CS 的密文水印被抹去, 同时将 IU 的密文水印嵌入. 在图像取回阶段, 恢复出原始不含水印的图像. 为了简洁起见, 本文将常用的词语标识为英文符号, 表 4 给出了这些英文符号及其含义.

在上传图像 X 阶段, IO 将 CS 的密文水印 EW_c 嵌入图像中. IO 假设 CS 的明文水印是 1 即 $W_c = 1$, 调整一对数值 A 和 B 的大小, 使得 A 小于 B, 调整后的数值记为 (A_1, B_1); 假设 $W_c = 2$, 调整 A 和 B 的大小, 使得 A 大于 B, 调整后的数值记为 (A_2, B_2). IO 用 EW_c 加密 (A_1, B_1), (A_2, B_2), 生成 $(\mathrm{EA}_1, \mathrm{EB}_1)$, $(\mathrm{EA}_2, \mathrm{EB}_2)$. 结果, 同一密文水印 EW_c 对应两组密文数据. 由不经意传输协议的特性决定只有假设的明文水印值等于密文中的明文水印值时, 才可解密相应的密文数据, 否则解密出错. 即当 $W_c = 1$ 时, CS 只能解密 $(\mathrm{EA}_1, \mathrm{EB}_1)$; 当 $W_c = 2$ 时, CS 只能解密 $(\mathrm{EA}_2, \mathrm{EB}_2)$.

在图像取回阶段, IO 要求取回原图, 也就是原数值对 A 和 B. IO 提前在上传图像阶段用公钥 pk_o 加密原数值与调整后数值之间的比值生成 $E_{\mathrm{pk}_o}(\frac{A}{A_1}, \frac{B}{B_1})$, $E_{\mathrm{pk}_o}(\frac{A}{A_2}, \frac{B}{B_2})$, 并将密文比值发送给 CS. 在收到 IO 取回原图的请求后, CS 根据水印值 $W_c = i$, $i \in \{1, 2\}$ 选择出 $E_{\mathrm{pk}_o}(\frac{A}{A_i}, \frac{B}{B_i})$, 进而计算出原数值的密文 $E_{\mathrm{pk}_o}(\frac{A}{A_i}, \frac{B}{B_i}) \times (A_i, B_i) = E_{\mathrm{pk}_o}(A, B)$, 将计算结果返还. IO 解密后获得原数值对 A 和 B.

在图像分享阶段, CS 需要抹除 CS 的水印同时嵌入 IU 的密文水印. 但是 CS 与 IU 在相同位置上的水印的取值可能不同, 因此 CS 和 IU 要选用不同的数值对. CS 需用已知的一对数值求出未知的另外一对数值. 为此, IO 提前在上传图像阶段用 pk_o 加密调整后数值之间的比值 $(\frac{A_2}{A_1}, \frac{B_2}{B_1})$, $(\frac{A_1}{A_2}, \frac{B_1}{B_2})$, 将生成的 $E_{\mathrm{pk}_o}(\frac{A_2}{A_1}, \frac{B_2}{B_1})$, $E_{\mathrm{pk}_o}(\frac{A_1}{A_2}, \frac{B_1}{B_2})$ 发送给 CS. 在收到 IU 的请求后, CS 根据水印值 $W_c = i$, $i \in \{1, 2\}$ 选出未知数值对与已知数值对的密文比值 $E_{\mathrm{pk}_o}(\frac{A_{j\neq i}}{A_i}, \frac{B_{j\neq i}}{B_i})$, 计算未知数值对的密文 $E_{\mathrm{pk}_o}(\frac{A_{j\neq i}}{A_i}, \frac{B_{j\neq i}}{B_i}) \times (A_i, B_i) = E_{\mathrm{pk}_o}(A_j, B_j)$, 用不经意传输协议 [6] 整合 IU 的密文水印和 $E_{\mathrm{pk}_o}(A_j, B_j)$, 获得 $E_{\mathrm{pk}_o}(\mathrm{EA}_j, \mathrm{EB}_j)$. CS 借

助 IO 解密 $E_{\mathrm{pk}_o}(\mathrm{EA}_j, \mathrm{EB}_j)$, 最后将 $(\mathrm{EA}_j, \mathrm{EB}_j)$ 发送给 IU. 在这个过程中 CS 只知道 (A_i, B_i) 但不知道 $(A_{j\neq i}, B_{j\neq i})$, 防止 CS 用 $(A_{j\neq i}, B_{j\neq i})$ 替换 (A_i, B_i), 更改嵌入的水印信息.

在图像泄露追踪阶段, 检测水印是 CS 的水印或 IU 的水印, 来判定泄露者是 CS 或者是 IU. 下面介绍各个阶段的具体交互过程.

4.1 初始化阶段

(1) IO 生成阶数为素数 q 的群 $\mathbb{G}$, 生成元 $g, h \in \mathbb{G}$, $\log_g h = t$; 生成相应的公私钥对 $\{\mathrm{pk}_o, \mathrm{sk}_o\} = \{h_o = g^{\theta_o} \bmod q, \theta_o\}$.

(2) CS 生成相应的公私钥对 $\{\mathrm{pk}_c, \mathrm{sk}_c\} = \{h_c = g^{\theta_c} \bmod q, \theta_c\}$.

(3) IU 生成相应的公私钥对 $\{\mathrm{pk}_u, \mathrm{sk}_u\} = \{h_u = g^{\theta_u} \bmod q, \theta_u\}$.

(4) IO 生成由 {1,2} 构成的版权水印 $W_o = \{w_{o,i} | w_{o,i} \in \{1,2\}\}$, $i = \{1, \ldots, L\}$.

(5) CS 生成由 {1,2} 构成的水印 $W_c = \{w_{c,i} | w_{c,i} \in \{1,2\}\}$; 选取 $r_{c,i}, t_{c,i} \in \mathbb{Z}_q^*$, 计算 CS 的密文水印 $\mathrm{EW}_c = \{\mathrm{ew}_{c,i} | \mathrm{ew}_{c,i} = g^{r_{c,i}} h^{w_{c,i}} \bmod q, r_{c,i} h_c^{t_{c,i}} \bmod q, g^{t_{c,i}} \bmod q\}$, $i = \{1, \ldots, L\}$; 将 EW_c 公布在云平台.

(6) IU 生成由 {1,2} 构成的水印 $W_u = \{w_{u,i} | w_{u,i} \in \{1,2\}\}$; 选取随机数 $r_{u,i}, t_{u,i} \in \mathbb{Z}_q^*$, 计算密文水印 $\mathrm{EW}_u = \{\mathrm{ew}_{u,i} | \mathrm{ew}_{u,i} = g^{r_{u,i}} h^{w_{u,i}} \bmod q, r_{u,i} h_u^{t_{u,i}} \bmod q, g^{t_{u,i}} \bmod q\}$, $i = \{1, \ldots, L\}$; 将 EW_u 公布在云平台.

