Go-Sharing: A Blockchain-Based Privacy-Preserving Framework for Cross-Social Network Photo Sharing

Ming Zhang, *Student Member, IEEE*, Zhe Sun, *Member, IEEE*, Hui Li, *Member, IEEE*, Ben Niu, *Member, IEEE*, Fenghua Li, *Member, IEEE*, Zixu Zhang, *Student Member, IEEE*, Yuhang Xie, *Student Member, IEEE*, and Chunhao Zheng, *Student Member, IEEE*

Abstract—The evolution of social media has led to a trend of posting daily photos on online Social Network Platforms (SNPs). The privacy of online photos is often protected carefully by security mechanisms. However, these mechanisms will lose effectiveness when someone spreads the photos to other platforms. In this article, we propose Go-sharing, a blockchain-based privacy-preserving framework that provides powerful dissemination control for cross-SNP photo sharing. In contrast to security mechanisms running separately in centralized servers that do not trust each other, our framework achieves consistent consensus on photo dissemination control through carefully designed smart contract-based protocols. We use these protocols to create platform-free dissemination trees for every image, providing users with complete sharing control and privacy protection. Considering the possible privacy conflicts between owners and subsequent re-posters in cross-SNP sharing, we design a dynamic privacy policy generation algorithm that maximizes the flexibility of re-posters without violating formers' privacy. Moreover, Go-sharing also provides robust photo ownership identification mechanisms to avoid illegal reprinting. It introduces a random noise black box in a two-stage separable deep learning process to improve robustness against unpredictable manipulations. Through extensive real-world simulations, the results demonstrate the capability and effectiveness of the framework across a number of performance metrics.

Index Terms—Photo sharing, privacy-preserving, cross-SNP, blockchain

1 Introduction

The number of photos being posted on various Social Network Platforms (SNPs) is increasing exponentially. More than 120 million active Instagram users will upload photos daily,[1] and more than 567 million photos are uploaded to Facebook[2] every day. This widespread photo sharing causes privacy concerns [2]. Users frequently attempt to repost or upload photos from other users without permission. This type of illegal republishing and photo snatching causes individual privacy breaches, which could potentially lead to dangers like stalking, sexual predators, etc [3], [4].

Some research attempts to circumvent these issues and increase user privacy protection. For instance, Privacy Respecting-based Sharing schemes [5], [6] have been proposed to protect users' privacy by requiring photo service providers (or picture takers) to enforce privacy policies. Several access control mechanisms [7], [8], [9], [10], [11] have been designed to restrict visiting. Privacy-preserving content-based image retrieval scheme [12] has been proposed to protect the privacy in part content of photos during image database and CBIR service outsourcing. Despite providing single access security, these mechanisms do not effectively solve the subsequent transfer. Cryptographic methods [13], [14], [15] have also been used to encrypt photos and limit their dissemination. Some of them [15] utilized deep learning methods to identify sensitive images and encrypt them by an attribute-based encryption scheme. Others [13], [14] restricted the photo sharing further through key distributions. Besides these techniques, photo obfuscation [16], [17]

- *Ming Zhang, Hui Li, Yuhang Xie, and Chunhao Zheng are with the State Key Laboratory of Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an 710071, China. E-mail: {xidianmingz, chunhaoz101}@gmail.com, {lihui, yhxie}@mail.xidian.edu.cn.*
- *Zhe Sun is with the Cyberspace Institute of Advanced Technology (CIAT), Guangzhou University, Guangzhou 510006, China. E-mail: sunzhe@gzhu.edu.cn.*
- *Ben Niu and Fenghua Li are with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100864, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: {niuben, lfh}@iie.ac.cn.*
- *Zixu Zhang is with the School of Electrical and Data Engineering, University of Technology Sydney, Sydney, NSW 2007, Australia. E-mail: zixu.zhang@student.uts.edu.au.*

Manuscript received 29 March 2022; revised 10 September 2022; accepted 20 September 2022. Date of publication 23 September 2022; date of current version 1 September 2023.
The work of Hui Li's was supported in part by the National Natural Science Foundation of China under Grants U1836203 and 61932015, in part by Shaanxi innovation team project under Grant 2018TD-007 and in part by Higher Education Discipline Innovation 111 project under Grant B16037. The work of Ben Niu's was supported in part by the the National Natural Science Foundation of China under Grant 61872441. The work of Zhe Sun's was supported in part by the National Natural Science Foundation of China under Grant 62002077, and in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2020A1515110385.
(Corresponding authors: Zhe Sun and Hui Li.)
Digital Object Identifier no. 10.1109/TDSC.2022.3208934

1. https://www.statista.com/topics/1882/instagram/
2. https://www.garyfox.co/social-media-statistics

has also been commonly used to protect strangers' privacy. e. t., [18] proposed a trust-based privacy-preserving mechanism for multi-owner photo sharing that anonymize the potential high privacy loss users in the photo. [19] used adversarial perturbation approaches against high accuracy face-recognizing in multi-owner photo sharing. Even so, these studies primarily focus on privacy preservation within a single platform, which does not address the cross-SNP sharing problem. As a result, they are not suitable to be extended to cross-SNP photo-sharing scenarios.

Unlike single-platform sharing, cross-SNP sharing poses many privacy challenges. First, since user data cannot be synchronized between databases across platforms, SNPs do not know whether a photo has been previously uploaded to other SNPs before. Nor can they identify the original owner of newly uploaded photos outside the platform. A second challenge is that users cannot impose any effective restrictions on privacy leakage during cross-SNP photo dissemination. This is because the existing privacy protection mechanism is independent and unaffiliated, which makes the support for user privacy policies across different platforms incompatible. One typical example is respecting the user's right have their data deleted. General Data Protection Regulations (GDPR) [20] stipulate that uploaded images should not be shared after the user has chosen to have their data deleted. At present, although platforms can remove the images of a user, they cannot prevent the circulation of these images reposted on other platforms. Additionally, widespread dissemination will likely involve many users from different SNPs, each with their own privacy demands for sharing, which can lead to privacy conflicts between different sharers. It is difficult to take care of all privacy preferences in every dissemination.

In light of the above issues, we propose Go-Sharing: a blockchain-based privacy-preserving framework for cross-SNP sharing. Instead of building a unified and synchronized database, we propose a set of smart contract-based protocols to achieve practical cross-SNP sharing control. Platforms in Go-Sharing do not need to expose all their data to meet the security requirements of users in cross-SNP sharing. To ensure that every cross-SNP share is under their owner's control, we build a platform-free dissemination tree for every photo. The dissemination tree takes the original owner as the root node and appends a new child node each time a repost or cross-platform upload occurs. This tree structure ensures that every subsequent share will be permanently affected by its previous dissemination node. We also design a dynamic policy generation algorithm to address the privacy conflict problem. By dynamically calculating an appropriate privacy policy for every dissemination, Go-Sharing satisfies the current forwarder's needs without violating privacy of previous users. To identify the original owner of newly uploaded photos outside the platform, we design an ownership identification scheme that watermarks every shared photo with its dissemination tree root address. To resist unpredictable modification in cross-SNP sharing, we also introduce a black box of random noise into a Two-stage Separable Deep Learning (TSDL) process to strengthen robustness further. Briefly, the main contributions of this paper are summarized as follows:

- We propose a novel cross-SNP photo sharing framework named Go-sharing by designing a series of smart contract-based protocols to guarantee the secure and effective control on cross-SNP sharing.
- We design a fine-grained multi-owner dissemination tree to keep photo owners' effective management during cross-SNP sharing. Additionally, we propose a dynamic privacy policy generation algorithm to settle the privacy conflict problem between multiple sharers.
- We propose a robust photo ownership identification scheme by introducing a random noise black box in TSDL to enhance its capability against unpredictable modifications in cross-SNP sharing.
- We conduct a preference study covering 533 participants on photo sharing and compile a user privacy preference set. We use this to evaluate performance of our proposed framework in real world. The result shows that our framework can serve as a high-utility solution for privacy-preserving across SNPs and photo sharing.

The remainder of this paper is organized as follows. In Section 2 we review the related work. Section 3 presents the system model, threat model and security assumptions of Go-sharing. In Section 4 we propose specific photo sharing schemes across social networks. Section 5 presents performance evaluation results. We conclude in Section 6.

2 Related Work

2.1 Privacy-Preserving in Photo Sharing

Photo sharing in online social networks is a typical privacy-preserving scenario and has hence received a lot of attention [21], [22]. Xu et al. [18] addressed the privacy loss of users sharing photos by tuning the anonymization threshold with a greedy method. K. Xu et al. [11] addressed the potential privacy leak of side users by proposing an efficient mechanism to recognize all individuals in a photo which allows them to participate in the decision-making process. Li et al. in [23] improved Xu's scheme to protect the privacy of bystanders in a photo. They proposed an algorithm that identifies bystanders (using the distance between the people) and then mosaicing their faces. However, all these solutions only consider the privacy of people in a photo. They do not avoid the abuse of the photo itself. To overcome this issue, some researchers attempted to encrypt photos when sharing them online. Sun et al. [24] investigated Facebook photo uploading and proposed a DCT-domain photo encryption/decryption framework to improve photo sharing security. Wang et al. [25] proposed a chaotic image encryption algorithm based on the matrix semi-tensor product with a compound secret key to secure color image sharing. [26] proposed a class of sorting matrices with fractal characteristics, which is termed as a fractal sorting matrix (FSM). Based on the FSM and global chaotic pixel diffusion, the author constructs a new chaotic photo encryption algorithm to further improve photo encryption efficiency. These encryption-based mechanisms give a high level of security to photo protection, but they are not suitable for the large scale of photo sharing in cross-SNP scenarios. In Go-Sharing, we propose a platform-free dissemination tree structure to protect photo sharing in cross-SNP scenarios.

Taking advantage of the immutable properties of the blockchain, some authors have proposed blockchain-based online sharing systems [27], [28]. R. Yu et al. in [29] proposed a blockchain-based social networking platform to implement news subscription notifications and friend recommendations. Rahman et al. in [30] proposed a decentralized online resource sharing management model based on Ethereum. These works deploy the blockchain inside SNPs, and is implemented as data storage module that replaces the traditional databases. Although these solutions can solve the problem of single-point dependencies in single-platform sharing, they do not contribute to the more complex issue of cross-SNP photo sharing. In our work, we implement a blockchain-based cross-SNP photo sharing framework that fully achieves automatic image dissemination control without any centralized server.

In addition to sharing security, multiparty privacy conflicts (MPCs) [31] are a large problem in online photo sharing [32]. The Universal Declaration of Human Rights [33] clarifies that everyone's right to privacy should be protected by law. Although the United States [34] and the European Union [20] have introduced specific regulations for privacy protection, they do not provide adequate protection, especially when considering the current high production and sharing rate of multimedia content. Technical solutions are necessary to reduce the undesirable effects of MPCs. Some studies have attempted to address MPCs through precautionary mechanisms, such as aggregated voting [35], [36], [37] and game theory [38], [39]. However, they rely on the accuracy of face recognition. Cherubini et al. in [31] proposed a dissuasive mechanism to minimize the occurrence of MPCs through supporting uploaders' decision making about sharing co-owned content. Mosca in [40] created an AI-based collaborative privacy decision model that used individual privacy preferences and social values to shape the negotiation process to an agreed sharing policy. [41], [42] presented a fully explainable personal assistant that identifies the optimal sharing policy for collectively owned content. Unfortunately, due to their centralized design, they cannot synchronize users' privacy data securely across other platforms. Hence these solutions cannot be used to resolve multi-owner privacy conflicts across platforms. More importantly, the above schemes only define people inside a photo as the owners and attempt to resolve their privacy conflicts via various decision models. In fact, due to different sharing intentions, the subsequent forwarders, who act as one of the sharing controllers, may also have privacy conflicts with the photo's original owner.

