electronics

MDPI

Article

Towards Building a Faster and Incentive Enabled Privacy-Preserving Proof of Location Scheme from GTOTP

Cong Ma, Yuhan Liu, Zheng Yang and Juan Ma *

College of Computer and Information Science College of Software, Southwest University, Chongqing 400010, China; swujkmc@email.swu.edu.cn (C.M.); lucky613@email.swu.edu.cn (Y.L.); youngzheng@swu.edu.cn (Z.Y.)
* Correspondence: mj321000311@email.swu.edu.cn

Abstract: In recent years, there has been significant growth in location-based services (LBSs) and applications. These services empower users to transmit their location data to location service providers, thereby facilitating the provisioning of pertinent resources and services. However, in order to prevent malicious users from sending fake location data, users must attest to their location for service providers, namely, through a proof of location (PoL). Such a proof should additionally prevent attackers from being able to obtain users' identity and location information through it. In this paper, we propose an efficient privacy-preserving proof of location (PPPoL) scheme. The scheme is based on the standard cryptographic primitives, including Group Time-based One-Time Password (GTOTP) and public key encryption, which achieves entity privacy, location privacy, and traceability. Unlike the previous GTOTP-based PPPoL scheme, our scheme enables instant location verification with additional hash operations. To encourage the active participation of witnesses in location proofs, we propose an incentive mechanism based on smart contracts. Additionally, we implement a proof of concept of our PPPoL scheme on an Android device. Our experimental results show that proof generation and verification time are on the order of milliseconds. Meanwhile, the total overhead for the incentive mechanism amounts to 0.0011 ETH. This result is practical for mobile device-based LBSs.

Keywords: proof of location; location privacy; anonymity; blockchain; smart contract

check for updates

Citation: Ma, C.; Liu, Y.; Yang, Z.; Ma, J. Towards Building a Faster and Incentive Enabled Privacy-Preserving Proof of Location Scheme from GTOTP. *Electronics* **2024**, *13*, 1443. https://doi.org/10.3390/electronics13081443

Academic Editor: Cheonshik Kim

Received: 25 February 2024
Revised: 9 April 2024
Accepted: 9 April 2024
Published: 11 April 2024

1. Introduction

Location-based services (LBSs) have gained significant prominence in recent years [1] due to the widespread adoption of mobile devices [2] and advancements in location-based technologies. In LBSs, users leverage location-based technologies to share their locations with location service providers (SPs) in order to access location-dependent services and resources such as location-based social networks (LBSN) [3], location-based assistive technology [4], and fitness monitoring. LBSN enables users to share their locations and interact with friends, family, or like-minded individuals while providing personalized venue recommendations based on their preferences and locations. Additionally, location-based assistive technology can aid users with visual or cognitive impairments in perceiving their surroundings, thereby enhancing their quality of life. Furthermore, combining LBSs with motion sensors can assist users in tracking their movement paths, distances, and speeds, which is highly beneficial for fitness enthusiasts and rehabilitation patients who need to monitor their physical activity capabilities. Nevertheless, a challenge arises in instances where LBSs offer incentives or rewards to users who are physically present at specific times and locations, potentially creating an environment where users are motivated to manipulate their reported geographical coordinates. To ensure the effectiveness of such LBSs, it becomes imperative to ascertain the authenticity of users' geographical claims. This verification is accomplished through the provision of a valid location proof, which substantiates that a user was indeed situated at a particular location during a specific time frame. For instance, certain businesses may issue coupons exclusively to users in proximity, which necessitates the presentation of a valid proof of location (PoL) to mitigate access by those who do not meet the criteria. In addition, PoL schemes play an important role in other areas, such as supply chain management [5], vehicular ad hoc networks (VANETs) [6], and the Internet of Things (IoT) [7]. Notably, PoL schemes play a vital role in infectious contact tracing scenarios. Users engage in regular PoL schemes, collecting location proofs over time; in the event of an infection, health authorities, such as the Centers for Disease Control (CDC), can leverage a person's historical location proofs for swift contact tracing, and can adopt infection mitigation methods such as publishing the movement paths of infected persons in order to rapidly curb the spread of infectious diseases [8].

A PoL system facilitates the generation of tamper-proof location proofs by a participant referred to as the prover with the assistance of nearby counterparts termed witnesses. A verifier then utilizes the location proofs to confirm the validity of the location stated by the prover. However, PoL systems may encounter challenges pertaining to identity and location privacy stemming from the interactions between the prover and proximate witnesses. Throughout the location proof generation process, both the prover and witnesses may seek to conceal their identities and locations from other participants. Because of the potential sensitivity of location data, even minor location information can offer attackers the ability to infer additional private details [9], such as user habits and residential addresses. In addition, for resource-limited devices such as smartphones, PoL systems should not involve excessive computational and communication overhead.