4.2 图像上传阶段

4.2.1 准备阶段

(1) IO 选取 $r_{o,i}, t_{o,i} \in \mathbb{Z}_q^*$, 用 CS 的公钥 h_c 加密 IO 的水印, 生成 $\mathrm{EW}_o = \{\mathrm{ew}_{o,i} | \mathrm{ew}_{o,i} = g^{r_{o,i}} h^{w_{o,i}} \bmod q, r_{o,i} h_c^{t_{o,i}} \bmod q, g^{t_{o,i}} \bmod q\}$, $i = \{1, \ldots, L\}$; 生成由 {1,2} 构成的长度为 nL 的串作为随机水印 W_s; 然后, 选取 $r_{s,i}, t_{s,i} \in \mathbb{Z}_q^*$, 用 h_c 加密随机水印, 生成随机水印的密文形式 $\mathrm{EW}_s = \{\mathrm{ew}_{s,i} | \mathrm{ew}_{s,i} = g^{r_{s,i}} h^{w_{s,i}} \bmod q, r_{s,i} h_c^{t_{s,i}} \bmod q, g^{t_{s,i}} \bmod q\}, i = \{1, \ldots, nL\}$. 随机水印是为了混淆真正的嵌入水印位置, 假装嵌入的水印, 实际没嵌入. 在下面的交互阶段将描述如何嵌入随机水印.

(2) IO 将 EW_o, EW_c 和 EW_s 组成长度为 $(n+2)L$ 的串, 再用密钥 K_{oc} 置乱, 生成 $\mathrm{EW}_{oc} = \{\mathrm{ew}_{oc,i} | \mathrm{ew}_{oc,i} = g^{r_{oc,i}} h^{w_{oc,i}} \bmod q,\ r_{oc,i} h_c^{t_{oc,i}} \bmod q,\ g^{t_{oc,i}} \bmod q\}$, $i = \{1, \ldots, (n+2)L\}$.

(3) IO 将 EW_{oc} 发送给 CS.

(4) CS 接收到 EW_{oc}, 并计算出 $r_{oc,i} = \frac{r_{oc,i} h_c^{t_{oc,i}}}{(g^{t_{oc,i}})^{\theta_c}} \bmod q$, 以便在交互阶段使用 $r_{oc,i}$.

4.2.2 交互阶段

(1) IO 拥有一幅图像 X, 将 X 进行 8×8 的分块, 按照文献 [2] 中的方式选取 $(n+2)L$ 个块进行 DCT 变换, 将 (4,5), (5,4) 位置的原数值分别记为 A_i, B_i. 调整 A_i 和 B_i 的大小, 使得 $A_{i,2} > B_{i,2}$, 同时使得 $A_{i,1} < B_{i,1}$. 计算原数值与调整后数值的比值: $d_{i,1,1} = \frac{A}{A_{i,1}}$, $d_{i,1,2} = \frac{B}{B_{i,1}}$, $d_{i,2,1} = \frac{A}{A_{i,2}}$, $d_{i,2,2} = \frac{B}{B_{i,2}}$, 计算调整后数值之间的比值: $d_{i,1,3} = \frac{A_{i,2}}{A_{i,1}}$, $d_{i,1,4} = \frac{B_{i,2}}{B_{i,1}}$, $d_{i,2,3} = \frac{A_{i,1}}{A_{i,2}}$, $d_{i,2,4} = \frac{B_{i,1}}{B_{i,2}}$, 其中 $i = \{1, 2, \ldots, (n+2)L\}$. 准备 $d_{i,1,1}$, $d_{i,1,2}$, $d_{i,2,1}$ 和 $d_{i,2,2}$ 是为了在图像取回阶段恢复出原数值; 准备 $d_{i,1,3}$, $d_{i,1,4}$, $d_{i,2,3}$ 和 $d_{i,2,4}$ 是为了在图像分享阶段当 CS 与 IU 在相同位置上的水印的取值不同时, CS 能用已知的数值对求出 CS 未知的数值对. 例如, 当 $W_{u,i} = W_{c,i} = 1$ 时, CS 能直接获得 $(A_{i,1}, B_{i,1})$; 但当 $W_{u,i} = 2, W_{c,i} = 1$ 时, CS 需借助 $d_{i,1,3}$ 和 $d_{i,1,4}$ 求出 $(A_{i,2}, B_{i,2})$.

(2) 在嵌入随机水印的位置, IO 将所有的 d 值取 1, 记 $A_{i,1}=A_{i,2}=A_i$, $B_{i,1}=B_{i,2}=B_i$, 即在嵌入随机水印的位置调整后的数值对都是原数值对, 无论选 $(A_{i,1},B_{i,1})$ 还是 $(A_{i,2},B_{i,2})$ 都没嵌入水印.

(3) IO 选取 $k_{i,j},l_{i,j}\in\mathbb{Z}_q^*$, $i=\{1,2,\ldots,(n+2)L\}$, $j=\{1,\ldots,6\}$, 计算两组密文 $c_{i,1}$, $c_{i,2}$ 如下:

$$c_{i,1}=\left\{\begin{array}{c} A_{i,1}\left(\dfrac{\mathrm{ew}_{oc,i}}{h}\right)^{k_{i,1}};\ g^{k_{i,1}};\ B_{i,1}\left(\dfrac{\mathrm{ew}_{oc,i}}{h}\right)^{k_{i,2}};\ g^{k_{i,2}};\ d_{i,1,1}{h_o}^{k_{i,3}};\ g^{k_{i,3}};\\ d_{i,1,2}{h_o}^{k_{i,4}};\ g^{k_{i,4}};\ d_{i,1,3}{h_o}^{k_{i,5}};\ g^{k_{i,5}};\ d_{i,1,4}{h_o}^{k_{i,6}};\ g^{k_{i,6}} \end{array}\right\},$$

$$c_{i,2}=\left\{\begin{array}{c} A_{i,2}\left(\dfrac{\mathrm{ew}_{oc,i}}{h^2}\right)^{l_{i,1}};\ g^{l_{i,1}};\ B_{i,2}\left(\dfrac{\mathrm{ew}_{oc,i}}{h^2}\right)^{l_{i,2}};\ g^{l_{i,2}};\ d_{i,2,1}{h_o}^{l_{i,3}};\ g^{l_{i,3}};\\ d_{i,2,2}{h_o}^{l_{i,4}};\ g^{l_{i,4}};\ d_{i,2,3}{h_o}^{l_{i,5}};\ g^{l_{i,5}};\ d_{i,2,4}{h_o}^{l_{i,6}};\ g^{l_{i,6}} \end{array}\right\}.$$

IO 将加密的置乱密钥 $E_{\mathrm{pk}_o}(K_{oc})$, $c_{i,1},c_{i,2}$, $i=\{1,2,\ldots,(n+2)L\}$ 以及剩余的明文 DCT 系数发送给 CS.

(4) CS 存储 $E_{\mathrm{pk}_o}(K_{oc})$, 用在准备阶段获得的 $r_{oc,i}$ 解密 $c_{i,1}$, $c_{i,2}$, 获得含 CS 水印的 DCT 系数.