In our work, we address the privacy conflicts among cross-SNP photo sharing dissemination by proposing a dynamic policy generation algorithm.

2.2 Robust Photo Ownership Identification

Photo ownership refers to the attribution right of online photos. The sharing decisions of co-owned photos can violate the privacy of others [43]. In cross-platform scenarios, Go-Sharing focuses on protecting the photo dissemination ownership during the cross-SNP sharing process. As mentioned in Section 1, the SNP's server database does not have any information about the posted photo, including its sharing records or history in other platforms. Verifying the photo's identity is hence necessary for authentication before accepting a new upload. Since cross-SNP forwarding is often associated with modification of a photo, a robust photo ownership identification scheme is needed to resist these possible modifications.

Currently there are mainly three methods in ownership identification: perceptual hashing, data hiding, and watermarking. The perceptual hashing identifies photo ownership and authentication through perception digests based on the understanding of photo content [44]. This method is easily affected by differences in visual features [45] and thus is not robust enough to handle possible modifications in cross-SNP sharing. On the other hand, since data hiding [46], [47] mainly focus on reducing quality loss to improve concealment from attackers, they are also not robust enough to handle the unpredictable noise in cross-SNP sharing. The last methods can be divided into non-blind watermarking and blind watermarking. The recovery of the non-blind watermark requires the original image, which is not acquired in the cross-SNP sharing scenarios. As for blind watermarking, they are mainly concerned with two properties [48]: the quality of the watermarked image and the robustness of the watermark. Some watermarking methods encode information in the least significant bits of image pixels [49] or frequency domain [50], [51]. To achieve a higher level of robustness, Zhu et al. in [52] proposed a convolutional neural network-based watermarking method that uses the convoluted image's feature information to co-encode with the message. [48] further improved the accuracy by proposing a TSDL framework that uses the end-to-end training method to train the encoder in the first stage while fixing the encoder unchanged and fully exploits the potential of the decoder in the second stage. However, these models only perform well against specific noises and cannot resist unknown modification attacks.

Therefore, in this paper, we design a random noise black box that generates multiple random combinations of noise in each training batch to enhance our ownership protection model's capability of resistance to unknown noise.

3 System Model

In this section, we present the system model of the framework we proposed, and give the threat models based on a series of security assumptions. For illustration purpose, we list the notations that may be used in this paper in Table 1.

3.1 Decentralized Go-Sharing System Model

To achieve the goal of secure cross-SNP sharing, we use a decentralized architecture as shown in Fig. 1, consisting of four components: Photo Owner, Photo Visitor, Social Network Platform, and Decentralized Sever. The specific system model is shown in Fig. 2.

Photo Owner, identified by $O = \{O_1, \ldots, O_i, \ldots, O_n\}$, posts his photo to the SNP (e.g., SNP 1) with a specific corresponding photo privacy $Policy_P = (AG_P, FG_P, DG_P, \ RUG_P)$, where AG_P, FG_P, DG_P, and RUG_P refer to Visiable, Forwardable, Downloadable and Re-uploadable groups of photo P, respectively. We regard a photo forwarding or re-uploading operation with no loss of generality as photo dissemination. According to the attribution of the photo, we further divided O into two categories as O_P^o and O_P^s, $(O = \{O_P^o \cup$

TABLE 1
The Notations of Explanation

Notation	Explanation
$O = \{O_1, \ldots, O_i, \ldots, O_n\}$	Set of photo owners, $i = 1, \ldots, n$
$V = V_1, \ldots, V_i, \ldots, V_m$	Set of photo visitors, $i = 1, \ldots, m$
$SNP = \{SNP_1, \ldots, SNP_s\}$	Set of SNPs, $i = 1, \ldots, s$
P	The photo posted by O
$Policy_P$	The policy of photo P
VG_P, FG_P, DG_P, RUG_P	Visiable, Forwardable, Downloadable and Reuploadable groups of photo P
$Ov(P)$	The ownership value of the photo P
O_P^o, O_P^s	The origin owner and subordinate owner of the photo P
$Pointer(Policy_P)$	Pointer to the policy storage address
$Pointer(P)$	Pointer to the photo storage address
T	The current timestamp

$O_P^s\}$), where O_P^o refers to the original owner who first upload this photo online and act as the root of the photo's dissemination tree. While O_P^s relates to the subsequent users who disseminate the photos and act as a dissemination node. As an owner, both O_P^o and O_P^s can set their own privacy policy $Policy_P$ in their uploading, which will implicitly affect subsequent photo sharing.

Photo Visitor, identified by $V = \{V_1, \ldots, V_i, \ldots, V_m\}$, requests to access (including visit and download) shared photos from SNPs (e.g., SNP 1) through other SNPs (e.g., SNP 2 and SNP 3).

Social Network Platform, identified by $SNP = \{SNP_1, \ldots, SNP_i, \ldots, SNP_s\}$, consists of existing third-party social networking platforms. SNPs play the role of a communication medium throughout the system and are charged with implementing the access results after delivering users' requests to the decentralized server with the extraction of $Ov(P)$. Go-Sharing does not ask for any photo data or user profiles through cross-SNP photo sharing. On the contrary, all shared photos and user profiles are still stored in the SNPs' respective databases.

Decentralized Server consists of a distributed blockchain and provides core dissemination control functions for all SNPs. As a decentralized server, it uses smart contracts to handle the requests from different SNPs uniformly. Each request is considered as a transaction that will be packaged and recorded on the blockchain and synchronized to every peer node in the whole federation through the broadcast mechanism.

As suggested in [53] the process of ownership attribution should be anonymous. In Go-Sharing, although the attribution information uploaded to the blockchain is not encrypted for performance reasons, all user information involved is invoked in the form of address pointers. In addition, smart contracts operate independently as a black box in the closed docker of the blockchain. Therefore, no information about the user's attribution will be leaked during the process. Besides, all SNPs in Go-Sharing involved are seen as an organization and connected with each other through a channel with a shared super ledger. Smart contracts are deployed on the peers of each organization to process sharing operation requests. Thus, in Go-sharing, it is no longer possible for malicious visitors to get rid of the control of photo owner's policies by reposting.

3.2 Threat Model

To achieve their goals, potential malicious attackers may use different methods to maximize their profits unscrupulously. Here, we give the threat model, which illustrates several potential threats and malicious behaviors that may occur as follows:

Malicious Owner

1) *Attacker's ability:* Malicious owners are allowed to visit the sharing photos and publish photos on SNPs in Go-Sharing.
2) *Attacker's goal:* They attempt to download the original owner's online photo and disseminate it to other platforms without the original owner's permission while claiming to be the photo's creator. Through a series of possible manipulations like resizing, changing contrast, cropping and brightness of the photos, the malicious owner expects to destroy photos' ownership identity as well as get

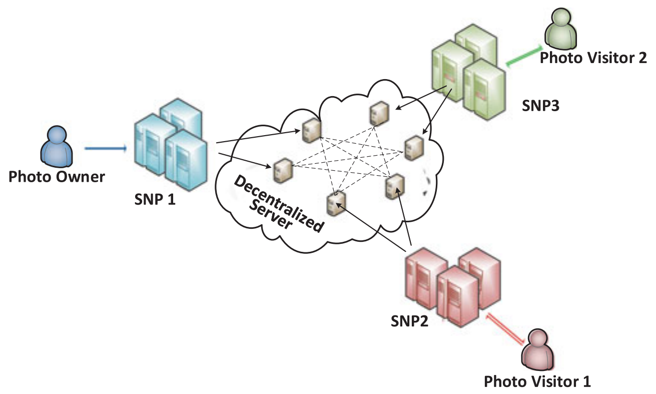


Fig. 1. System architecture of Go-Sharing.

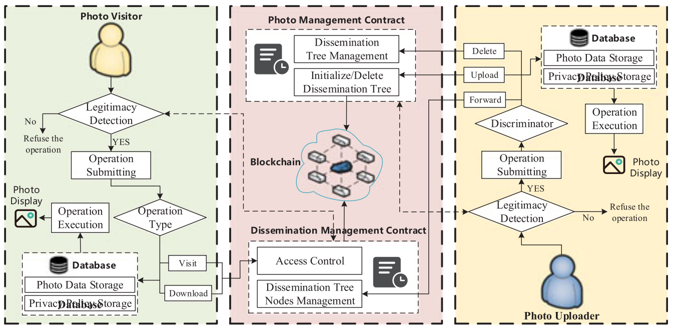
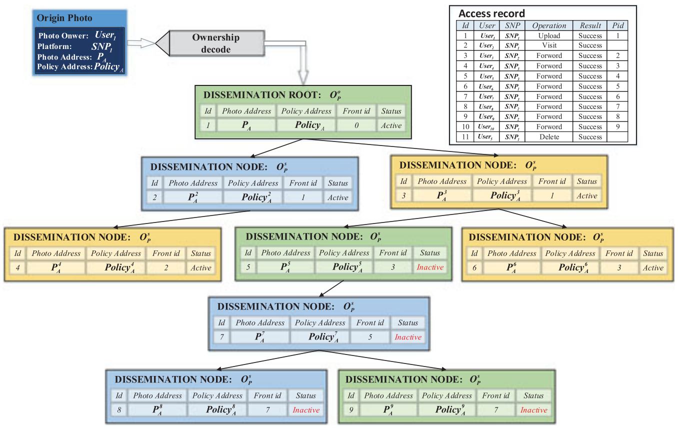


Fig. 2. System model of Go-sharing.


Origin Photo
Photo Onwer: $User_1$
Platform: SNP_1
Photo Address: P_A
Policy Address: $Policy_A$
Ownership decode
DISSEMINATION ROOT: O_P^o
Id | Photo Address | Policy Address | Front id | Status
1 | P_A | $Policy_A$ | 0 | Active
DISSEMINATION NODE: O_P^s
2 | P_A^2 | $Policy_A^2$ | 1 | Active
3 | P_A^3 | $Policy_A^3$ | 1 | Active
4 | P_A^4 | $Policy_A^4$ | 2 | Active
5 | P_A^5 | $Policy_A^5$ | 3 | Inactive
6 | P_A^6 | $Policy_A^6$ | 3 | Active
7 | P_A^7 | $Policy_A^7$ | 5 | Inactive
8 | P_A^8 | $Policy_A^8$ | 7 | Inactive
9 | P_A^9 | $Policy_A^9$ | 7 | Inactive

Access record

Id	User	SNP	Operation	Result	Pid
1	$User_1$	SNP_1	Upload	Success	1
2	$User_2$	SNP_1	Visit	Success	
3	$User_3$	SNP_2	Forword	Success	2
4	$User_4$	SNP_3	Forword	Success	3
5	$User_5$	SNP_3	Forword	Success	4
6	$User_6$	SNP_1	Forword	Success	5
7	$User_7$	SNP_3	Forword	Success	6
8	$User_8$	SNP_2	Forword	Success	7
9	$User_9$	SNP_2	Forword	Success	8
10	$User_{10}$	SNP_1	Forword	Success	9
11	$User_5$	SNP_1	Delete	Success	

Fig. 3. Platform-freed dissemination tree.

rid of the original owner's restrictions on the distribution and gain its complete control.