The PoL system should prevent impersonation of honest users during the location proof generation process; thus, it is imperative for both the prover and witnesses to accomplish anonymous identity authentication. Group Time-based One-Time Password (GTOTP) [10] emerges as an efficient group-based authentication scheme in which each group member can assert their membership while concealing their true identity. Thanks to the anonymity and efficiency offered by GTOTP, it is exceptionally well suited for the construction of efficient location proof schemes. Yang et al. [10] proposed a PoL scheme based on GTOTP; however, their scheme does not take into consideration instant location proof verification, making it unsuitable for certain application scenarios that demand low verification latency. For instance, in applications such as bike sharing and car sharing, instant location verification is crucial to ensure that users are within the permitted geographical area when using shared resources [11]. In addition, from the perspective of the witnesses who contribute to the location proof, the witnesses have no incentive to participate in the location proof after receiving the location request from the prover, and may often refuse to participate in the location proof, as it may consume their own resources or reveal their private information when participating in the location proof.

Our work. To address the aforementioned issues, we propose an efficient privacy-preserving proof of location (PPPoL) scheme based on GTOTP that supports identity and location privacy for both the prover and witnesses while providing instant location verification. We emphasize that we mainly consider the location privacy of witnesses with respect to other participants, such as the prover and other witnesses. In applications such as contact tracing, the verifier is trusted; thus, a witness can provide their location information to the verifier. Furthermore, we design an incentive mechanism based on smart contracts to enhance the participation of witnesses in our PPPoL scheme. We use GTOTP to protect participants' identity privacy and the traceability of location proofs. To achieve location privacy, we employ a public key encryption algorithm that can resist chosen plaintext attacks to encrypt the location data of both the prover and witnesses. The prover creates a smart contract based on our incentive mechanism, uploads it to the blockchain, and then broadcasts a location proof request, including the contract address, to nearby witnesses. Upon receiving the location proof request, nearby witnesses have the option of examining the smart contract. If a witness decides to participate in the location proof, they generate a location proof piece and send it back to the prover. The prover integrates all the location proof pieces into a location proof, which is subsequently forwarded to the verifier. The verifier employs auxiliary verification information from both the prover and the relevant witnesses to verify the validity of the location proof. Following successful location verification, the rewards in the smart contract are automatically transferred to the registration authority (RA), which then transfers the rewards to the relevant witnesses. We make the following contributions:

- We construct an efficient PPPoL scheme which is suitable for resource-constrained devices by leveraging GTOTP. Compared to the prior GTOTP-based PPPoL scheme, our proposal can support instant location proof verification.
- We design an incentive mechanism based on smart contracts to encourage witnesses to participate in generating location proofs for the location prover.
- We evaluate our PPPoL scheme on an Android device, while the incentive mechanism is benchmarked using the Goerli Ethereum. The experimental results demonstrate the practical feasibility of our schemes. Specifically, both the location proof generation and verification times are within the range of tens of milliseconds, while the total overhead for the incentive mechanism amounts to 0.0011 ETH.

The rest of the paper is organized as described below. Section 2 reviews related work. Section 3 describes the main cryptographic techniques used in this work. Section 4 briefly analyzes the PoL scheme proposed by Yang et al. [10]. Section 5 describes the concrete details of our PPPoL scheme. Section 6 shows the incentive mechanism. Section 7 shows the performance of our PPPoL scheme and incentive mechanism. Finally, Section 8 concludes the paper.

2. Related Work

Time-based One-Time Password. The Time-based One-Time Password (TOTP) approach is widely utilized in scenarios involving two-factor authentication, enabling a prover to generate a time-dependent TOTP password that remains valid only within a predefined time period. The TOTP scheme contains two types, symmetric and asymmetric; the symmetric TOTP scheme requires the prover and verifier to share the secret key, while the asymmetric one does not. The first asymmetric TOTP scheme was proposed by Lamport [12]. This scheme was based on a one-way function, with each password derived by computing the one-way function on the previous password. Currently, most of the asymmetric TOTP schemes use architectures similar to the Lamport scheme. For example, Kogen et al. [13] proposed a hash-based T/key scheme by combining an S/key scheme [14] that instantiates a one-way function using a hash function with the Lamport scheme. However, in these schemes the verifier must first know the identity of the prover. To achieve privacy of the prover with regard to the verifier, Yang et al. [10] proposed the GTOTP scheme as an extension of asymmetric TOTP schemes.

Proof of Location. Many relevant PoL schemes have been proposed for protecting user privacy and improving the usability of mobile devices. PoL schemes running on mobile devices can be classified into two types, namely, infrastructure-dependent and infrastructure-independent.

Among infrastructure-dependent PoL schemes, Javali et al. [15] proposed a proof of location scheme that uses the unique characteristics of WiFi access points, i.e., channel state information (CSI) in combination with a fuzzy vault to generate location proofs for the presence of a user within an area of interest at a particular time. Li et al. [16] proposed a proof of location scheme based on WiFi or existing cellular network access points, which prevents location cheating and enables the database to verify the location without knowing the users' exact location.