假设 $w_{oc,i}=1$ 时, CS 可计算出 $A_{i,1}=\frac{c_{i,1,1}}{(g^{k_{i,1}})^{r_{oc,i}}}$, $B_{i,1}=\frac{c_{i,1,3}}{(g^{k_{i,2}})^{r_{oc,i}}}$, 但无法计算出 $A_{i,2}$ 和 $B_{i,2}$, 因为 $\frac{c_{i,2,1}}{(g^{l_{i,1}})^{r_{oc,i}}}=\frac{A_{i,2}}{h^{l_{i,1}}}\neq A_{i,2}$, $\frac{c_{i,2,3}}{(g^{l_{i,2}})^{r_{oc,i}}}=\frac{B_{i,2}}{h^{l_{i,2}}}\neq B_{i,2}$. 同理, 假设 $w_{oc,i}=2$ 时, CS 可计算出 $A_{i,2}=\frac{c_{i,2,1}}{(g^{l_{i,1}})^{r_{oc,i}}}$, $B_{i,2}=\frac{c_{i,2,3}}{(g^{l_{i,2}})^{r_{oc,i}}}$, 但无法计算出 $A_{i,1}$ 和 $B_{i,1}$, 因为 $\frac{c_{i,1,1}}{(g^{k_{i,1}})^{r_{oc,i}}}=A_{i,1}h^{k_{i,1}}\neq A_{i,1}$, $\frac{c_{i,1,3}}{(g^{k_{i,2}})^{r_{oc,i}}}=B_{i,1}h^{k_{i,2}}\neq B_{i,1}$. CS 利用明文 DCT 系数和解密出的 $(A_{i,1},B_{i,1})$ 或 $(A_{i,2},B_{i,2})$ 进行逆 DCT 变换, 获得含 CS 水印的图像.

CS 通过比较 $(A_{i,1},B_{i,1})$ 或 $(A_{i,2},B_{i,2})$ 数值对的大小关系, 推出 $w_{oc,i}$ 的取值是 1 还是 2, 并存储 $c_{i,1}$ 或 $c_{i,2}$ 的后 8 项密文数据, 如下所示:

若 $w_{oc,i}=1$, 存储 $\{d_{i,1,1}{h_o}^{k_{i,3}};g^{k_{i,3}};d_{i,1,2}{h_o}^{k_{i,4}};g^{k_{i,4}};d_{i,1,3}{h_o}^{k_{i,5}};g^{k_{i,5}};d_{i,1,4}{h_o}^{k_{i,6}};g^{k_{i,6}}\}$.

若 $w_{oc,i}=2$, 存储 $\{d_{i,2,1}{h_o}^{l_{i,3}};g^{l_{i,3}};d_{i,2,2}{h_o}^{l_{i,4}};g^{l_{i,4}};d_{i,2,3}{h_o}^{l_{i,5}};g^{l_{i,5}};d_{i,2,4}{h_o}^{l_{i,6}};g^{l_{i,6}}\}$.

在图像上传阶段, CS 的水印是密文状态, 大部分图像的 DCT 系数以明文的状态传输到 CS, 而用来嵌入水印的 DCT 系数与 CS 的密文水印经过不经意传输协议 [6] 到 CS.

4.3 图像取回阶段

(1) 若 $w_{oc,i}=1$, CS 计算 $R_i=\{A_{i,1}\cdot d_{i,1,1}(h_o)^{k_{i,3}};g^{k_{i,3}};B_{i,1}\cdot d_{i,1,2}(h_o)^{k_{i,4}};g^{k_{i,4}}\}$; 若 $w_{oc,i}=2$, CS 计算 $R_i=\{A_{i,2}\cdot d_{i,2,1}(h_o)^{l_{i,3}};g^{l_{i,3}};B_{i,2}\cdot d_{i,2,2}(h_o)^{l_{i,4}};g^{l_{i,4}}\}$, 记 $R_i=\{R_{i,1};R_{i,2};R_{i,3};R_{i,4}\}$. CS 将明文 DCT 系数以及 $R_i,i=\{1,2,\ldots,(n+2)L\}$ 返还给 IO.

(2) IO 用私钥 θ_o 解密出原数值 $A=\frac{R_{i,1}}{(R_{i,2})^{\theta_o}}$ 和 $B=\frac{R_{i,3}}{(R_{i,4})^{\theta_o}}$. 然后, 将所有 DCT 系数进行逆 DCT 变换, 获得原始图像块.

4.4 图像分享阶段

4.4.1 准备阶段

(1) IO 选取 $r_{o,i},t_{o,i}\in\mathbb{Z}_q^*$, 用 IU 的公钥 h_u 加密 IO 水印, 获得 $\mathrm{EW}_o=\{\mathrm{ew}_{o,i}|\mathrm{ew}_{o,i}=g^{r_{o,i}}h^{w_{o,i}}\bmod q,\ r_{o,i}h_u^{t_{o,i}}\bmod q, g^{t_{o,i}}\bmod q\}$, $i=\{1,\ldots,L\}$; 生成由 {1,2} 组成的长度为 nL 的串作为随机水印 W_s; 然后, 选取 $r_{s,i},t_{s,i}\in\mathbb{Z}_q^*$, 用 h_u 加密随机水印获得 $\mathrm{EW}_s=\{\mathrm{ew}_{s,i}|\mathrm{ew}_{s,i}=g^{r_{s,i}}h^{w_{s,i}}\bmod q,\ r_{s,i}h_u^{t_{s,i}}\bmod q, g^{t_{s,i}}\bmod q\}$, $i=\{1,\ldots,nL\}$.

(2) IO 将 EW_o, EW_u 和 EW_s 组成长度为 $(n+2)L$ 的串, 再用密钥 K_{ou} 置乱, 获得 $\mathrm{EW}_{ou}=\{\mathrm{ew}_{ou,i}|\mathrm{ew}_{ou,i}=g^{r_{ou,i}}h^{w_{ou,i}}\bmod q,\ r_{ou,i}h_u^{t_{ou,i}}\bmod q,\ g^{t_{ou,i}}\bmod q\}$, $i=\{1,\ldots,(n+2)L\}$.

(3) IO 将置乱密钥 $E_{\mathrm{pk}_o}(K_{ou})$ 发送给 CS.

(4) IU 从 CS 下载 EW_{ou}, 并计算出 $r_{ou,i}=\frac{r_{ou,i}h_u^{t_{ou,i}}}{(g^{t_{ou,i}})^{\theta_u}} \bmod q$, 以便在交互阶段使用 $r_{ou,i}$.

4.4.2 交互阶段

(1) CS 选择随机数 $x_{i,j},y_{i,j}\in\mathbb{Z}_q^*$, $i=\{1,2,\ldots,(n+2)L\}$, $j=\{1,\ldots,4\}$.