Malicious Visitor

1) *Attacker's ability:* Malicious visitors have the ability to log in to different SNPs and visit other users' social account.
2) *Attacker's goal:* They attempt to gain unauthorized access to others' photos from the different social networks, which may contain owners' significant sensitive information like the places of private gatherings, daily outdoor activities, and personal relationships. In other words, their goal is to get others' privacy-sensitive photos under unauthorized conditions and try to erase their own access and operation records to avoid auditing and traceability.

Malicious Social Networking Platforms

1) *Attacker's ability:* Malicious social networking platforms pretend as normal SNPs and can communicate normally with other SNPs.
2) *Attacker's goal:* By accessing online photo data from other SNPs' databases, they attempt to abuse the data. Data mining and analysis unscrupulously of the user's online photos may be carried out to obtain a large amount of sensitive private information, which users do not expect.

3.3 Security Assumption

Majority Peer Security (MPS). The security of Go-sharing is closely related to blockchain security. We assume that attackers (including malicious owners, malicious visitors, and malicious social networking platforms) cannot compromise the overall security of the blockchain network. In other words, the attackers do not have the ability to hold most of the computing power and network resources in the blockchain. Most of the nodes in the blockchain network are honest and reliable, and the messages are synchronized promptly and transmitted smoothly between each other.

Secure User Account (SUA). In Go-sharing, users use their respective accounts on SNPs to share and manipulate photos, so the privacy security of shared images is related to the security of users' accounts. We assume that the user's account on the social network is secure. An attacker cannot modify the user's privacy policy by stealing the user's account or directly downloading or deleting shared pictures by logging into the picture source account.

Secure Encryption Algorithm (SECA). We assume that Go-sharing uses secure symmetric and asymmetric cryptographic algorithms to implement encryption and decryption. In practical applications, users' shared photos and the corresponding privacy policy information are stored in ciphertext in the SNP.

4 A Concrete Implementation Of Go-Sharing

4.1 Platform-Freed Dissemination Tree

In cross-SNP sharing, the forwarding of online photos breaks through the platform's limitations, which expands the communication space from a single social network to multiple social networks. To help users control the entire lifecycle of their photos during sharing, we propose the architecture of the dissemination tree in Go-Sharing.

The dissemination tree is unique for each photo, which is created with the ownership sequence simultaneously during the first upload of the photo. The ownership sequence is used to specify the identity of the photo, which is the address of the dissemination tree root. So as long as the sequence of photo ownership is extracted correctly, Go-Sharing is capable of locating the uploaded photo within its dissemination tree in the social network.

As depicted in Fig. 3, the dissemination tree consists of a root node and several dissemination nodes, where the root and dissemination nodes can be located on different social platforms and connected based on forwarding relationships.

Dissemination Root is generated when O_P^o uploads a photo that has not been encoded with an ownership sequence in Go-Sharing before. As the source of photo dissemination, the root node is managed by the Photo Management Contract (PMC) in the global photo pool. O_P^o's privacy policy stored in its policy address determines the maximum dissemination scale of this photo.

Dissemination Node is generated when the subsequent dissemination user O_P^s gets a photo from another user and posts it to his social account. The dissemination node is managed by the Dissemination Management Contract (DMC) and establishes a directed edge pointing to the parent node through the front id. The frond id records the id of the dissemination node's parent node, which is marked by the DMC and is used to indicate the origin of this dissemination node. Without loss of generality, the dissemination node can be created mainly by the following two methods: forwarding and downloading&re-uploading. The former one is commonly performed with the aid of calling on the given function furnished by the SNP, while the latter is obtained through downloading or taking a screenshot of the photo and then uploading it directly to his account. Benefitting from the robustness of the ownership identification scheme, the dissemination sequence carried by the photo will not be erased no matter which method is used. Hence the dissemination node can still be created in the correct position in that photo's dissemination tree. As a participant in photo dissemination, the privacy policy carried by the

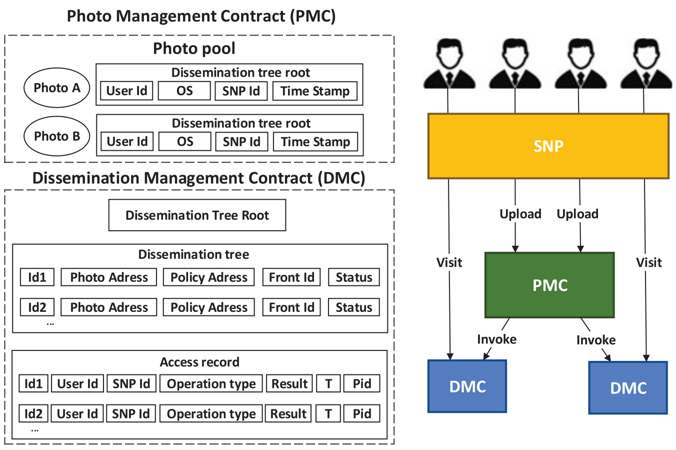


Fig. 4. The relationship and structure of smart contracts.

disseminating node also affects the scope of image dissemination. But this effect is backward and invalid for the previous nodes.

During the cross-SNP sharing, the dissemination tree performs a vital role in privacy-preservation. It provides a platform-freed photo dissemination control structure, permitting photo owner to have the whole control over the photo's subsequent forwarding. In Go-Sharing, any dissemination node receives real-time control from the smart contract. Through that the user can inactivate all following nodes while deleting the shared photo to put in force his right to have their data deleted. Meanwhile, the dissemination tree allows users to introduce individual privacy policies into the photo's dissemination with a clear distribution path, supplying a groundwork for resolving privacy conflicts mentioned in Section 3.

4.2 Go-Sharing Smart Contract

This section details the design and implementation of our proposed privacy-preserving cross-SNP photo sharing framework, in which all dissemination control is completed by decentralized smart contracts: Photo Management Contract (PMC) and Dissemination Management Contract (DMC). The relationship between these two smart contract structures is shown in Fig. 4 and the general interaction flow of the framework is depicted in Fig. 6.

4.2.1 Photo Management Contract (PMC)

PMC is only responsible for photo uploading, it holds a global photo pool which consists of all dissemination tree roots to record the online photo and their original owners' information. In PMC, every shared photo is only registered once in their first uploading. To reduce the load on the blockchain, we do not upload the source data of photos to the blockchain. At the same time, to protect users' privacy, users' detailed privacy policy will also not be uploaded to the blockchain. The information set that will be uploaded to the blockchain includes follows: uploading User id, SNP id for submitting, the Ownership Sequence (OS) of photos and Timestamp.

During upload, PMC first identifies the uploading photo through its OS, which is decoded in the SNP and delivered to the PMC with the upload request. To distinguish existing shared photos with unencoded new images, OS uses (-1,1) encoding. Every element of the sequence decoded from an unencoded image will be close to zero, while a valid OS will correctly be decoded like a -1,1 sequence, which correctly points to the photo's dissemination tree root in the global photo pool. Such uploading will be regarded as a dissemination of an existing photo, and then be delivered to DMC to check the legality and record this upload request as a new dissemination node. Otherwise, a new dissemination tree will be created, and the information of this uploader will be recorded in the photo pool as the tree root.

4.2.2 Dissemination Management Contract (DMC)

Unlike PMC, DMC focus on managing the subsequent sharing after the first upload. The access record listed by DMC includes access user id, SNP id, Operation type, Result, Timestamp, and Pid, in which the user id and SNP id are used to bind the visitor's identity information. The Operation type is an enumeration variable to mark the type of access with four optional values: Visiting, Uploading,

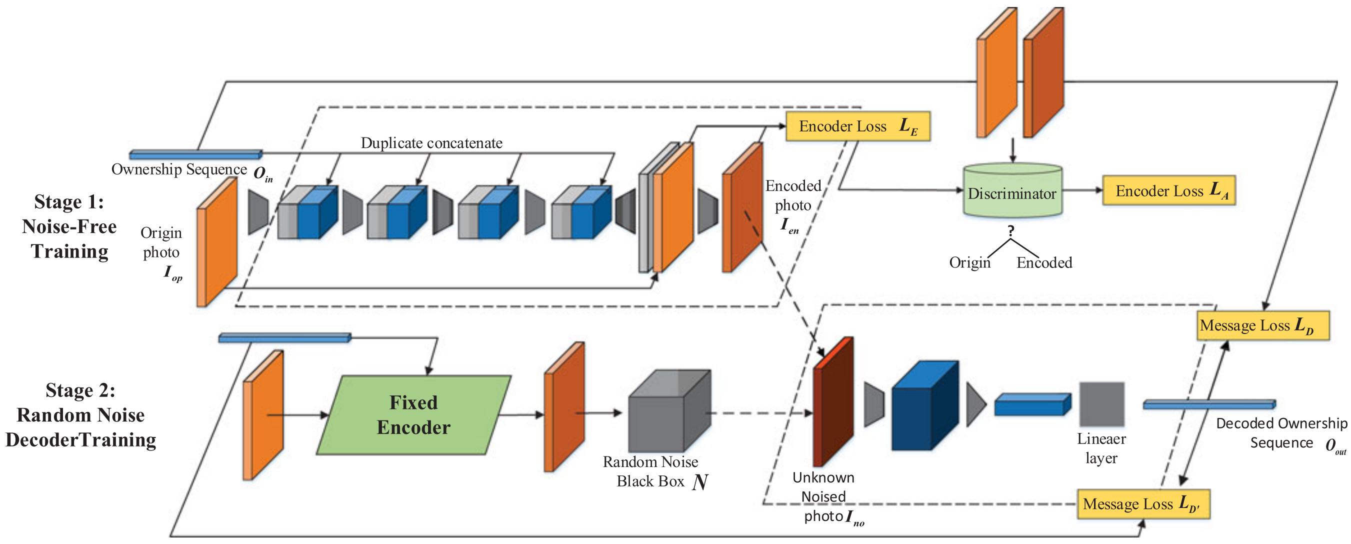


Fig. 5. Overall model architecture.

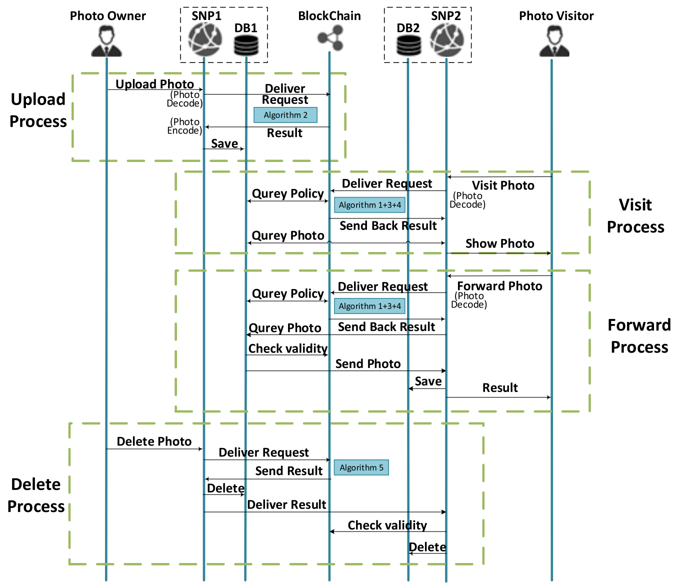


Fig. 6. The interactive flow of Go-sharing.