Among infrastructure-independent PoL schemes, Zhu and Cao [17] proposed APPLAUS, which allows users to exchange location proofs and signed messages with each other via Bluetooth and to periodically change their pseudonyms. However, users' behavior can compromise the location privacy of the scheme, which may suffer a non-negligible loss of location privacy due to the use of a user-centric location privacy model. Wu et al. [18] proposed a proof of location scheme based on blockchain technology and zero-knowledge proofs. The user obtains a location certificate from multiple location beacons and generates a zero-knowledge proof-of-location certificate with the help of beacons. Yang et al. [10] proposed a proof of location scheme based on GTOTP, the privacy-preserving location proximity (PPLP) scheme from [19], and a commitment scheme. This scheme achieves both user identity and location privacy; however, because each witness requires an individually blinded proximity distance set that is sent to the verifier for proximity verification, the proof size grows as the number of witnesses increases. Moreover, the scheme does not support instant location verification. Notably, most existing PoL schemes do not take into account incentives for witness participation.

Smart Contracts. In the mid-1990s, Nick Szabo first coined the concept of a "smart contract" [20], defined as a computer program that automatically executes the terms of a contract when certain predefined conditions are met. Early design took place mainly around the premise of avoiding reliance on a trusted third party to act as a common trust agent between two parties to a contract, as the computer can automatically execute the contract in the actual transaction situation while efficiently and securely fulfilling the terms set by the smart contract [21]. Due to the limitations on computing scenarios in the early days, smart contracts were not widely applied. The blockchain technology used as the base of the decentralized peer-to-peer transaction cryptocurrency proposed by Satoshi Nakamoto has enabled the further development of smart contracts. Thanks to the decentralized and tamper-proof nature of blockchains [22], the programmability of smart contracts can be de-trusted and efficiently executed. In the early days, the Bitcoin network proposed by Satoshi Nakamoto [23] was mainly used to realize simple transaction execution through script language, which is not applicable to real life nowadays. Therefore, the combination of blockchain and smart contract technology not only realizes the programmability of a decentralized network but also further promotes the popularity of decentralized applications, which can effectively allow blockchain technology to promote the development of society.

3. Preliminaries

In this section, we briefly review the main cryptographic techniques used in our constructions. We denote the security parameter with κ. Let $[n] = \{1, \ldots, n\} \subset \mathbb{N}$ be the set of integers between 1 and n. We denote by $x \overset{\$}{\leftarrow} S$ the operation of sampling x uniformly and at random from the set S. We write L_C to denote the two-dimensional location coordinates $(x_\mathsf{C}, y_\mathsf{C})$ of user C. We denote the concatenation of two strings by $\|$. In this paper, unless otherwise specified, we assume that H(m) represents the hash operation applied to the message m.

3.1. Group Time-Based One-Time Passwords

GTOTP is a novel authentication scheme that enables a prover belonging to a group to prove its group membership with a verifier while hiding its identity; furthermore, the identity of group members can be traced if needed. GTOTP is based on asymmetric TOTP schemes, in which the prover generates a random secret seed and continuously hashes the value to a hash chain. Based on the current time, the prover sends the hash chain nodes as TOTP passwords to the verifier for verification in reverse order. The verifier performs continuous hash operations on the TOTP password and compares the obtained hash value with the tail node (also known as the verify point), where the count of hashes relates to the TOTP password generation time. If the two values are equal, then the TOTP password is valid. Due to the one-way nature of the hash function, the prover cannot forge a one-time password that has not been used. Table 1 describes the notation used by GTOTP.

A GTOTP scheme has $M \in \mathbb{N}$ members $\mathsf{GM} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_M)$, one registration authority (RA) that deals with protocol initialization and group membership registration, and one verifier. Figure 1 illustrates the authentication process of GTOTP, which begins with the users registering with the RA as group members. Subsequently, the group member generates a GTOTP password and sends it to the verifier for authentication. If necessary, such as when the group member provides an invalid GTOTP password or exhibits dishonest behavior, the verifier may initiate a request to the RA for member identity tracing.

Table 1. Some notation for GTOTP.

Notation	Description
M	Number of group members.
ID_u	Identify of group member ID_u.
$C^i{}_{\mathsf{ID}_u}$	Identity cipher for group member ID_u in the i-th verify epoch.
T_{start}, T_{end}	Start and end times of a protocol instance.
V, K	Numbers of TOTP instances and GTOTP passwords of a TOTP, respectively.
$vp^i{}_{\mathsf{ID}_u}$	The i-th verify point of group member ID_u.
$uvp^i{}_{\mathsf{ID}_u}$	The i-th updated verify point of group member ID_u.
vps_{ID_u}	Verify points of group member ID_u.
vps_G	Group member authentication information.
$sd^i_{\mathsf{ID}_u}$	Secret seed for generating the passwords of the i-th verify epoch of ID_u.
sk_{ID_u}	Secret key of ID_u.
$pw^{i,j}_{\mathsf{ID}_u}$	The j-th GTOTP password of the i-th verify epoch of ID_u.
$T_{current}, T$	Current timestamp and a certain timestamp, respectively.
Δ_{vp}, Δ_{pw}	Valid period for each verify point and password generation interval, respectively.