当 $w_{oc,i}=1$ 时, CS 只能计算出与 $A_{i,1}$ 和 $B_{i,1}$ 有关的密文 $c_{i,1}$, $m_{i,1}$, 如下:

$$c_{i,1}=\left\{\left(A_{i,1}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{x_{i,1}},g^{x_{i,1}}\right);\left(B_{i,1}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{x_{i,2}},g^{x_{i,2}}\right)\right\},$$

$$m_{i,1}=\left\{\left(A_{i,1}\cdot d_{i,1,3}h_o^{k_{i,5}}\cdot\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{x_{i,3}},g^{k_{i,5}},g^{x_{i,3}}\right);\left(B_{i,1}\cdot d_{i,1,4}h_o^{k_{i,6}}\cdot\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{x_{i,4}},g^{k_{i,6}},g^{x_{i,4}}\right)\right\}.$$

当 $w_{oc,i}=2$ 时, CS 只能计算出与 $A_{i,2}$ 和 $B_{i,2}$ 有关的密文 $c_{i,2}$, $m_{i,2}$, 如下:

$$c_{i,2}=\left\{\left(A_{i,2}\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{y_{i,1}},g^{y_{i,1}}\right);\left(B_{i,2}\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{y_{i,2}},g^{y_{i,2}}\right)\right\},$$

$$m_{i,2}=\left\{\left(A_{i,2}\cdot d_{i,2,3}h_o^{l_{i,5}}\cdot\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{y_{i,3}},g^{l_{i,5}},g^{y_{i,3}}\right);\left(B_{i,2}\cdot d_{i,2,4}h_o^{l_{i,6}}\cdot\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{y_{i,4}},g^{l_{i,6}},g^{y_{i,4}}\right)\right\}.$$

若 $w_{ou,i}=w_{oc,i}$, IU 可解密 $c_{i,1}$ 或 $c_{i,2}$. 但当 $2=w_{ou,i}\neq w_{oc,i}=1$ 时, IU 不能解密 $c_{i,1}$, 因为 $c_{i,1}$ 中缺少与 $A_{i,2}$ 和 $B_{i,2}$ 有关的密文. 为了增加 $c_{i,1}$ 中与 $A_{i,2}$ 和 $B_{i,2}$ 有关的密文, CS 需借助 IO 解密 $m_{i,1}$, 因为 $m_{i,1}$ 含有被 IO 的公钥加密的 $A_{i,2}$ 和 $B_{i,2}$. 但这个过程不能向 IO 暴露 $A_{i,2}$ 和 $B_{i,2}$ 的值, 防止 IO 知道 CS 的水印, 也不能向 CS 暴露 $A_{i,2}$ 和 $B_{i,2}$ 的值, 防止 CS 用 $A_{i,2}$ 和 $B_{i,2}$ 替换原有的 $A_{i,1}$ 和 $B_{i,1}$, 更改原嵌入的水印. CS 用密钥 K_1 置乱由 $m_{i,1}$ 构成的集合获得 M_1. 同理, 当 $1=w_{ou,i}\neq w_{oc,i}=2$ 时, $c_{i,2}$ 中缺少与 $A_{i,1}$ 和 $B_{i,1}$ 有关的密文. 为此, CS 用密钥 K_2 置乱由 $m_{i,2}$ 构成的集合获得 M_2. CS 将 M_1 和 M_2 发送给 IO, 请求 IO 解密.

(2) IO 选取 $z_{i,j}\in\mathbb{Z}_q^*$, $i=\{1,2,\ldots,(n+2)L\}$, $j=\{1,2\}$, 根据 M_1 计算 $g^{x_{i,3}}\cdot g^{z_{i,1}}=g^{x_{i,3}+z_{i,1}}$, $A_{i,1}\cdot d_{i,1,3}h_o^{k_{i,5}}\cdot(\frac{\mathrm{ew}_{ou,i}}{h^2})^{x_{i,3}}\cdot(g^{k_{i,5}})^{-\theta_o}\cdot(\frac{\mathrm{ew}_{ou,i}}{h^2})^{z_{i,1}}=A_{i,2}(\frac{\mathrm{ew}_{ou,i}}{h^2})^{x_{i,3}+z_{i,1}}$; $g^{x_{i,4}}\cdot g^{z_{i,2}}=g^{x_{i,4}+z_{i,2}}$, $B_{i,1}\cdot d_{i,1,4}h_o{}^{k_{i,6}}\cdot(\frac{\mathrm{ew}_{ou,i}}{h^2})^{x_{i,4}}\cdot(g^{k_{i,6}})^{-\theta_o}\cdot(\frac{\mathrm{ew}_{ou,i}}{h^2})^{z_{i,2}}=B_{i,2}(\frac{\mathrm{ew}_{ou,i}}{h^2})^{x_{i,4}+z_{i,2}}$. 这里若不添加指数 $z_{i,1}$, CS 可利用 $(\frac{\mathrm{ew}_{ou,i}}{h^2})^{x_{i,3}}$ 从 $A_{i,2}(\frac{\mathrm{ew}_{ou,i}}{h^2})^{x_{i,3}}$ 中求出 $A_{i,2}$; 同理, $z_{i,2}$ 可防止 CS 计算出 $B_{i,2}$. 简化指数形式, 令 $x_{i,3}+z_{i,1}=x'_{i,3}$, $x_{i,4}+z_{i,2}=x'_{i,4}$, 所得序列的每一项为 $m_{i,1}$, 如下:

$$m_{i,1}=\left\{\left(A_{i,2}\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{x'_{i,3}},g^{x'_{i,3}}\right);\left(B_{i,2}\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{x'_{i,4}},g^{x'_{i,4}}\right)\right\}.$$

选取 $p_{i,j}\in\mathbb{Z}_q^*$, $i=\{1,2,\ldots,(n+2)L\}$, $j=\{1,2\}$, 根据 M_2 计算 $g^{y_{i,3}}\cdot g^{p_{i,1}}=g^{y_{i,3}+p_{i,1}}$, $A_{i,2}\cdot d_{i,2,3}h_o{}^{l_{i,5}}\cdot(\frac{\mathrm{ew}_{ou,i}}{h})^{y_{i,3}}\cdot(g^{l_{i,5}})^{-\theta_o}\cdot(\frac{\mathrm{ew}_{ou,i}}{h})^{p_{i,1}}=A_{i,1}(\frac{\mathrm{ew}_{ou,i}}{h})^{y_{i,3}+p_{i,1}}$; $g^{y_{i,4}}\cdot g^{p_{i,2}}=g^{y_{i,4}+p_{i,2}}$, $B_{i,2}\cdot d_{i,2,4}h_o{}^{l_{i,6}}\cdot(\frac{\mathrm{ew}_{ou,i}}{h})^{y_{i,4}}\cdot(g^{l_{i,6}})^{-\theta_o}\cdot(\frac{\mathrm{ew}_{ou,i}}{h})^{p_{i,2}}=B_{i,2}(\frac{\mathrm{ew}_{ou,i}}{h})^{y_{i,4}+p_{i,2}}$. 这里添加指数 $p_{i,1}$ 和 $p_{i,2}$ 是为了防止 CS 计算出 $A_{i,1}$ 和 $B_{i,1}$.