Forwarding, and Downloading. The Pid is an optional variable to link the corresponding record with the id of the newly created dissemination node if this access updates the dissemination tree and creates a new dissemination node.

```
Algorithm 1. Dynamic Privacy Policy Generation
Require: New dissemination node $N$, Tree root node $N_{root}$,
    New photo owner $O_P^s$, Raw policy of $O_P^s$ $policy_O$,
Ensure: The generated policy $policy_N$
 1: $N_{front} \leftarrow getFrontNode(N)$
 2: $policy_N \leftarrow \emptyset$
 3: while $N_{front} \neq N_{root}$ do
 4:    $Policy_F \leftarrow getPolicy(N_{front})$
 5:    for $P \in policy_O$ do
 6:       if $P \in policy_F$ then
 7:          $policy_N = policy_N \cup \{P\}$
 8:       end if
 9:    end for
10: end while
11: return $policy_N$
```

The DMC is in charge of every transmission of the photo on social networks and holds every dissemination node. Each dissemination node represents a photo owner and stores the node id, Photo Address, Policy Address, Front id, and Status on it. Photo Address and Policy Address are used to point to the photo storage and privacy policy storage in the new SNP. Front id is adopted to mark the parent node's id, which may be the same as other nodes since they may come from the same node.

4.3 Dynamic Privacy Policy Generation Algorithm

Both O_P^o and O_P^s have their own individual privacy preferences about photo sharing. Hence, privacy conflicts are likely to arise between the users from different dissemination nodes when they have opposite policies. In other words, the conflict arises when some users want to share photos with specific users, and others do not. Formally [54]:

Definition 1 (Privacy conflict). *Given a set of photo owners O with corresponding privacy policy $Policy_P$; O is said to be in conflict if $a, b \in O$, their individual policy $Policy_p^a$, $policy_p^b \in Policy_P$ satisfy*

$$Policy_p^a \cup Policy_p^b - Policy_p^a \cap Policy_p^b \neq \emptyset. \quad (1)$$

In Go-Sharing, we need a way to address the privacy conflict during the sharing between large quantity of photo owners and find an optimal privacy policy for each dissemination node that maximizes the current owner's sharing scope while satisfying all formers' privacy requirements.

4.3.1 Conflict Resolution

In previous privacy conflicts scenarios [32], [40] between photo content-involved users, all users are considered the parallel status, and their privacy preferences should be treated equally. But in photo dissemination, there exists a clear order of priority for privacy preferences between different users.

In the photo's dissemination tree, directed edges are used to represent the direction of dissemination. Specifically, it indicates that a user has acquired the photo from the previous node and subsequently disseminated it. In this process, if there is a privacy conflict, that means the originator's photo is spread to a place beyond his willingness. We do not say that the originator's privacy policies violate re-posters from sharing because they have different priorities where the reporter's distribution must adhere to the originator's privacy policies.

On the other hand, even if they share the same photo, the nodes in different branches of a dissemination tree do not have a mutual constraint relationship since they get the photo from a different dissemination path. As long as their dissemination does not violate the policy of the previous node, they can each disseminate it in a completely disjoint range. In fact, such privacy conflicts between nodes are reasonable and acceptable. We summarize the principles for handling privacy conflicts among different photo owners during dissemination as follows:

- The policy of any dissemination node in a dissemination tree cannot exceed the sharing scope of the user of the dissemination root node.
- For a node in a dissemination tree, if a reachable dissemination path exists from another node to it, then its privacy policy must be premised on the privacy policy of that node.
- For any two nodes in a dissemination tree, if there does not exist a reachable dissemination path between them, then, in any case, their privacy policies will not conflict with each other.

4.3.2 Policy Generation

To address the privacy conflict problem in dissemination process, we dynamically generate policies for each newly generated dissemination node that fine-tune the node's original policy to match the previous nodes' privacy expectations.

According to Definition 1, it is easy to see that conflict vanishes only if $Policy_p^a$ and $Policy_p^b$ have subordinate relationships or there exists an empty set between $Policy_p^a$ and $Policy_p^b$. Considering that it is not a wise choice to clear all privacy policy of subsequent nodes, we choose to address the privacy conflict by narrowly restricting the privacy policy of the post-forwarding user until the conflict disappears. Algorithm 1 concretely illustrates the dynamic policy generation process, in which we can see that the actual policy of a new dissemination node is the intersection of the current owner policy and the policies of all the front nodes in its dissemination path.

4.4 Random Noise Resistant Photo Ownership Identification Scheme

The above work provides a concrete dissemination control for cross-SNP photo dissemination. However, it is still far from our goal of achieving safe and secure cross-SNP photo sharing. The SNPs in Go-sharing can still easily be spoofed, and are still unable to associate the modified photo with its real owner. To solve this problem, in this section, we introduce a TSDL-based photo ownership identification scheme to provide secure sharing in our framework.

4.4.1 Model Architecture

Two-stage separable deep learning (TSDL) is a machine learning solution that aims to strengthen the decoder's robustness by separating the decoder's training from the encoder. As depicted in Fig. 5, the TSDL-based photo ownership identification model comprises four main components: (1) An encoder E takes the origin image $I_{op} \in \mathbb{R}^{C*H*W}$ (C, H, W refer to channel, height, and width, respectively) that uploaded first and its ownership sequence O_{in} calculated by the dissemination root node's address as input, and generates the encoded image $I_{en} \in \mathbb{R}^{C*H*W}$; (2) a random noise black box N tampered I_{en} with several unpredictable random noises with the output of the noised image I_{no}; (3) a decoder D receives the encoded image I_{in} or I_{no} and recovers the ownership sequence O_{out}; (4) an adversarial discriminator A that evaluates the probability that the given image is an encoded one.

Encoder. The encoder is trained to mask the first uploaded origin photo with a given ownership sequence as a watermark. In the encoder, the ownership sequence is first duplicate concatenated to expanded into a 3-dimension tensor $-1, 1_{L*H*W}$ (L refers to the length of the ownership sequence) and concatenated to the encoder's intermediary representation. Since the watermarking based on a convolutional neural network uses the different levels of feature information of the convoluted image to learn the unvisual watermarking injection, this 3-dimension tensor is repeatedly used to concatenate to every layer in the encoder and generate a new tensor $\in \mathbb{R}^{(C+L)*H*W}$ for the next layer. After multiple convolutional layers, the encode produces the encoded image I_{en}. To ensure the availability of the encoded image, the encoder should train to minimize the distance between I_{op} and I_{en}

$$\mathcal{L}_E = MSE(I_{op}, I_{en}) = MSE(I_{op}, E(I_{op}, O_{in})), \quad (2)$$

and the probability to detect an encoded image by the adversarial discriminator A

$$\mathcal{L}_G = \log(1 - A(I_{en})). \quad (3)$$

Decoder. The decoder consists of several convolutional layers, a global spatial average pooling layer, and a single linear layer, where convolutional layers are used to produce L feature channels while the average pooling converts them into the vector of the ownership sequence's size. Finally, the single linear layer produces the recovered ownership sequence O_{out}. It should be noted that the distribution of the recovered sequence indicates whether the image is encoded. If the $O_{out} \in \{0, 1\}^L$ rather than $\{-1, 1\}^L$, we say that this image is in its first uploading. To ensure the availability of the recovered ownership sequence, the decoder should train to minimize the distance between O_{in} and O_{out}

$$\mathcal{L}_D = MSE(O_{in}, O_{out}) = MSE(O_{in}, D(O_{in})). \quad (4)$$

Adversary Discriminator. The adversary discriminator has a similar structure to the decoder and outputs a binary classification. Acting as a critical role in the adversarial network, the adversary attempts to classify I_{en} from I_{op} correctly to prompt the encoder to improve the visual quality of I_{en} until it is indistinguishable from I_{op}. The adversary should train to minimize the following:

$$\mathcal{L}_A = \log(1 - A(I_{op})) + \log(A(E(I_{op}))). \quad (5)$$

Random Noise Black Box. In blind watermarking, recent works are often trained with fixed types of noise and parameters [52], [55]. Such models only work well for identical noise attacks while performing poorly against unpredictable random noise combinations attacks. Unfortunately, in cross-SNP sharing, photos are often subject to more than one type of unknown noise attack. Hence, to improve robustness, we design a random noise black box to simulate the unpredictable modifications during photo dissemination. Given an I_{en} as input, the random noise black box selects 0~3 types of processing as black-box noise attacks from Resize, Gaussian noise, Brightness&Contrast, Crop, and Padding to output the noised image I_{no}. Note that in addition to the type and the amount of noise, the intensity and parameters of the noise are also randomized to ensure the model we trained can handle any combination of noise attacks.

4.4.2 Two Stage Training

Due to the use of the random noise black box, the mainstream one-stage end-to-end training (OET) network will tend to degrade the quality of the encoder to ensure the accuracy of the O_{out}. To this end, in this paper, we use a two-stage separable deep learning network (TSDL) [48] that trains the encoder without noise in the first stage adversarial networks and then fixes the encoder while exploiting the potential of the decoder to cope with random noise in the second stage training.

Stage 1: Noise-Free Adversary Training. As shown in the upper left of Fig. 5, at the first training stage, both encoder, decoder, and adversary discriminator are trained in an end-to-end adversary network with the absence of noise, which is aiming to minimize

$$\mathcal{L}_1 = \lambda_E \mathcal{L}_E(I_{op}, I_{en}) + \lambda_D \mathcal{L}_D(O_{in}, O_{out}) + \lambda_A \mathcal{L}_A(I_{op}, I_{en}), \quad (6)$$

where λ_E, λ_D, and λ_A control the relative weights of the losses.

Stage 2: Random Noise Decoder Training. After getting a well-trained encoder in stage 1 training, we fix the parameters of the encoder and focus on tapping the ability of the decoder. In this stage, the random noise black box is introduced to simulate the realistic lossy photo dissemination scene, producing irregular noise combinations to enhance the decoder's recovery of ownership sequence from the noised photo by minimizing the recover sequence loss $\mathcal{L}_D$.

4.5 Interaction Flow of Secure Cross-SNP Dissemination

In this section, we specify the algorithm for implementing the Go-sharing function that consists of four main parts: photo upload, photo access or download, photo forwarding or secondary upload, and photo deletion. To illustrate the interactions of the objects of each component of the Go-sharing during the execution of the above algorithm, we present the general interaction flow in Fig. 6.

4.5.1 Photo Uploading

Assuming the user O_A uploads the shared photo P_A to SNP_1 with the corresponding $Policy_P = \{AG_P, FG_P, RG_P, RUG_P\}$ to state its visible users. Instead of P_A and $Policy_P$, SNP_1 delivers O_A's upload request with the corresponding storage pointers $Pointer(P_A)$ and $Pointer(Policy_P)$. Meanwhile, SNP_1 decodes $OS(P_A)$, which is also being submitted to the blockchain. The existence of $OS(P_A)$ in the photo pool is used to help determine whether it has been uploaded before. Finally, the complete request sent to SNP_1 is formed as: $R_D = \{O_A, SNP_1, Pointer(P_A), Pointer(Policy_P), OS(P_A)\}$. Algorithm 2 illustrates the implementation process of image uploads.