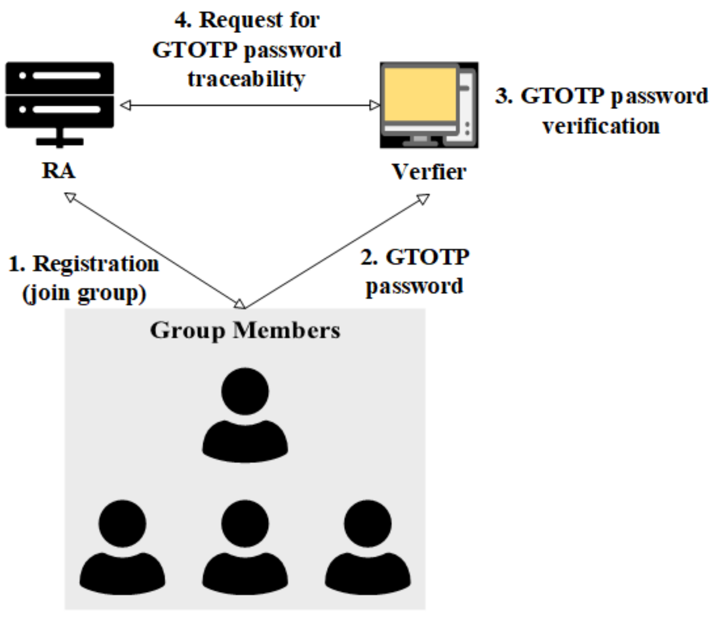

4. Request for GTOTP password traceability
RA
Verfier
3. GTOTP password verification
1. Registration (join group)
2. GTOTP password
Group Members

Figure 1. Flow chart of GTOTP scheme.

In a GTOTP scheme, each group member generates V TOTP instances for generating TOTP passwords. To prevent linkability between verify points, each TOTP instance has a valid period Δ_{vp}, i.e., each verify point (or verify epoch) has a validity period of Δ_{vp}. Figure 2 shows the hash chains used by the group members to generate the TOTP passwords and verify points. The group members compute the secret seed $sd^i_{\mathsf{ID}_u}$ of the i-th hash chain based on the current time $T_{current}$ and the start time T_{start} of the GTOTP instance as well as the valid period Δ_{vp}, then compute the j-th node in the chain as the TOTP password in combination with the password generation interval Δ_{pw}. This is used as part of the GTOTP password, where $i := \frac{\lceil T_{current} - T_{start} \rceil}{\Delta_{vp}}$ and $j := \frac{\lceil T_{current} - T_{start} - i \cdot \Delta_{vp} \rceil}{\Delta_{pw}}$.

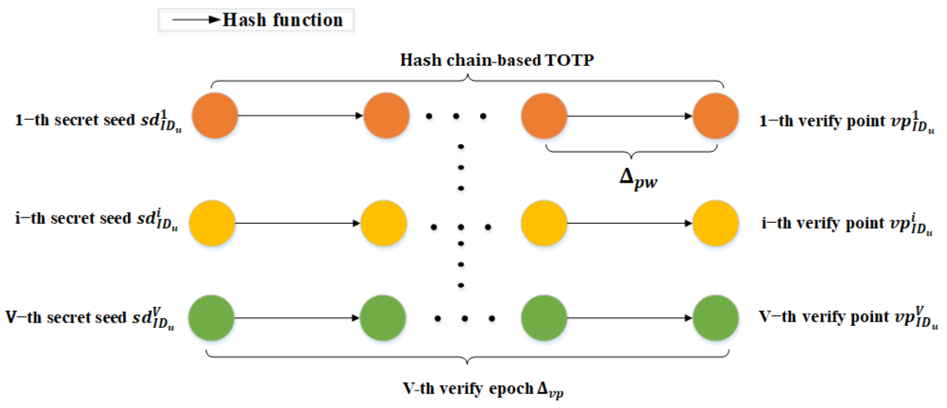

Hash function
Hash chain-based TOTP
1−th secret seed $sd^1_{ID_u}$
1−th verify point $vp^1_{ID_u}$
Δ_{pw}
i−th secret seed $sd^i_{ID_u}$
i−th verify point $vp^i_{ID_u}$
V−th secret seed $sd^V_{ID_u}$
V−th verify point $vp^V_{ID_u}$
V-th verify epoch Δ_{vp}

Figure 2. Structure of the hash chains.

When group members register with the RA, they are required to send the V verify points of the hash chains. The RA receives the set of verify points sent from all group members and computes the group authentication information vps_G. To achieve membership traceability, the RA encrypts the identity of each group member using an authenticated symmetric encryption (ASE) algorithm [24] and binds the identity cipher $C^i{}_{\mathrm{ID}_u}$ to each verify point using a collision-resistant hash function, i.e., the RA updates each verify point. To achieve anonymity, RA uses a permutation scheme [25] to scatter all updated verify points. For better performance, the RA divides all updated verify points into z subsets, constructs a Merkle tree [26] on each subset, and inserts all Merkle tree roots into a Bloom filter [27] for GTOTP passwords verification, as depicted in Figure 3.