简化指数形式, 令 $y_{i,3}+p_{i,1}=y'_{i,3}$, $y_{i,4}+p_{i,2}=y'_{i,4}$, 所得序列的每一项为 $m_{i,2}$, 如下:

$$m_{i,2}=\left\{\left(A_{i,1}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{y'_{i,3}},g^{y'_{i,3}}\right);\left(B_{i,1}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{y'_{i,4}},g^{y'_{i,4}}\right)\right\}.$$

IO 发送 M_1 和 M_2 给 CS.

(3) CS 用密钥 K_1, K_2 分别逆置乱 M_1, M_2, 利用 $c_{i,1}$ 与 $m_{i,2}$ 构成 $u_{i,1}$, 并统一 $u_{i,1}$ 的指数为 x, 利用 $c_{i,2}$ 与 $m_{i,1}$ 构成 $u_{i,2}$, 并统一 $u_{i,2}$ 的指数为 y. $u_{i,1}$, $u_{i,2}$ 的形式如下:

$$u_{i,1}=\left\{\left(A_{i,1}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{x_{i,1}},g^{x_{i,1}}\right);\left(B_{i,1}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{x_{i,2}},g^{x_{i,2}}\right)\right\},$$
$$u_{i,2}=\left\{\left(A_{i,2}\left(\frac{\mathrm{ew}_{ou,i}}{h^2}\right)^{y_{i,1}},g^{y_{i,1}}\right);\left(B_{i,2}\left(\frac{\mathrm{ew}_{ou,i}}{h}\right)^{y_{i,2}},g^{y_{i,2}}\right)\right\}.$$

CS 将 $u_{i,1}$, $u_{i,2}$, $i=\{1,2,\ldots,(n+2)L\}$ 与剩余的明文 DCT 系数发送给 IU.

(4) IU 利用 $r_{ou,i}$ 解密 $u_{i,1}$ 或 $u_{i,2}$, 其过程与上传阶段中 CS 解密 $c_{i,1}$ 或 $c_{i,2}$ 一样, 获得含 IU 水印的 DCT 系数. IU 利用明文 DCT 系数与解密出的 $(A_{i,1},B_{i,1})$ 或 $(A_{i,2},B_{i,2})$, 进行逆 DCT 变换, 获得含 IU 水印的图像.

在图像分享阶段, IU 的水印是密文状态, 大部分图像的 DCT 系数以明文的状态发送给 IU, 而用来嵌入水印的 DCT 系数在传输中是密文.

4.5 水印检测阶段

如果 IO 发现图像 X 的一个可疑副本 Y, IO, CS 和 J 通过检测 CS 的水印是否存在于 Y 来判断是否是 CS 泄漏了 Y. 若能从 Y 中检测出 CS 的密文水印 EW_c, 则判定 Y 是从 CS 中流出. 检测副本 Y 是否是 IU 泄露的也是一样的步骤, 只是把 CS 的水印换成 IU 的水印, 置乱密钥 K_{oc} 换成 K_{ou}.

(1) IO 将版权水印 W_o、可疑副本 Y 和 K_{oc} 发送给 J.

(2) J 向 CS 索要 $g^{r_{c,i}}$, $i=\{1,2,\ldots,L\}$.

(3) J 将 Y 分成 8×8 的块, 按照文献 [2] 中的方式选取 $(n+2)L$ 个块进行 DCT 变换, 将 $(4,5),(5,4)$ 位置的数值分别记为 A_i 和 B_i, 根据其数值的大小关系, 推算出 $w'_{oc,i}, i=\{1,2,\ldots,(n+2)L\}$. 当 $A_i>B_i$ 时, $w'_{oc,i}=2$. 当 $A_i<B_i$ 时, $w'_{oc,i}=1$. 然后, 用 K_{oc} 逆置 $W'_{oc,i}$, 获得 W'_o, W'_c 和 W'_s.

(4) 若 W_o 与 W'_o 相匹配, 则 J 断定 Y 的版权属于 IO.

(5) J 计算 $g^{r_{c,i}}\cdot h^{w'_{c,i}}$ 并判断其结果是否等于 $\mathrm{ew}_{c,i}$; 若相等, 说明可疑 Y 中含有 CS 的水印, 是 CS 泄漏了 Y.

5 安全性分析

5.1 云端水印和图像使用者水印的隐私

在与 CS 交互后, IO 不会知道 CS 获得的是哪一个消息. IO 所知道的数据是 g, q, $\mathbb{G}$, A_1, B_1, A_2, B_2, CS 的密文水印 EW_c 以及 EW_c 的构成方式 (EW_c 是由 g 的 r 次幂与 h 或 h^2 的乘积构成的), 但 IO 不知道 CS 随机选择的指数 r_c, 从而也就无法从 EW_c 中分离出 h^{w_c}, 所以 IO 无法确定 w_c 是 1 还是 2. 同样, IO 和 CS 也无法知道 IU 水印 w_u 的取值.

5.2 图像版权人数据的隐私

(1) 图像上传阶段. 在与 IO 交互后, CS 仅仅知道它选择的那个消息 $(A_{i,1},B_{i,1})$ 或 $(A_{i,2},B_{i,2})$, 无法获得另一个消息 $(A_{i,2},B_{i,2})$ 或 $(A_{i,1},B_{i,1})$. CS 在协议中所知道的数据是 g, h, q, $\mathbb{G}$, $c_{i,1}$, $c_{i,2}$ 以及 $c_{i,1}$, $c_{i,2}$ 的构成方式, 但并不知道 IO 随机选取的指数 $k_{i,j},l_{i,j}\in\mathbb{Z}_q^*$, $i=\{1,2,\ldots,(n+2)L\}$, $j=\{1,\ldots,6\}$, 从而无法从另一个密文中恢复出另一个消息. 当 $w_{oc,i}=1$ 时, CS 可计算出 $A_{i,1}$, $B_{i,1}$, $\frac{A_{i,2}}{h^{l_{i,1}}}$, $\frac{B_{i,2}}{h^{l_{i,2}}}$. 当 $w_{oc,i}=2$ 时, CS 可计算出 $A_{i,2}$, $B_{i,2}$, $A_{i,1}h^{k_{i,1}}$, $B_{i,1}h^{k_{i,2}}$. 因 $l_{i,1}$, $l_{i,2}$, $k_{i,1}$, $k_{i,2}$ 是 IO 随机选取的, CS 无法从 $\frac{A_{i,2}}{h^{l_{i,1}}}$, $\frac{B_{i,2}}{h^{l_{i,2}}}$, $A_{i,1}h^{k_{i,1}}$, $B_{i,1}h^{k_{i,2}}$ 中分离 $A_{i,2}$, $B_{i,2}$, $A_{i,1}$, $B_{i,1}$.