Algorithm 2. Upload Photo

```
Require: Photo owner $O_A$, Social Network Platform $SNP_1$,
    pointer of $P_A$ $Pointer(P_A)$, pointer of privacy policy
    $Pointer(Policy_P)$, Ownership sequence of $P_A$ $OS(P_A)$, time-
    stamp $T$, uploaded photo pool $P_{pool}$, and Photo Manage-
    ment Contract (PMC)
Ensure: Result $Result_{SNP_1}$ returned to $SNP_1$
 1:   if checkExistence($OS(P_A), P_{pool}$) then
 2:     $P_A$ has already been uploaded before;
 3:     goto Algorithm 3;
 4: else
 5:   $DMC^P \leftarrow PMC.createTree(O_A, SNP_1, Pointer(P_A), Pointer(Policy_P))$
 6:   $Opera_T \leftarrow$ upload
 7:   $Id \leftarrow DMC^P.createAccessRecord(O_A, SNP_1, Opera_T)$
 8:     $Pid \leftarrow DMC^P.\ createTreeRoot(Pointer(P_A),\ Pointer(Policy_P))$
 9:   $DMC^P.setPid(Id, Pid)$
10:   $P \leftarrow \{OS(P_A), O_A, SNP_1, T, DMC^P\}$
11:   $P_{pool} \leftarrow P_{pool} \cup P$
12:   $updatePMCContract(PMC, P_{pool})$
13:   $updateDMCContract(DMC^P, Id)$
14:   $DMC^P.setStatus(Pid, Active)$
15:   $Result_{SNP_1} \leftarrow \{DMC^P.Status, PMC_{id}, T\}$
16: end if
```

4.5.2 Ownership Legitimacy Detection

Ownership legitimacy detection determines whether the current user legally owns the photo. It returns False unless the photo exists in the photo pool, the user has legal ownership of the photo, and the corresponding dissemination node is currently active. Specifically, when the photo visitor V_B initiates access to P_A through the dissemination node O_A on SNP_1, the existence of P_A will be checked by Algorithm 3 to avoid forging access requests from dishonest SNPs. Algorithm 3 further examines the records of the dissemination node $SNP_1.O_A$ in the dissemination tree to ensure the legitimacy and the activity of the corresponding dissemination node.

Algorithm 3. Ownership Legitimacy Detection

```
Require: Photo owner $O_A$, Ownership Sequence $OS(P_A)$ of $P_A$,
    uploaded photo pool $P_{pool}$, and Photo Management Con-
    tract (PMC)
Ensure: Ownership Legitimacy of $O_A$ to $P_A$
 1: if checkNotExistence($OS(P_A), P_{pool}$) then
 2:   There is no photo $\in P_{pool}$ matched $P_A$;
 3:   return False;
 4: else
 5:   $DMC^P \leftarrow getTree(OS(P_A))$
 6:    if $checkAccessRecord(O_A, SNP_1, DMC^P)$ is not success
    then
 7:     $SNP_1.O_A$ is an illegal Photo owner;
 8:     return False;
 9:   end if
10:   $Pid \leftarrow getPid(O_A, SNP_1, DMC^P)$
11:   if $Pid == 0$ then
12:     $SNP_1.O_A$ is not an legal dissemination tree node;
13:     return False;
14:   end if
15:   if $getStatus(Pid)! = active$ then
16:     The dissemination state of $SNP_1.O_A$ is not active;
17:     return False;
18:   end if
19: end if
20: return $Pid, DMC^P$
```

4.5.3 Photo Access

Following the Legitimacy detection, DMC then gets P_A's policy by $Pointer'_P$ to determine whether V_B is allowed to visit/download P_A, and record the rusult on the blockchain. Since forwarding and re-uploading will change the photo's dissemination domain and the number of photo owners, they need more operations than visiting and downloading. If DMC comfirms the forwarding/re-uploading access, V_B will be added to the dissemination Tree in P_A's DMC as a new tree node. A new pair of $(Pointer_{P'}, Pointer_{Policy'})$ points to the storage address in SNP_2 will be recorded on the blockchain by DMC. The comprehensive process of photo access is illustrated in Algorithm 4.

4.5.4 Photo Deleting

After the uploading, photo owners can stop showing their shared photos on SNPs by submitting a photo deletion request. The request will remove the photo stored in all

SNPs by secondary owners and change the status of entire dissemination tree nodes behind them into inactive. In other words, all active dissemination child nodes in this dissemination path will be forcibly killed. If the image owner O_A from SNP_1 submits the deletion request, PMC and DMC will check whether O_A is a legitimate active node before changing the active status of all the dissemination tree subnodes including itself to inactive. Lastly, DMC will send a command to delete the corresponding data to the subnodes' SNPs. The specific photo deletion operation is described in Algorithm 5.

Algorithm 4. Access Photo

```
Require: Photo visitor $V_B$, Social Network Platform $SNP_2$ for $V_B$, Operation type $Opera_T$, Photo owner $O_A$, Social Network Platform $SNP_1$ for $O_A$, Ownership Sequence $OS(P_A)$, $V_B$'s privacy policy $Policy_P^{V_B}$, new pointer of $P_A$ $Pointer'_P$, pionter of $V_B$'s $Policy'_p$ $Pointer'_{Policy}$, Timestamp $T$, Uploaded photo pool $P_{pool}$, and Photo Management Contract (PMC)
Ensure: Result $Result_{SNP_2}$ returned to $SNP_2$
 1: if Algorithm 3$(OS(P_A), O_A, P_{pool})$!=True then
 2:     $O_A$ is not a legal dissemination tree node;
 3:     return False;
 4: else
 5:     $Pid, DMC^P \leftarrow$ Algorithm 3$(OS(P_A), O_A, P_{pool})$
 6: end if
 7: $Pointer(Policy_P), Pointer(P) \leftarrow getPolicyPointer(Pid, DMC^P)$;
 8: $Policy_P \leftarrow getPolicy(Pointer(Policy_P))$;
 9: $result \leftarrow checkPolicy(Policy_P, Opera_T, V_B, SNP_2)$;
10: $Id \leftarrow DMC^P.createAccessRecord(V_B, SNP_2, Opera_T, result)$;
11: if $result$ == false then
12:     $SNP_2.V_B$ has no right to operate $Opera_T$;
13:     $updateDMCContract(DMC^P, Id)$
14:     return False;
15: end if
16: if $Opera_T$ == Forward or Reupload then
17:     $O_{root} \leftarrow$ getTreeRoot $DMC^P$
18:     $Policy'_P \leftarrow$Algorithm 1$(O_A, O_root, V_B, Policy_P^{V_B})$;
19:     $Pid_B \leftarrow DMC^P.addTreeNode(Pointer'_P, Pointer'_{Policy})$
20:     $DMC^P.setStatus(Pid_B, Active)$
21:     $DMC^P.setFrontId(Pid_B, Pid_A)$
22:     $updateDMCContract(DMC^P, Id)$
23:     return$Result_{SNP_2} \leftarrow \{V_B, result, Opera_T, T\}$
24: end if
```

4.6 Privacy Protection in Our Framework

To protect owners' privacy, in our framework, the photo access and dissemination from any dissemination node must comply with that node's privacy policies, and both O_P^o and O_P^s have the right to control the sharing of their photos. Still, their control domains have differences: O_P^o's privacy policies cover all dissemination nodes, while O_P^s can only cover subsequent nodes with him as the root node. When a new node is generated, the DMC calculates the intersection of the requested privacy policies with the previous node as the actual policies of the new node. It is not hard to see that the actual privacy policy $Policy'_P$ at each node is the intersection of all the parent nodes' user privacy policies, which can be expressed as Equation (7). If the intersection is empty, which means that this request violates the previous owners' privacy, then this new node will be inactive with no valid accessible objects.

$$\mathrm{Policy}'_P = \bigcap_{i=1}^{n} \mathrm{policy}_P^i (\mathrm{n} = |\,\mathrm{FrontOwners}\,|). \tag{7}$$

In general, subsequent O_P^s can only narrow the scope of the photos' distractibility by setting their own privacy policy, but cannot expand it, which is in line with the reality of privacy protection needs.

Algorithm 5. Delete Photo

```
Require: Photo owner $O_A$, Social Network Platform $SNP_1$, Ownership Sequence of $P_A$ $OS(P_A)$, timestamp $T$, Uploaded photo pool $P_{pool}$, and Photo Management Contract (PMC)
Ensure: Set of command $Comand_{SNP_n}$ which will be sent to subnodes' $SNP_n$
 1: if Algorithm 3$(OS(P_A), O_A, P_{pool})$!=True then
 2:     $O_A$ is not a legal dissemination tree node;
 3:     return False;
 4: else
 5:     $Pid, DMC^P \leftarrow$ Algorithm 3$(OS(P_A), O_A, P_{pool})$
 6: end if
 7: $PidSet \leftarrow getAllSubNode(Pid)$
 8: for $Pid_n$ in $PidSet$ do
 9:     $DMC^P.setStatus(Pid_n, Inactive)$
10:     $(O_n, SNP_n, Pointer_{P_n}, Pointer_{Policy_n}) \leftarrow SMC^P_{addr}.getOwner(Pid_n)$
11:     $Command_{SNP_n} \leftarrow (O_n, Pointer_{P_n}, Pointer_{Policy_n}, delete, T)$
12: end for
13: $Id \leftarrow DMC^P.addAccessRecord(O_A, SNP_1, Delete, adopt)$
14: $updateDMCContract(DMC^P, Id)$
15: return $Command_{SNP_n}$
```

4.7 Security Analysis

To explain the security of the framework, we analyze the performance against the threat models previously mentioned in Section 3.2 under the security assumptions described in Section 3.3.

4.7.1 Resistance of Malicious Owner

In our framework, users of SNPs are not entitled to access control decisions, and malicious owner cannot bypass smart contracts to perform illegal dissemination operations on shared photos directly. Because of the legitimacy check before each access, even if the attackers successfully bypass and forward/re-upload the photos, they cannot create a corresponding dissemination record and forge a new dissemination node in all distributed peers' ledgers in the blockchain, and will be detected and traced immediately once they have access traffic from the networks.

4.7.2 Resistance of Malicious Visitor

Unlike existing privacy protection mechanisms for online photo sharing, in our framework, the photo owners' privacy policies remain valid for any access or dissemination, no matter in any platform. Thus malicious visitors cannot

TABLE 2
Survey Result of Friend Classification and Privacy Intention

Groups	Set Rate	Less than 10	10-50	50 to 100	More than 100	Can't view	Viewable	Downloadable	Forwardable
Close friends	44%	29.8%	12%	1.4%	0.8%	5.6%	30.7%	15.2%	15.2%
Classmates	71.9%	3.9%	29.2%	18.4%	20.3%	4.4%	60%	20.4%	20.7%
Colleague	47.6%	12%	24.8%	8.4%	2.5%	14.4%	31.1%	7.4%	8.1%
Teachers	56%	29%	25.3%	1.4%	0.3%	33.3%	22.6%	5.6%	5.9%
Superior	21.4%	13.4%	7.2%	0.6%	0.3%	15.9%	5.9%	0.4%	0.4%
Elders	35.9%	18.4%	16.4%	1.1%	0%	23%	31.5%	8.1%	10.4%
Relative	57.9%	16.7%	38.4%	2.5%	0.3%	23%	31.5%	8.1%	10.4%
Spouse	14.8%	13.4%	0.6%	0.3%	0.6%	1.1%	8.5%	5.9%	8.9%
Strangers	29.8%	9.7%	11.7%	5%	3.3%	25.6%	7.8%	1.9%	2.2%
Others	15.9%	4.5%	6.1%	3.3%	1.9%	8.9%	5.2%	0.7%	1.1%

access any sensitive photos of their targets via online social network platforms without the owner's permission.

4.7.3 Resistance of Malicious Social Networking Platforms

As previously mentioned, malicious SNPs may attempt to gain other platforms' user photos without authorization. However, due to the decentralized architecture, every platform maintains a consistent ledger in its local peer, which can easily confirm the access result, making it impossible to obtain photo data without the confirmation of smart contracts. Besides, according to the Byzantine consensus, under the premise of a total of N, even if the number of dishonest or non-working social network servers reaches $k(3k+1 \leq N)$, the blockchain server can still guarantee normal operation and provide normal response and service for requests.