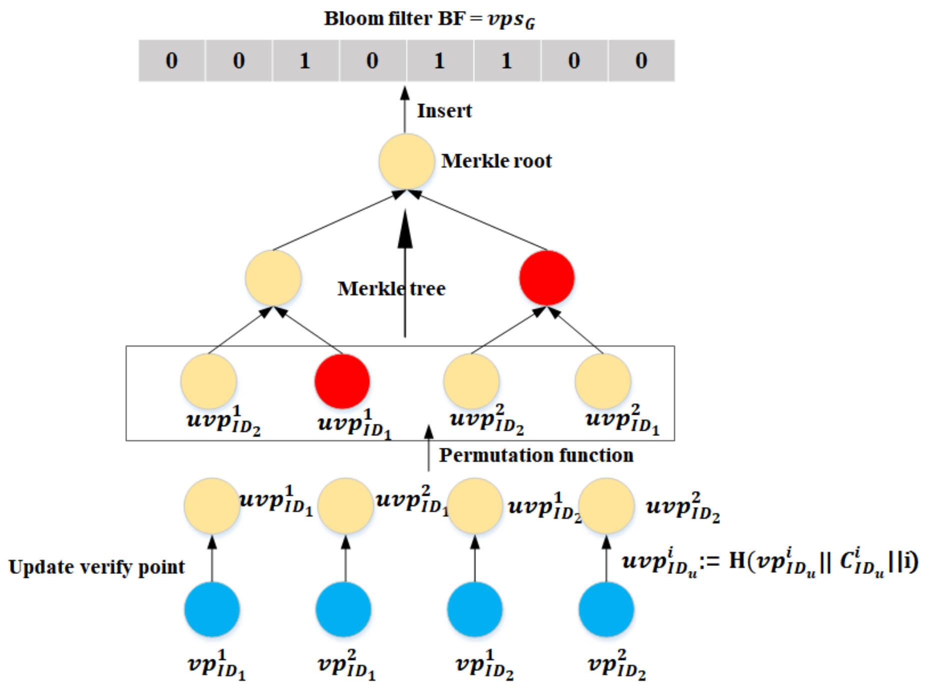


Figure 3. Overview of the process of group member authentication information vps_G generation, where $M = 2$, $V = 2$, $z = 1$. The red nodes in the figure represent the Merkle proof of the updated verification point $uvp^1{}_{\mathrm{ID}_2}$ (shown as an example).

A GTOTP password contains the password of the TOTP password, the identity ciphertext of the group member, and the Merkle proof of the current updated verify point. In the verification phase, after the verifier updates the verify point using the same method it generates a Merkle tree root using the updated verify point and related Merkle proof, checks whether the root is in the Bloom filter and whether the TOTP password is valid, then, if both of the above pass the verification, the GTOTP password is valid and 1 is returned. This process consists of seven algorithms $(\mathsf{RASetup}, \mathsf{GMInit}, \mathsf{GVKGen}, \mathsf{GetSd}, \mathsf{PdGen}, \mathsf{Verify}, \mathsf{IDOpen})$, described in detail as follows:

- $\mathsf{RASetup}(1^\kappa, \Delta_{pw}, \Delta_{vp}, T_{start}, T_{end})$ takes the security parameter 1^κ, the password generation interval Δ_{pw}, the valid period for each verify point Δ_{vp}, and the start T_{start} and end T_{end} times of the protocol instance as inputs. The RA runs this algorithm to generate the system parameters smp and its secret key $\mathsf{K}_{\mathsf{RA}} \overset{\$}{\leftarrow} \mathcal{K}_{\mathsf{RA}}$, where $\mathcal{K}_{\mathsf{RA}}$ is the secret key space of the RA.
- $\mathsf{GMInit}(\mathsf{ID}_u)$ takes the group member identities ID_u as inputs; each group member runs this initialization algorithm to generate its secret key $sk_{\mathsf{ID}_u} \overset{\$}{\leftarrow} \mathcal{K}_{\mathsf{GM}}$ and the set of verify points vps_{ID_u}, where $\mathcal{K}_{\mathsf{GM}}$ is the secret key space for group members.
- $\mathsf{GVKGen}(\mathsf{GM}, \{vps_{\mathsf{ID}_u}\}_{u\in[M]})$ takes the identity and set of verify points for all group members as inputs; the RA runs this algorithm to generate the group member authentication information vps_G and secret outputs $\{St_{\mathsf{ID}_u}\}_{u\in[M]}$ for ID_u. $\mathsf{GetSd}(\mathsf{K}_{\mathsf{ID}_u}, T)$ takes the secret key sk_{ID_u} and a timestamp T as inputs; member ID_u runs this algorithm to generate the secret seed $sd^i{}_{\mathsf{ID}_u}$ (i.e., the head of the hash chain-based TOTP instance) used to generate the password at T of the i-th verify period.
- $\mathsf{GVKGen}(\mathsf{GM}, \{vps_{\mathsf{ID}_u}\}_{u\in[M]})$ takes the identity and set of verify points for all group members as inputs; the RA runs this algorithm to generate the group member authentication information vps_G and the secret outputs $\{St_{\mathsf{ID}_u}\}_{u\in[M]}$ for ID_u. $\mathsf{GetSd}(\mathsf{K}_{\mathsf{ID}_u}, T)$ takes the secret key $\mathsf{K}_{\mathsf{ID}_u}$ and a timestamp T as inputs; member ID_u runs this algorithm to generate the secret seed $sd^i{}_{\mathsf{ID}_u}$ (i.e., the head of the hash chain-based TOTP instance) used to generate the password at T of the i-th verify period.
- $\mathsf{PdGen}(sd^i{}_{\mathsf{ID}_u}, T)$ takes the secret seed $sd^i{}_{\mathsf{ID}_u}$ and a timestamp T as inputs; member ID_u runs this password generation algorithm to generate a GTOTP password $pw^{i,j}{}_{\mathsf{ID}_u}$ for the j-th password of the i-th verify period.
- $\mathsf{Verify}(pw^{i,j}{}_{\mathsf{ID}_u}, T, vps_G)$ takes the password $pw^{i,j}{}_{\mathsf{ID}_u}$, the group verification information vps_G, and a timestamp T as inputs. The verifier runs this password verification.
- $\mathsf{IDOpen}(\mathsf{K}_{\mathsf{RA}}, pw^{i,j}{}_{\mathsf{ID}_u}, T)$ takes the secret key of the RA, the password $pw^{i,j}{}_{\mathsf{ID}_u}$, and a timestamp T as inputs. The RA runs this identity tracing algorithm to open the identity of group member ID_u.