表 5 理论代价

Table 5 Theoretical cost

Computation cost	Upload phase	Retrieval phase	Sharing phase
IO	$36\|q\|(n+2)L$	$3\|q\|(n+2)L$	$18\|q\|(n+2)L$
CS	$12\|q\|(n+2)L$	–	$24\|q\|(n+2)L$
IU	–	–	$12\|q\|(n+2)L$
Communication cost	Upload phase	Retrieval phase	Sharing phase
IO	$24\|q\|(n+2)L$	–	$4\|q\|(n+2)L$
CS	–	$4\|q\|(n+2)L$	$15\|q\|(n+2)L$
IU	–	–	–
Storage cost	Upload an image	Add a new IO	Add a new IU
CS	$8\|q\|(n+2)L$	$3\|q\|(n+2)L$	$3\|q\|(n+2)L$

如果 CS 知道离散对数 $\log_g h = t$, 那么 CS 可以恢复出另一个消息, CS 就可以攻击本协议. 当 $w_{oc,i} = 1$ 时, CS 可计算出 $A_{i,2} = \frac{A_{i,2}}{h^{l_{i,1}}} \cdot (g^{l_{i,1}})^t$, $B_{i,2} = \frac{B_{i,2}}{h^{l_{i,2}}} \cdot (g^{l_{i,2}})^t$; 当 $w_{oc,i} = 2$ 时, CS 可计算出 $A_{i,1} = \frac{A_{i,1}h^{k_{i,1}}}{(g^{k_{i,1}})^t}$, $B_{i,1} = \frac{B_{i,1}h^{k_{i,2}}}{(g^{k_{i,2}})^t}$. 所以 CS 不能知道 $\log_g h = t$.

(2) 图像分享阶段. 为了增加 $c_{i,1}$ 中与 $A_{i,2}$ 和 $B_{i,2}$ 有关的密文, CS 需借助 IO 解密 $m_{i,1}$, 但这个过程不能向 CS 暴露 $A_{i,2}$ 和 $B_{i,2}$, 否则 CS 便知道另一对数值. 同样, 在 CS 借助 IO 解密 $m_{i,2}$ 的过程中 IO 不能向 CS 暴露 $A_{i,1}$ 和 $B_{i,1}$. 为此, IO 在解密 $m_{i,1}$, $m_{i,2}$ 时选用 $z_{i,j}, p_{i,j} \in \mathbb{Z}_q^*$, $i = \{1, 2, \ldots, (n+2)L\}$, $j = \{1, 2\}$ 作为指数重新加密解密后的数据, 再将结果返回给 CS. 因 CS 不知道 $z_{i,j}, p_{i,j}$, CS 无法获得 $(A_{i,2}, B_{i,2})$ 或 $(A_{i,1}, B_{i,1})$.

IU 在与 CS 交互后, IU 也仅仅知道他选择的那个消息 $(A_{i,1}, B_{i,1})$ 或 $(A_{i,2}, B_{i,2})$, 无法获得另一个消息 $(A_{i,2}, B_{i,2})$ 或 $(A_{i,1}, B_{i,1})$. 其原因与图像上传阶段中 CS 无法获得另一个消息一样. 同样, IU 不能知道 $\log_g h = t$.

6 性能分析

6.1 理论代价

本小节统计了图像上传、取回阶段和分享阶段的各个参与者理论上的计算代价和通信代价. 因图像上传阶段和图像分享阶段的准备工作只进行一次且与上传图像的内容无关, 可在任意时间进行, 故未统计准备阶段的消耗.

假设一个幂运算的指数的长度为 $|q|$, 则该幂运算指数需要 $1.5|q|$ 次乘法 [29] (例如, r 的长度是 $|q|$, 计算 g^r 需要 $1.5|q|$ 次乘法). 由于求幂运算比乘法运算的开销大得多, 我们在分析中忽略了固定次数的乘法运算. IO, CS 和 IU 在图像上传阶段、取回阶段和分享阶段的理论计算代价、通信代价, 以及 CS 的存储代价记录在表 5 中.

从表 5 的前两行看到, 计算代价与通信代价与 $|q|$ 的大小、随机水印的个数 n, 以及水印长度 L 成正比; IO 的计算代价和传输代价都高于 CS, 这是因为 IO 提前在图像上传阶段为图像的取回与分享阶段做了准备工作, 也导致图像上传阶段的代价均高于取回与分享阶段的代价; CS 在取回阶段的计算代价为无, 这是因为理论计算代价只统计指数运算, 而 CS 在该阶段只进行乘法运算没有指数运算. 从表 5 的后两行看到 CS 的存储代价与 $|q|$ 的大小、随机水印的个数 n, 以及水印长度 L 成正比; 同时,

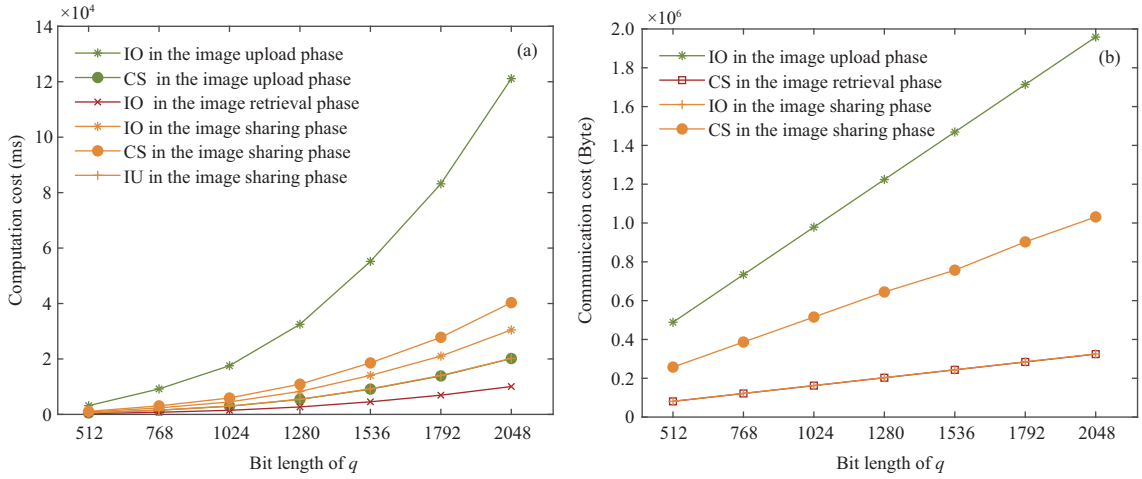

图 2　(网络版彩图) (a) 实验中参与者的计算代价; (b) 实验中参与者的通信代价

Figure 2　(Color online) (a) Experimental computation cost of participants; (b) experimental communication cost of participants

CS 的存储量随着上传图像的数量、IO 数量, 以及 IU 数量的增加而增加.

6.2　实验结果

在这一部分, 我们先统计了图像上传、取回阶段和分享阶段的各个参与者实际计算代价和实际通信代价. 接着, 评估水印的性能包括含水印明文图像的视觉质量、水印的提取错误率、水印的抗屏摄性分析. 最后, 对比解密后的含水印图像与文献 [2] 中明文域含水印图像的差别.