5 Performance Evaluations

The main goal of Go-Sharing is to achieve a secure extended control system for photo sharing across social networks. We develop and implement a prototype system based on Hyperledger Fabric 2.0 to test our proposed framework, and simulate the social network platform as well as mobile app client on Github.[3]

5.1 Dataset

5.1.1 User Dataset for Photo Sharing

Go-sharing runs on the top of several near-real SNPs. To recreate real-life social networking scenarios, we conducted a survey[4,5] to investigate friend classification and privacy preference desires of real-world Internet users. 533 responses were received, and results are summarized in Table 2. Based on this, we calculate the approximate friend denial rate, visitable rate, and forwardable rate for each user, and depending on which, we divide visiting test users and forwarding test users into three groups.

For visiting, as shown in Fig. 7a, about 48.85% of users do not set up any friend groups and have no limit for their share of photos, we classify them as tolerant users. Normal users set up groups to restrict access to shared photos, but their denial rate is below 25%. Strict users, who account for 26.04%, set up friend groups and refuse more than 25% of their friends to visit their photos. As shown in Fig. 7b, for forwarding it appears that users are generally less tolerant than visiting users. According to feedback, the ratio of strict users increased to 47.4% among the forwarding users. This means that nearly half of users set up friend groups and refuse more than 25% of their friends from forwarding their photos. The normal group users account for 36%, while tolerant group users decreased from 48.8% to 16.6%, which means that more than 32% of visiting tolerant users turned into normal or strict users and no longer set no limit to their online friends in forwarding cases.

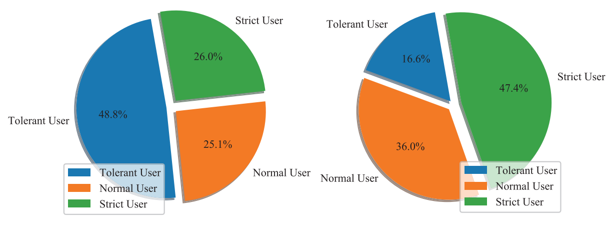
5.2 Evaluations of Go-Sharing

5.2.1 Performance Analysis

We run Go-sharing on a blockchain network consisting of five servers. As shown in Table 3, each server is equipped with a 128 G RAM and an Intel Core Xeon E5-2630 v4. To test the user experience of the Go-sharing, we analyze the temporal effects of uploading, visiting, forwarding and deleting access operations.

Photo Uploading. We upload 1,000 photos binding with random policies on Go-sharing and record their time cost. As a result, the entire uploading takes about 195.18 ms on average. Specifically, it takes 20.332 ms for ownership detection, 0.121 ms for ownership insertion, and 132.67 ms for smart contracts execution and blockchain updating, which actually cost most of the uploading.

Photo Visiting. According to Section 5.1.1, we divide visiting users into strict users, normal users, and tolerant users and test their time cost of visiting separately. On average, strict users take an average of 133.27 ms to access their photos. This rises to 133.53 ms for normal users and takes about 132.53 ms on average for tolerant users. The result shows that different privacy policy choices do not significantly affect the time overhead of photo visiting.



(a) Group of visiting (b) Group of forwarding

Fig. 7. Test group of visiting and forwarding.

3. https://github.com/Mirecle/Go-Shairng
4. https://wj.qq.com/s2/7948438/bb6b/
5. https://forms.gle/wU55fVxywfcCY1c57

TABLE 3
Testing Equipment

	SNP 1	SNP 2	SNP 3	SNP 4	SNP 5
IP	192.168. 1.101	192.168.1. 1.102	192.168.1. 1.103	192.168. 1.104	192.168. 1.105
CPU	Intel(R) Xeon(R) E5-2630 v4	Intel(R) Xeon(R) E5-2630 v4	Intel(R) Xeon(R) E5-2630 v4	Intel(R) Xeon(R) E5-2630 v4	Intel(R) Xeon(R) E5-2630 v4
Memory	128 GB	128 GB	128 GB	128GB	128GB
Name of Peers	Org1.Peer1 Org1.Peer2 Org1.Peer3 Org1.Peer4	Org2.Peer1 Org2.Peer2 Org2.Peer3 Org2.Peer4	Org3.Peer1 Org3.Peer2 Org3.Peer3 Org3.Peer4	Org4.Peer1 Org4.Peer2 Org4.Peer3 Org4.Peer4	Org5.Peer1 Org5.Peer2 Org5.Peer3 Org5.Peer4

Photo Forwarding. Depending on policy preferences, we divide the forwarding users into three groups. As shown in Fig. 8, photo forwarding takes a longer time than photo access due to the need for legitimacy testing, and it takes about 134.99 ms, 135.08 ms, and 135.22 ms on average for test groups respectively. Similarly, the time spent on photo forwarding is also not affected by different privacy preferences.

Photo Deleting. We delete 3,000 photos on Go-sharing and measure the costs of different forwarding groups. As shown in Fig. 8, it takes 131.68 ms for strict users and 135.74 ms for normal users. As the average number of forwarding increases, the deletion cost for tolerant users rises to 138.76 ms. This is because the deletion mechanism is responsible for inactivating all subnodes and pushing their SNPs to delete photos.

5.2.2 *Peak Flow Analysis*

To demonstrate the performance of Go-Sharing, we test the extreme performance of photo uploading, visiting, forwarding, and deleting under ideal conditions.

To achieve higher throughput and lower latency, it is better to set no limit for shared photos. Fig. 9 plots the average throughout, average latency and max latency for various access operations. As we can see, with the increase in transaction arrival rate, every operation's throughput increases linearly as expected till it reaches the saturation point at around 1802 tps (transaction per second), 1740 tps, 470 tps, and 433 tps. The average latency does not fluctuate with the growth of aps (arrival per second) and stays within 1 s stably. In contrast, the max latency is sensitive with the aps and reaches its peak early in extreme performance test.

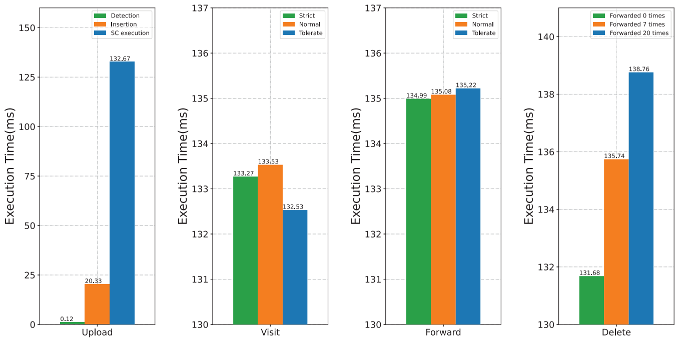


Fig. 8. Time cost for different operations.

5.3 Evaluations of Ownership Identification Scheme

According to the results in Section 4.2, faced with potential noise attacks, the validity of Go-sharing highly depends on the accuracy of ownership identification. Thus in this section, we evaluate the performance of the ownership identification scheme in Go-Sharing.

To implement the proposed ownership identification scheme, We use 100,000 and 1,000 images from the COCO data set [56] for model training and testing, respectively. Further, to better analyze the performance, we compared our work with the state-of-the-art [52] in the experiment with another 1,000 randomly chosen COCO photos. Unlike other works that train multiple decoders for different noises under several fixed intensities, we only train one generic decoder under unknown noise scenarios without any prior knowledge by using a random noise black box.

5.3.1 *Photo Quality of Ownership Encoding*

In Go-sharing, every photo will be encoded with its ownership sequence to help identification during the sharing. To keep the social value of the sharing photos from being destroyed, such encoding should avoid affecting the photo quality as much as possible. Hence, in this section, we demonstrate the performance of the ownership encoding in our model by using the metrics of peak-signal-to-noise ratio (PSNR) to evaluate the photos' quality from encoded images, where PSNR is the mean square error logarithm between the original image and the processed image relative to $(2^n - 1)^2$. Generally, a higher PSNR value depicts a higher image quality.

Table 4 shows the comparison of the PSNR of our models and several specified encoders in HiDDeN that are trained for different noise scenarios: No noise, Crop ($p = 20\%$), and Resize ($p = 70\%$), in which we can see that our scheme has significant advantages over various HiDDeN encoders in terms of image quality. In details, HiDDeN encoders for

TABLE 4
Spnr Under Different Encoders

encoder	PSNR(R)	PSNR(G)	PSNR(B)	PSNR(average)
Ours	**46.39**	**46.25**	**45.70**	**45.99**
HiDDeN(No noise)	39.02	38.68	39.58	39.00
HiDDeN(Crop)	35.66	36.54	35.32	35.68
HiDDeN(Resize)	36.50	35.35	35.42	35.59

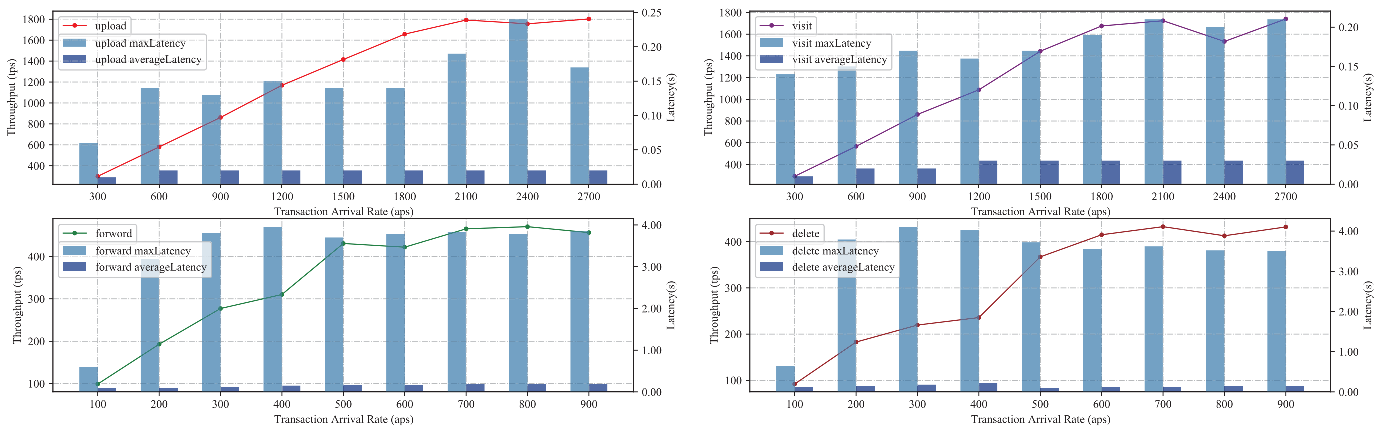


Fig. 9. Extreme performance of throughout & latency.