3.2. Public Key Encryption

A public key encryption (PKE) scheme consists of three algorithms $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ [28,29]. The key generation algorithm $\mathsf{KGen}(1^\kappa)$ takes the security parameter κ as input and outputs a key pair $(\mathsf{pk}, \mathsf{sk})$. The encryption algorithm $\mathsf{Enc}(m, \mathsf{pk})$ takes a plaintext $m \in \mathcal{M}$ and public key pk as inputs and outputs a ciphertext $c \in \mathcal{C}$, where $\mathcal{M}$ and $\mathcal{C}$ are the plaintext space and ciphertext space, respectively. The decryption algorithm $\mathsf{Dec}(c, \mathsf{sk})$ takes a ciphertext c and the secret key sk as inputs and outputs the plaintext m. Here, we need a PKE scheme that can resist chosen-plaintext attacks.

3.3. Threat Model

Our PPPoL scheme focuses on the identity and location privacy of the prover and witnesses, including three security properties: traceability, anonymity, and location privacy. Although an attacker may corrupt the witnesses, we assume that most of the witnesses are honest when generating location proofs with the prover. The verifier honestly verifies and sends the verification result of the location proof to the trusted third-party RA. Neither the RA nor the verifier will collude with an attacker. Suppose the communication between the participants and the RA is secure, but the communication between the participants may be controlled by the attacker, such as by intercepting and tampering with the information. Informally speaking, traceability needs to prevent an attacker from forging a valid location proof, even if such a location proof is only traceable to an invalid or dishonest participant. Anonymity requires that the identity of the prover and the witnesses be hidden from other participants. Location privacy means that an attacker cannot obtain any information about the location of uncorrupted witnesses from a location proof.

4. Review of the PoL Scheme Based on GTOTP

In this section, we provide a brief overview of the PoL scheme proposed by Yang et al. [10] based on GTOTP. This scheme leverages GTOTP, a privacy-preserving location proximity (PPLP) scheme that runs between the requesting party P and the responding party W to determine whether their locations are proximate, along with a hash-based commitment scheme. The scheme is described below.

During the Initialization phase, the RA manages the registration of both the prover and the witnesses, and generates and publishes the relevant system parameters.

During the location proof generation phase, the prover first broadcasts the GTOTP password to nearby witnesses. Upon successfully verifying the prover's password, each witness engages in the PPLP protocol with the prover to determine whether they are in proximity. The witness commits the results of the PPLP computation using the GTOTP secret seed used in the current verify period and commits the relevant location proof information sent to the prover using the next used GTOTP password. When the secret seed expires, the witness publishes the associated commitment keys and the prover verifies the identity of the witnesses and the proximity of the locations between them. The prover finally filters the valid location proof pieces made by the witnesses to obtain the location proof, commits the location proof using the GTOTP secret seed used in the current verify period, commits the relevant location proof information sent to the verifier using the next used GTOTP password, and finally sends the relevant location proof information to the verifier.

During the location proof verification phase, after the commitment key expires, the prover reveals the relevant commitment keys. The verifier then uses these commitment keys along with the respective GTOTP passwords of the witnesses and the prover to verify the validity of the location proof.

Existing Problems. In the PoL scheme proposed by Yang et al. [10], we observe that location proofs are only validated after the commitment keys of the prover and relevant witnesses expire. This characteristic is not suitable for instant location verification. In addition, the requirement for the PPLP protocol to run between the witnesses and the prover leads to increased storage and computational overhead, potentially making it unsuitable for resource-constrained devices. Finally, the scheme does not consider the motivation for witnesses to participate in the PoL scheme; thus, witnesses might refuse to assist the prover in generating location proofs.