(1) 计算代价与通信代价. 本实验使用的个人电脑配置为 Intel (R) Core (TM) i5-4490@3.30 GHz 处理器, 8 GB RAM 内存和 Windows 7 专业操作系统; 使用 Java 中构建的自定义模拟器来评估所提出的协议中的计算代价和通信代价, 其数值是 1000 次实验结果的平均值; 在 Matlab R2014b 中完成对图像明文数据的处理.

本实验构成水印的方式与文献 [2] 相同, 本实验设置单个水印即 W_o, W_c, W_u, W_s 的长度都是 13 bit. 在图像上传阶段, 水印 W_{oc} 由 W_o, W_c, W_s 构成, 其长度为 39 bit; 在嵌入前经过纠错编码扩充到 64 bit, 再进行加密, 重复嵌入 5 次, 实际嵌入的总量为 320 bit. 在图像分享阶段, 水印 W_{ou} 由 W_o, W_u, W_s 构成, 其长度是 39 bit, 也是经过纠错编码扩充到 64 bit, 再进行加密, 重复嵌入 5 次, 总共嵌入 320 bit.

选用 512×512 大小的灰度图像, 利用 SIFT 算法定位特征点并筛选出 5 个大小为 64×64 区域, 每个区域进行 8×8 块的 DCT 变换, 取每个块的 $(4,5)$ 和 $(5,4)$ 的数值为 A 和 B, 扩大 $Q = 2^{16}$ 倍后取整, 调整大小获得 A_1, B_1, A_2, B_2. 最后根据协议进行交互, 协议中设置 $L = 64$, $n = 1$, 统计参与者的计算代价于图 2(a), 通信代价于图 2(b).

图 2(a) 和 (b) 中参与者的计算代价和通信代价均随着 $|q|$ 的增加而增加, 这是因为模乘和指数运算增加随着模数 $|q|$ 的增加; 由于 $|q|$ 比特数的增加, 传输比特数也在增加. 从图 2 看到 IO 在上传图像阶段的计算代价和通信代价是最高的, 这是因为 IO 为了取回图像和分享图像进行了大量的前期计算; CS 在图像分享阶段扮演图像上传阶段 IO 的角色, CS 在该阶段承担了主要的计算和通信工作, 这使得 CS 在图像分享阶段的消耗高于 IO 和 IU 在该阶段的消耗.

图 3 (a) 上传阶段的图像; (b) 分享阶段的图像; (c) 取回阶段的图像

Figure 3 (a) Image in the upload phase; (b) images in the sharing phase; (c) image in the retrieval phase

(2) 水印性能.

含水印图像的质量. 本实验用峰值信噪比 (peak signal to noise ratio, PSNR) 评价含水印图像的质量. 以 512×512 大小的图像 Lena 为例, 本实验展示在无攻击情况下图像上传、取回、分享阶段的图像, 如图 3(a)~(c) 所示.

从图 3 看出, 上传阶段和分享阶段明文图像的 PSNR 值大于 30, 对人眼来说是可接受的. 取回的图像已经接近原图, 这是因为 DCT 系数是小数, 在量化为整数的过程中, 带来了少量的损失.

水印的提取错误率. 本实验提取水印比特的错误率 (bit error rate, BER) 评价提取水印效果. 以 512×512 大小的图像 Lena 为例, 在无攻击情况下图像上传阶段的水印 BER 为 0, 图像分享阶段的 BER 也为 0, 这说明本协议中的密文水印解密到明文状态后能被正确提取.

水印的抗屏摄性分析. 本文在密文域实现文献 [2] 中的抗屏摄算法, IO 将密文水印嵌入图像, 将生成含水印的密文图像, 并发送给 CS 或 IU, CS 或 IU 解密后获得含水印的图像. 本文需要在解密后含水印的图像上测试水印的抗屏摄性, 文献 [2] 的作者对不同距离不同角度拍摄的含水印图像进行了测试, 证明水印算法具有抗屏摄性. 本文对比解密后含水印图像与文献 [2] 中含水印图像的差别, 若二者的差别小到可以忽略, 说明本协议中的水印和文献 [2] 中的水印一样都能抗屏摄攻击.

在图像上传阶段, 本文统计含水印 W_{oc} 的 DCT 系数与文献 [2] 中含水印 W_{oc} 的 DCT 系数差值的绝对值之和为 0.0016; 在图像分享阶段, 统计本文含水印 W_{ou} 的 DCT 系数与文献 [2] 中含水印 W_{ou} 的 DCT 系数差值的绝对值之和为 0.0097; 这说明虽然水印以密文的形式嵌入, 但在解密回到明文状态后的数据与文献 [2] 直接嵌入明文水印后的数据是一样的, 本协议具有文献 [2] 中水印的抗屏摄性.

7 讨论

存储在云端中明文图像包含了云的身份水印, 因此不能将存在云中的图像展示在各个用户的终端. 在本协议中, 显示在 IO 的终端的是取回后的图像, 不包含任何水印, IO 可下载该图像到本地; 在 IU 的终端展示的是分享后的图像, 已包含 IU 的身份水印. 即使 IU 通过拍摄获取图像, 但因该图像中嵌入了能抗屏摄的水印, 屏摄图像仍含有 IU 的水印, 可指证 IU 的偷拍行为.

本文设置 IO, CS, IU 水印的长度都是 13 bit. 实际上, 云服务商的数量通常远少于用户数量, 用户指 IO 和 IU, 因此可为用户分配更多的比特数, 以支持更多的用户. 只要组合水印 W_{oc} 和 W_{ou} 在纠错编码前的长度都为 39 bit, 可保证文中水印的抗屏摄性与文献 [2] 中水印的抗屏摄性相同. 随着用户数

量的增加, 组合水印 W_{oc} 和 W_{ou} 在纠错编码前的长度超过 39 bit, 可将组合水印分割为 39 bit 一组; 若组数增加, 这就需要更多的嵌入块、更大的图像, 才能达到文献 [2] 中水印的抗屏摄性. 本文为了实验设置与文献 [2] 一致, 采用长为 512 像素, 宽为 512 像素的图像, 而目前手机拍摄照片的长宽通常为几千像素, 所以在现实应用中可以支持更长的水印信息.

8 总结

本文针对支持上传、取回、分享、可浏览功能的云存储场景, 提出云与用户之间交互式抗屏摄密文水印协议. 嵌入抗屏摄的密文水印一方面保证水印能抵抗屏摄攻击, 发现图像使用者泄露屏摄图像的行为; 另一方面消除了泄露源的二义性, 保证含水印的明文图像唯一指向用该水印代表身份的图像使用者. 实验结果显示协议中的计算消耗在毫秒级, 说明本文中的协议具有一定的可用性. 图像拥有者为了监控云端和图像使用者对图像的使用, 在上传图像阶段进行了较多的计算. 如何保证图像拥有者的图像数据不被泄露同时减少图像拥有者的计算量是我们接下来的工作方向.