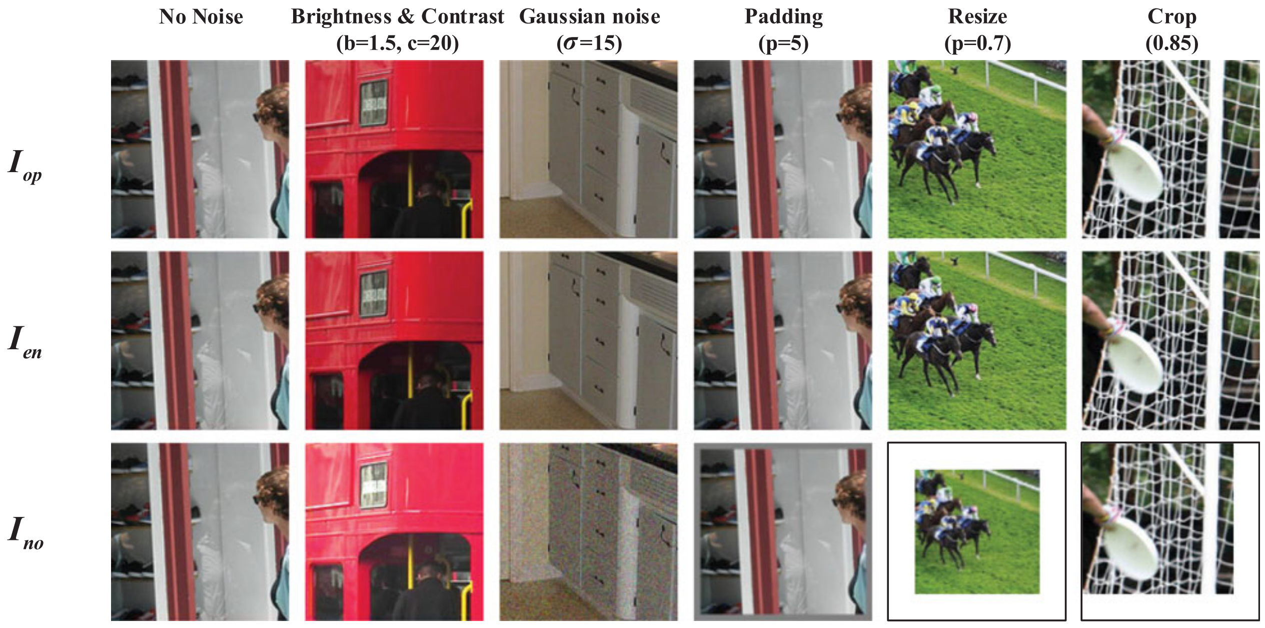
different noise types have different performances in terms of image quality. Their PNSR fluctuates between 35 and 36 and reaches 39 for no noise encoder. While the PSNR of our scheme can get 45.99 in average, compared to the best performance of HiDDeN improved by nearly 25.1% (17.9% to No noise, 28.90% to Crop and 29.22% to Resize). This is because HiDDeN uses one-stage end-to-end training (OET), in which both encoder and decoder are training together. So in order to improve the robustness to maintain high accuracy, the encoder is required to continuously strengthens the existence of watermark, resulting in an inevitable degradation of the quality of the encoder image. On the contrary, since we adopt TSDL, which places the training of encoder and decoder in two independent stages, the second stage of decoder training does not affect the performance of the previous stage of the encoder, regardless of the noise level.

5.3.2 The Robustness of Ownership Identification

The above evaluation demonstrates the advantages of our scheme in remaining photo quality. Besides, Go-Sharing also shows the robustness of ownership identification in cross-SNP photo sharing.

Since the noise present in the actual image propagation is unpredictable and diverse, without the support of the original image to clarify the type of noise, the ownership identification in Go-Sharing is trained robust enough to handle all possible noise situations with a single generic model. To better demonstrate the robustness of our scheme, in this section, we compare our model with three selected models (no noise, Resize, and Crop) from HiDDeN to test the *bit accuracy* of their decoders under random types, amounts as well as combinations of noises.

Single Noise Attack. Fig. 10 shows the test results of the bit accuracy under single typical noise examples, which includes:



	No Noise	Brightness & Contrast (b=1.5, c=20)	Gaussian noise (σ=15)	Padding (p=5)	Resize (p=0.7)	Crop (0.85)
Ours	99.95%	**99.37%**	**92.20%**	**99.85%**	**99.54%**	81.42%
Hidden (No Noise)	**99.96%**	94.28%	68.89%	89.44%	59.80%	98.96%
Hidden (Resize)	89.75%	82.16%	62.86%	76.99%	99.31%	80.48%
Hidden (Crop)	99.94%	95.13%	69.68%	98.61%	63.23%	**99.94%**

Fig. 10. Robustness against single noises.

TABLE 5
Robustness Against Multi-Noise Attack

Noise Type	HiDDeN (No Noise)	HiDDeN (Resize)	HiDDeN (Crop)	Ours
B&C, Padding	80.56%	71.96%	88.47%	**99.73%**
B&C, Gaussian noise	68.81%	62.49%	69.11%	**89.82%**
Crop, Resize	57.16%	78.89%	62.09%	**81.95%**
Gaussian noise, Resize	52.51%	72.18%	51.82%	**88.82%**
Padding, Resize	53.74%	76.93%	56.91%	**98.76%**
Gaussian noise, Padding	62.20%	58.70%	63.53%	**90.66%**
B&C, Resize, Padding	53.18%	71.25%	54.56%	**98.82%**
B&C, Gaussian noise, Padding	62.37%	59.38%	61.49%	**88.26%**
B&C, Crop, Padding	55.35%	72.49%	57.72%	**79.98%**
Gaussian noise, Crop, Resize	51.73%	58.60%	52.24%	**71.29%**
Gaussian noise, Padding, Resize	50.55%	60.71%	51.33%	**87.29%**
Padding, Crop, Resize	54.61%	76.45%	55.94%	**83.65%**
B&C, Crop, Resize	55.44%	72.49%	57.72%	**80.02%**

Brightness & Contrast ($b = 1.5, c = 20$), Gaussian noise ($\sigma = 15$), Padding ($p = 5$), Resize ($p = 0.7$) and Crop ($p = 0.85$). We compared our scheme with three specialized HiDDeN models and marked the top accuracy value with red color.

From the results we can see that although we have only one generic model, our model performs the best among the four methods for the noise of Brightness & Contrast, Gaussian noise, Padding, and Resize. Accuracy improves by up to 33.84% under Gaussian noise than the second one. While for No Noise and Crop, HiDDeN (no noise) and HiDDeN (Crop) occupy the top accuracy with 99.96% and 99.94%, respectively. Unfortunately, due to deterministic noise training, even though these models achieve highest accuracy in their characteristic noisy scenes, they perform poorly against other kinds of noise (e.t. 59.80% accuracy in Resize for HiDDeN (No Noise) and 69.68% accuracy in Gaussian noise for HiDDeN (Crop)). It turns out that when the bit accuracy is lower than 70%, the ownership sequence decoded will not be able to match the address of its corresponding dissemination tree root through the error correction code. Overall, the average accuracy of our model reaches 95.39% in the face of the above single noise, which is much higher than the other three models whose average are at about 85.22%, 81.92%, and 87.76%, respectively.

Multi-Noise Attack. Since the dissemination of photos involves numerous forwarding users, all of whom may make certain changes to the images, photo sharing in Go-Sharing is also likely to face attacks from compound noise. So in addition to the single noise attack, we also evaluate the robustness of our scheme in the multi-noise attacks.

Table 5 lists 13 combinations, including 2 to 3 different noises, and shows their corresponding bit accuracy in different decoders, in which we also mark the highest accuracy value in red color. The results shows that compared with the HiDDeN decoders trained with given specified noises, the accuracy of our generic model presents a more obvious advantage in the face of multiple noises. Once more than one noise is introduced, the accuracy of the HiDDeN decoders decreases significantly, specifically dropping from 85.22% (No noise), 81.92% (Resize), and 87.76% (Crop) to 54.98%, 68.65%, and 60.23%, far below the minimum standard of 70%. On the contrary, our model's accuracy consistently stays above 70% and achieves an average accuracy of 87.62%, which is 27.63% above the former's best performance.

6 Conclusion

In this article we introduced Go-Sharing a blockchain-based cross-social network privacy picture sharing framework for efficient extended control of photos in cross-SNP scenarios. Go-Sharing provides automatic image dissemination control through smart contracts, and as a result, Go-Sharing helps users regain control of their pictures and to fully use their legal rights. Through our random noise black box we also proposed a TSDL based anti-random noise photo ownership identification scheme that allows robust identification during cross-SNP sharing. Our evaluation results demonstrate the effectiveness of our framework.

7 Future Work

In Go-Sharing, the first uploader of the photo is treated as the photo owner, and subsequent forwarders are treated as co-owners. The aim of Go-Sharing is to protect the user's dissemination ownership in cross-SNP photo sharing. But photo ownership is a complex phenomenon with different manifestations, such as people, objects, and some symbols appearing in the image. Therefore, in our future work, we will consider improving the protection mechanism in Go-Sharing to provide more comprehensive ownership protection. What's more, as a decentralized technique to solve privacy conflicts in photo dissemination, Go-Sharing prevents the infringing uploader from sharing elsewhere. However, it does not support the decision-making of the uploader about sharing co-owned content. Therefore, in future work, we consider incorporating a type of dissuasive mechanism to give users a clearer understanding of the impact of sharing. Besides, Go-Sharing generates the optimal privacy policy for each dissemination. For future work we will consider improving the dynamic privacy policy generation algorithm to meet the demands of content-based multiuser privacy protection scenarios.

ACKNOWLEDGMENTS

This is an extended version of our conference article [1].

REFERENCES

[1] M. Zhang et al., “A blockchain-based privacy-preserving framework for cross-social network photo sharing,” in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2022, pp. 1–6.

[2] A.-M. Olteanu, “Interdependent and multi-subject privacy: Threats, analysis and protection,” p. 155, 2019. [Online]. Available: http://infoscience.epfl.ch/record/264794

[3] Y. Xu, T. Price, J.-M. Frahm, and F. Monrose, “Virtual U: Defeating face liveness detection by building virtual models from your public photos,” in *Proc. 25th USENIX Secur. Symp.*, 2016, pp. 497–512.

[4] Privacy concerns with social networking services, 2022. [Online]. Available: https://en.wikipedia.org/wiki/ Privacy_concerns_with_social_networking_services#Potential_dangers

[5] C. Bo, G. Shen, J. Liu, X.-Y. Li, Y. Zhang, and F. Zhao, “Privacy. tag: Privacy concern expressed and respected,” in *Proc. 12th ACM Conf. Embedded Netw. Sensor Syst.*, 2014, pp. 163–176.

[6] L. Zhang, X.-Y. Li, K. Liu, C. Liu, X. Ding, and Y. Liu, “Cloak of invisibility: Privacy-friendly photo capturing and sharing system,” *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2488–2501, Nov. 2019.

[7] P. Klemperer et al., “Tag, you can see it! using tags for access control in photo sharing,” in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2012, pp. 377–386.

[8] J. Pang and Y. Zhang, “A new access control scheme for facebook-style social networks,” *Comput. Secur.*, vol. 54, pp. 44–59, 2015.

[9] H. Hu, G.-J. Ahn, and J. Jorgensen, “Multiparty access control for online social networks: Model and mechanisms,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 7, pp. 1614–1627, Jul. 2013.

[10] E. Palomar, L. González-Manzano, A. Alcaide, and Á. Galán, “Implementing a privacy-enhanced attribute-based credential system for online social networks with co-ownership management,” *IET Inf. Secur.*, vol. 10, no. 2, pp. 60–68, 2016.

[11] K. Xu, Y. Guo, L. Guo, Y. Fang, and X. Li, “My privacy my decision: Control of photo sharing on online social networks,” *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 2, pp. 199–210, Mar./Apr. 2017.

[12] Z. Xia, Y. Zhu, X. Sun, Z. Qin, and K. Ren, “Towards privacy-preserving content-based image retrieval in cloud computing,” *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 276–286, Jan.-Mar. 2018.

[13] M.-R. Ra, R. Govindan, and A. Ortega, “P3: Toward privacy-preserving photo sharing,” in *Proc. 10th USENIX Symp. Netw. Syst. Des. Implementation*, 2013, pp. 515–528. [Online]. Available: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/ra

[14] H. Wang, C. Wang, N. Shu, and J. Huang, “A practical image encryption algorithm for privacy protection,” in *Proc. IEEE 19th Int. Conf. Commun. Technol.*, 2019, pp. 1611–1615.