5. Privacy-Preserving Proof of Location Scheme

In this section, we present our efficient PPPoL scheme based on GTOTP and a commitment scheme.

Our PPPoL scheme consists of four entities: the Registration Authority (RA), prover, witness, and verifier. The prover generates a location proof that confirms its presence at a specific location at a specific time. The prover is a user device (such as a smartphone, IoT device, or smart vehicle) that generates a location proof located at a specific location at a specific time. A witness is either a device similar to the prover or a location beacon such as a Roadside Unit (RSU) that assists the prover in generating the location proof based on the location's proximity to the prover. The verifier may be a location service provider, a government organization, or another entity that needs to verify the location of the prover on the basis of the location proof. The RA manages the registration of the prover, witnesses, and verifier. Figure 4 provides an overview of an outdoor application scenario for the PPPoL scheme. The prover and nearby witnesses obtain location information through GPS and utilize Dedicated Short-Range Communications (DSRC) technology to communicate and generate a location proof. Subsequently, the prover and witnesses transmit the location proof and verification information to the verifier using 5G or other communication technologies.

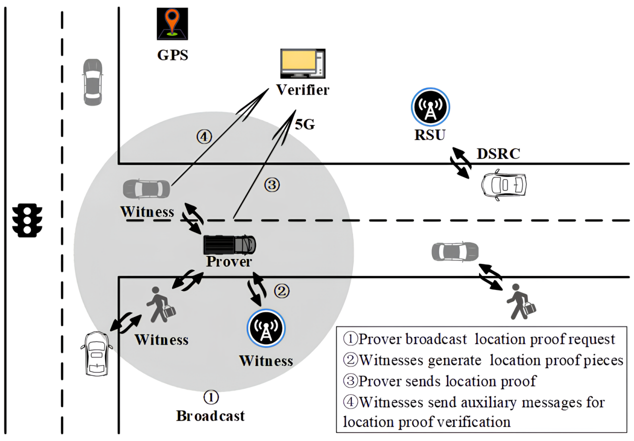

GPS
Verifier
5G
RSU
DSRC
④
③
Witness
Prover
②
Witness
Witness
①
Broadcast
①Prover broadcast location proof request
②Witnesses generate location proof pieces
③Prover sends location proof
④Witnesses send auxiliary messages for location proof verification

Figure 4. Overview of PPPoL scheme.

5.1. Security Requirements

Informally, we articulate that our PPPoL scheme should satisfy the following security requirements:

Anonymity. Both prover and witnesses aim to preserve their identity privacy, ensuring that attackers cannot extract their identity information from location proofs.

Traceability. Only valid location proofs can be traceable to honest provers; in other words, even if a forged location proof is valid, it cannot be traced back to the corresponding honest user.

Location Privacy. Witnesses aspire to assist in generating location proofs without compromising their location privacy, that is, attackers cannot extract the location information of witnesses from the location proofs.

5.2. Description of the Scheme

In a decentralized PoL system, the prover obtains the location proof from nearby witnesses that can verify its location. We generate the location proof based on the location proximity between the prover and witnesses. We assume that each participant has an internal clock that synchronizes with the other participants.

We need to design an efficient PPPoL scheme that ensures the identity and location privacy of entities while supporting instant location verification. To encourage witnesses' participation in our PPPoL scheme, we incorporate an incentive mechanism based on smart contracts, which is detailed in the next section. We leverage the anonymity provided by GTOTP to achieve identity privacy for both witnesses and prover. We leverage the traceability provided by GTOTP and the commitment scheme to ensure the traceability of location proofs. We protect the location privacy of witnesses using a PKE scheme that can resist chosen-plaintext attacks. To enable our PPPoL scheme to support instant location verification, we set the commitment keys of witnesses and the prover as their respective GTOTP passwords (the password expiration time can be adjusted according to specific application scenarios).

We consider our PPPoL scheme with three interactive sub-protocols, i.e., the initialization Setup, the location proof generation LPGen, and the location verification LPVer. The important notation for our PPPoL scheme is described in Table 2.

Figure 5 shows the flow of the PPPoL scheme and incentive mechanism. We describe the details of our incentive mechanism in the next section. To delineate the procedural flow of our scheme, solid and dotted arrows are employed to depict communication between entities, denoting request and response operations, respectively.

Table 2. Notation for PPPoL scheme.

Notation	Description
$\mathsf{P}, \mathsf{W}, \mathsf{V}$	Identities of prover, witness and verifier, respectively.
X	Number of witnesses.
$\mathsf{L}_{\mathsf{P}}, \mathsf{L}_{\mathsf{W}_x}$	Locations of P and W_x, respectively.
$\mathsf{LPC}_{\mathsf{W}_x}, \mathsf{LPP}_{\mathsf{W}_x}$	Location proximity commitment and location proof piece of W_x, respectively.
$\mathsf{LC}, \mathsf{LP}_{\mathsf{P}}$	Location commitment and location proof of P.
$\mathsf{vk}_{\mathsf{PoL}}$	Group authentication key.
smp	System parameter.