参考文献

1 He S Y. Research on copyright protection in cloud storage services. Dissertation for Master's Degree. Kunlin: Yunnan University, 2018 [何诗奕. 云存储服务中的版权保护研究. 硕士学位论文. 昆明: 云南大学, 2018]
2 Fang H, Zhang W M, Zhou H, et al. Screen-shooting resilient watermarking. IEEE Trans Inform Forensic Secur, 2019, 14: 1403–1418
3 Fang H, Chen D D, Huang Q D, et al. Deep template-based watermarking. IEEE Trans Circ Syst Video Technol, 2021, 31: 1436–1451
4 Chen W T, Zhu C Q, Ren N, et al. Screen-cam robust and blind watermarking for tile satellite images. IEEE Access, 2020, 8: 125274
5 Chen W T, Ren N, Zhu C Q, et al. Screen-cam robust image watermarking with feature-based synchronization. Appl Sci, 2020, 10: 7494
6 Tzeng W G. Efficient 1-out-n oblivious transfer schemes. In: Proceedings of International Workshop on Practice & Theory in Public Key Cryptosystems: Public Key Cryptography, 2002
7 Sadeghi A R, Shi Y Q, Kim H J, et al. The marriage of cryptography and watermarking-beneficial and challenging for secure watermarking and detection. In: Proceedings of International Workshop on Digital Watermarking, 2007. 2–18
8 Cox I J, Kilian J, Leighton F T, et al. Secure spread spectrum watermarking for multimedia. IEEE Trans Image Process, 1997, 6: 1673–1687
9 Chen B, Wornell G W. Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. IEEE Trans Inform Theor, 2001, 47: 1423–1443
10 Seungwoo H, Hakjae K, Sungju L, et al. Analyzing the secure and energy efficient transmissions of compressed fingerprint images using encryption and watermarking. In: Proceedings of International Conference on Information Security and Assurance, 2008. 316–320
11 Metkar S P, Lichade M V. Digital image security improvement by integrating watermarking and encryption technique. In: Proceedings of IEEE International Conference on Signal Processing, Computing and Control, 2013. 1–6
12 Arumugam S, Annadurai K. A hybrid of watermark scheme with encryption to improve security of medical images. In: Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, 2021
13 Chen W T, Ren N, Zhu C Q, et al. Joint image encryption and screen-cam robust two watermarking scheme. Sensors, 2021, 21: 701
14 Zhang X P. Separable reversible data hiding in encrypted image. IEEE Trans Inform Forensic Secur, 2012, 7: 826–832
15 Wang X, Chang C C, Lin C C. Reversible data hiding in encrypted images with block-based adaptive MSB encoding. Inf Sci, 2021, 567: 375–394
16 Xiong L Z, Han X, Yang C N, et al. Robust reversible watermarking in encrypted image with secure multi-party based on lightweight cryptography. IEEE Trans Circ Syst Video Technol, 2022, 32: 75–91
17 Memon N D, Wong P W. A buyer-seller watermarking protocol. IEEE Trans Image Process, 2001, 10: 643–649
18 Rial A, Deng M, Bianchi T, et al. A provably secure anonymous buyer-seller watermarking protocol. IEEE Trans Inform Forensic Secur, 2010, 5: 920–931

19 Dong X J, Zhang W M, Hu X J, et al. A cloud-user watermarking protocol protecting the right to be forgotten for the outsourced plain images. Int J Digital Crime Foren, 2018, 10: 118–139
20 Liu K Y, Zhang W M, Dong X J. A cloud-user protocol based on ciphertext watermarking technology. Secur Commun Netw, 2017, 2017: 1–14
21 Dong X J, Zhang W M, Shah M, et al. Watermarking-based secure plaintext image protocols for storage, show, deletion and retrieval in the cloud. IEEE Trans Serv Comput, 2022, 15: 1678–1692
22 Liu X, Deng R H, Choo K K R, et al. An efficient privacy-preserving outsourced calculation toolkit with multiple keys. IEEE Trans Inform Forensic Secur, 2016, 11: 2401–2414
23 Li S D, Wang W L, Du R M. Protocol for millionaires' problem in malicious models. Sci Sin Inform, 2021, 51: 75–88 [李顺东, 王文丽, 杜润萌. 抗恶意敌手的百万富翁问题解决方案. 中国科学: 信息科学, 2021, 51: 75–88]
24 Erkin Z, Franz M, Guajardo J, et al. Privacy-preserving face recognition. In: Proceedings of Privacy Enhancing Technologies, International Symposium, 2009
25 Paillier P. Public-key cryptosystems based on composite degree residuosity classes. In: Proceedings of International Conference on Theory and Application of Cryptographic Techniques, 1999. 223–238
26 Dong X J, Zhang W M, Shah M, et al. A restrained paillier cryptosystem and its applications for access control of common secret. 2019. ArXiv:1912.09034
27 Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inform Theory, 1985, 31: 469–472
28 Zhang W M, Yu N H, Fang H. Digital watermarking method robust to the screen-shooting process. China Patent, CN109886856A, 2019-06-14 [张卫明, 俞能海, 方涵. 对屏幕拍摄过程鲁棒的数字水印方法. 中国专利, CN109886856A, 2019-06-14]
29 Knuth D E. The Art of Computer Programming, vol. 2. Seminumerical Algorithms. Gottingen: Atmospheric Chemistry & Physics, 1981

Interactive screen-shooting resilient ciphertext watermarking protocol between the cloud and the user

Xiaojuan DONG, Weiming ZHANG*, Han FANG & Nenghai YU

Key Laboratory of Electromagnetic Space Information, Chinese Academy of Sciences, University of Science and Technology of China, Hefei 230026*, China*
* Corresponding author. E-mail: zhangwm@ustc.edu.cn

Abstract The current cloud storage services generally launch the function of browsing files, which brings great convenience for users to steal information by shooting files on the screen, and also increases the risk of information leakage. As a result, users taking photos to steal information poses a potential security risk for cloud storage services. The existing method is for the cloud to embed a screen-shooting resilient watermark associated with the user's identity in an image before sending it to the user. Then the cloud can detect the user's watermark from the captured image and discover the user's identity. Both the cloud and the user have the image embedded with the user's watermark, so the leaked images cannot be confirmed to be from the cloud or from the user. To eliminate the ambiguity of the leakage source, ciphertext watermark is utilized, that is, the user's ciphertext watermark is embedded into a plaintext image in the cloud to generate the ciphertext image containing the user's watermark, which can only be decrypted by the user to obtain the plaintext image. Once this image is secretly shot and leaked, it must flow out from the user. However, the current screen-shooting resilient watermarking algorithm does not support the above functions and only supports the embedding of the plaintext watermark. This paper proposes a screen-shooting resilient ciphertext watermarking protocol between the cloud and users, which aims to protect the copyright users' ownership of their images and prevent other users who use the images from stealing images by taking pictures secretly. This paper verifies the security and effectiveness of the proposed protocol through theoretical analysis and simulation experiments.

Keywords cloud storage, ciphertext watermark protocol, screen-shooting resilient watermark, copyright protection, leakage tracking