[15] E. Yang, S. Fang, C. E. Grant, and L. Gruenwald, “Poster: PhotoLock: Autonomous privacy-preserving photo sharing in online social networks” 2020.

[16] L. Fan, “A demonstration of image obfuscation with provable privacy,” in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, 2019, pp. 608–608.

[17] G. Boracchi and A. Foi, “Modeling the performance of image restoration from motion blur,” *IEEE Trans. Image Process.*, vol. 21, no. 8, pp. 3502–3517, Aug. 2012.

[18] L. Xu, T. Bao, L. Zhu, and Y. Zhang, “Trust-based privacy-preserving photo sharing in online social networks,” *IEEE Trans. Multimedia*, vol. 21, no. 3, pp. 591–602, Mar. 2019.

[19] A. Rajabi, R. B. Bobba, M. Rosulek, C. V. Wright, and W. Feng, “On the (im) practicality of adversarial perturbation for image privacy,” *Proc. Privacy Enhancing Technol.*, vol. 1, pp. 85–106, 2021.

[20] P. Regulation, “Regulation (eu) 2016/679 of the european parliament and of the council,” *Regulation (eu)*, vol. 679, p. 2016, 2016.

[21] N. Vishwamitra, Y. Li, K. Wang, H. Hu, K. Caine, and G.-J. Ahn, “Towards PII-based multiparty access control for photo sharing in online social networks,” in *Proc. 22nd ACM Symp. Access Control Models Technol.*, 2017, pp. 155–166.

[22] F. Li et al., “Cyberspace-oriented access control: A cyberspace characteristics-based model and its policies,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1471–1483, Apr. 2019.

[23] F. Li, Z. Sun, A. Li, B. Niu, H. Li, and G. Cao, “HideMe: Privacy-preserving photo sharing on social networks,” in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 154–162.

[24] W. Sun, J. Zhou, S. Zhu, and Y. Y. Tang, “Robust privacy-preserving image sharing over online social networks (OSNs),” *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 14, no. 1, pp. 1–22, 2018.

[25] X. Wang and S. Gao, “Image encryption algorithm based on the matrix semi-tensor product with a compound secret key produced by a boolean network,” *Inf. Sci.*, vol. 539, pp. 195–214, 2020.

[26] Y. Xian and X. Y. Wang, “Fractal sorting matrix and its application on chaotic image encryption,” *Inf. Sci.*, vol. 547, pp. 1154–1169, 2021.

[27] L. Jiang and X. Zhang, “BCOSN: A blockchain-based decentralized online social network,” *IEEE Trans. Computat. Social Syst.*, vol. 6, no. 6, pp. 1454–1466, Dec. 2019.

[28] M. Li et al., “CrowdBC: A blockchain-based decentralized framework for crowdsourcing,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1251–1266, Jun. 2019.

[29] R. Yu et al., “Authentication with block-chain algorithm and text encryption protocol in calculation of social network,” *IEEE Access*, vol. 5, pp. 24944–24951, 2017.

[30] M. U. Rahman, B. Guidi, and F. Baiardi, “Blockchain-based access control management for decentralized online social networks,” *J. Parallel Distrib. Comput.*, vol. 144, pp. 41–54, 2020.

[31] M. Cherubini, K. Salehzadeh Niksirat, M.-O. Boldi, H. Keopraseuth, J. M. Such, and K. Huguenin, “When forcing collaboration is the most sensible choice: Desirability of precautionary and dissuasive mechanisms to manage multiparty privacy conflicts,” *Proc. ACM Hum.- Comput. Interaction*, vol. 5, no. CSCW1, pp. 1–36, 2021.

[32] K. S. Niksirat, E. Anthoine-Milhomme, S. Randin, K. Huguenin, and M. Cherubini, ““I thought you were okay”: Participatory design with young adults to fight multiparty privacy conflicts in online social networks,” in *Proc. Designing Interactive Syst. Conf.*, 2021, pp. 104–124.

[33] United nations portal. 1948. universal declaration of human rights, 2021. [Online]. Available: https://www.un.org/en/universal-declarationhuman-rights/

[34] Secretariat for legal affairs. 1969. american convention on human rights, 2021. [Online]. Available: http://www.oas.org/dil/treaties_B-32_American_Convention_on_Human_Rights.htm

[35] B. Carminati and E. Ferrari, “Collaborative access control in online social networks,” in *Proc. 7th Int. Conf. Collaborative Comput.: Netw.*, Appl. Worksharing, 2011, pp. 231–240.

[36] C. Phelan, C. Lampe, and P. Resnick, “It’s creepy, but it doesn’t bother me,” in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2016, pp. 5240–5251.

[37] K. Thomas, C. Grier, and D. M. Nicol, “Unfriendly: Multi-party privacy risks in social networks,” in *Proc. Int. Symp. Privacy Enhancing Technol. Symp.*, 2010, pp. 236–252.

[38] S. Rajtmajer, A. Squicciarini, J. M. Such, J. Semonsen, and A. Belmonte, “An ultimatum game model for the evolution of privacy in jointly managed content,” in *Proc. Int. Conf. Decis. Game Theory Secur.*, 2017, pp. 112–130.

[39] J. M. Such and M. Rovatsos, “Privacy policy negotiation in social media,” *ACM Trans. Auton. Adaptive Syst.*, vol. 11, no. 1, pp. 1–29, 2016.

[40] F. Mosca, J. M. Such, and P. J. McBurney, “Value-driven collaborative privacy decision making,” in *Proc. AAAI Spring Symp. PAL: Privacy-Enhancing Artif. Intell. Lang. Technol.*, 2018, pp. 13–20.

[41] F. Mosca and J. Such, “ELVIRA: An explainable agent for value and utility-driven multiuser privacy,” in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2021, pp. 916–924.

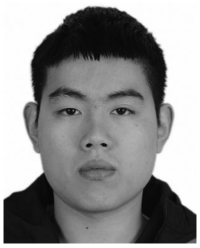
[42] F. Mosca and J. Such, “An explainable assistant for multiuser privacy,” *Auton. Agents Multi-Agent Syst.*, vol. 36, no. 1, pp. 1–45, 2022.

[43] J. M. Such, J. Porter, S. Preibusch, and A. Joinson, “Photo privacy conflicts in social media: A large-scale empirical study,” in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2017, pp. 3821–3832.

[44] L. Du, A. T. Ho, and R. Cong, “Perceptual hashing for image authentication: A survey,” *Signal Process.: Image Commun.*, vol. 81, 2020, Art. no. 115713.

[45] S.-H. Han and C.-H. Chu, “Content-based image authentication: Current status, issues, and challenges,” *Int. J. Inf. Secur.*, vol. 9, no. 1, pp. 19–32, 2010.

[46] M. Boroumand, M. Chen, and J. Fridrich, “Deep residual network for steganalysis of digital images,” *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 5, pp. 1181–1193, May 2019.
[47] D. Shu, W. Cong, J. Chai, and C. S. Tucker, “Encrypted rich-data steganography using generative adversarial networks,” in *Proc. 2nd ACM Workshop Wireless Secur. Mach. Learn.*, 2020, pp. 55–60.
[48] Y. Liu, M. Guo, J. Zhang, Y. Zhu, and X. Xie, “A novel two-stage separable deep learning framework for practical blind watermarking,” in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 1509–1517.
[49] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, “A digital watermark,” in *Proc. 1st Int. Conf. Image Process.*, 1994, pp. 86–90.
[50] M.-S. Hsieh, D.-C. Tseng, and Y.-H. Huang, “Hiding digital watermarks using multiresolution wavelet transform,” *IEEE Trans. Ind. Electron.*, vol. 48, no. 5, pp. 875–882, Oct. 2001.
[51] V. M. Potdar, S. Han, and E. Chang, “A survey of digital image watermarking techniques,” in *Proc. 3rd IEEE Int. Conf. Ind. Informat.*, 2005, pp. 709–716.
[52] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, “HiDDeN: Hiding data with deep networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 657–672.
[53] J. M. Such and N. Criado, “Multiparty privacy in social media,” *Commun. ACM*, vol. 61, no. 8, pp. 74–81, 2018.
[54] J. M. Such and N. Criado, “Resolving multi-party privacy conflicts in social media,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1851–1863, Jul. 2016.
[55] M. Ahmadi, A. Norouzi, N. Karimi, S. Samavi, and A. Emami, “ReDMark: Framework for residual diffusion watermarking based on deep networks,” *Expert Syst. Appl.*, vol. 146, 2020, Art. no. 113157.
[56] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.



Ming Zhang (Student Member, IEEE) received the BTech degree in computer science and engineering from the School of Cyber Engineering, Xidian University, in 2019, and now is working toward the PhD degree with the School of Cyber Engineering, Xidian University, Xi’an, China. His current research interests focus on social network image privacy protection, blockchain technology and applications. he has a publication in conferences such as INFOCOM workshop MobiSec.

Zhe Sun (Member, IEEE) received the BS degree from the Dalian University of Technology, in 2010, the MS degree from the University of Science and Technology of China, in 2012, and the PhD degree from the University of Chinese Academy of Sciences, in 2019. He is currently an associate professor with the Cyberspace Institute of Advanced Technology, Guangzhou University, China. He has authored more than 20 publications published by refereed international conferences and journals. His current research interests include privacy-preserving mechanism in multiparty data sharing and deep learning. His research projects are funded by the National Natural Science Foundation of China and the China Postdoctoral Science Foundation.

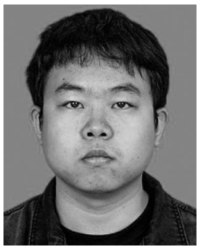
Hui Li (Member, IEEE) received the BSc degree from Fudan University, in 1990, and the MASc and PhD degrees from Xidian University, in 1993 and 1998, respectively. Since June 2005, he has been the professor with the School of Cyber Engineering, Xidian University, Xi’an, Shaanxi, China. His research interests are in the areas of cryptography, wireless network security, information theory and network coding. He is a co-author of two books. He served as technique committee co-chairs of ISPEC 2009 and IAS 2009.

Ben Niu (Member, IEEE) received the BS degree in information security, and the MS and PhD degrees in cryptography from Xidian University, in 2006, 2010 and 2014, respectively. Currently, he is working as associate professor with the Institute of Information Engineering, Chinese Academy of Sciences. He was a visiting scholar with Pennsylvania State University from 2011 to 2013. His current research interests include wireless network security, privacy computing.

Fenghua Li (Member, IEEE) received the BS degree in computer software, the MS and PhD degrees in computer systems architecture from Xidian University, in 1987, 1990 and 2009, respectively. Currently, he is working as professor and doctoral supervisor in Institute of Information Engineering, Chinese Academy of Sciences. He is also a doctoral supervisor with Xidian University, and University of Science and Technology of China. His current research interests include network security, system security, privacy computing and trusted computing.

Zixu Zhang (Student Member, IEEE) received the master of information technology degree from the University of Technology Sydney, in 2021. His main research focus on graph theory, blockchain, cyber security, federation learning.

Yuhang Xie (Student Member, IEEE) received the BTech degree in information security from Xidian University, Xi‘an, Shannxi, China, in 2019. He is currently working toward the master’s degree with Xidian University. His main research interests include blockchain and cyberspace security. He holds two patents for inventions in blockchain and image processing.



Chunhao Zheng (Student Member, IEEE) received the bachelor’s degree in network engineering from Southwest Jiaotong University, Chengdu, Sichuan, China, in 2020. His main research interests include content blockchain, privacy protection, network security.