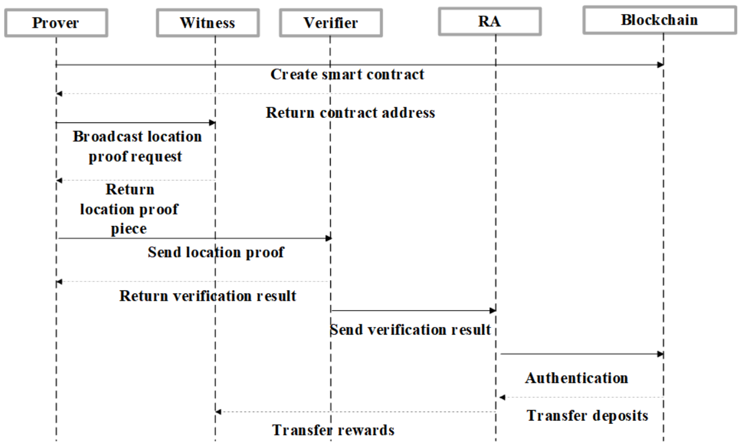


Figure 5. Flow of PPPoL scheme and incentive mechanism.

5.2.1. Setup of PPPoL Scheme

In the initialization phase of the PPPoL scheme, group members, who are either witnesses or the prover, engage with the RA to generate the group authentication key, denoted as $\mathsf{vk}_{\mathsf{PoL}}$. We assume that the communication between RA and entities is secure (using a mutual authentication channel such as TLS [30]) and that the prover and witnesses communicate with each other over a short-range communication channel such as BlueTooth [31]. Figure 6 illustrates the detailed process of sub-protocol Setup.

5.2.2. Location Proof Generation of PPPoL

The prover generates a location proof with the assistance of nearby witnesses following the process depicted in Figure 7. The prover broadcasts its GTOTP password and location cipher with nearby users. If a nearby witness verifies this GTOTP password successfully, then the witness encrypts the location data using the verifier's ephemeral public key, uses the next used GTOTP password as the commitment key for committing the location cipher, and obtains the location proximity commitment. The witness sends the location proximity commitment, the currently valid GTOTP password, and the location cipher to the prover over the short-range communication channel. The prover verifies the identity of the witnesses using their respective GTOTP passwords, integrates the location proof pieces of all witnesses into a location proof, and uses the next used GTOTP password as a commitment key for committing the location proof with a location commitment.

Input: Security parameter 1^κ, start time T_{start} and end time T_{end} of the protocol, location proof generation interval Δ_{pw}, the valid period for each verify point Δ_{vp}
Output: Group authentication key $\mathsf{vk_{PoL}}$

1. RA generates the system parameter smp in the following steps:
   - Runs $\mathsf{smp} := \mathsf{GTOTP.RASetup}(1^\kappa, \Delta_{pw}, \Delta_{vp}, T_{start}, T_{end})$;
   - Publishes the system parameter smp.
2. Each group member (i.e., witnesses and prover) requests enrollment with the RA by the following steps:
   - Generates enrollment request by running the initialization algorithm $(sk_{\mathsf{ID}_u}, vps_{\mathsf{ID}_u}) := \mathsf{GTOTP.GMInit}(\mathsf{ID}_u)$;
   - Sends the initialized set of verify points vps_{ID_u} to RA via the mutual authentication channel;
   - Stores the secret key $\mathsf{sk}_{\mathsf{ID}_u}$ locally.
3. After receiving the vps_{ID_u} from all group members, RA finishes member registration and protocol initialization as follows:
   - Runs $(vps_G, \{St_{\mathsf{ID}_u}\}_{u\in[M]}) := \mathsf{GTOTP.GVKGen}(\mathsf{GM}, \{vps_{\mathsf{ID}_u}\}_{u\in[M]})$;
   - Publishes the group authentication key $\mathsf{vk_{PoL}} = vps_G$;
   - Sends the corresponding secret output $\{St_{\mathsf{ID}_u}\}_{u\in[M]}$ for each group member ID_u via the mutual authentication channel.

Figure 6. Setup of PPPoL.

Input: Current time $T_{current}$, the location $\mathsf{L_P}$ of prover, the locations $\{\mathsf{L}_{\mathsf{W}_x}\}_{x\in[X]}$ of witnesses, the secret key sk_{P} of prover, the secret keys $\{sk_{\mathsf{W}_x}\}_{x\in[X]}$ of witnesses
Output: Location proof $\mathsf{LP_P}$

1. The steps for generating location proof requests by prover are outlined as follows:
   - Requests an ephemeral public key $\mathsf{epk_V}^{ctr}$ generated by running the key generation algorithm $(\mathsf{epk_V}^{ctr}, \mathsf{esk_V}^{ctr}) := \mathsf{PKE.KGen}(1^\kappa)$ from the verifier, where ctr is a counter stored by the verifier to tag the corresponding location proof;
   - Generates the currently used secret seed $sd_{\mathsf{P}}{}^{i} := \mathsf{GTOTP.GetSd}(sk_{\mathsf{P}}, T_{current})$, the GTOTP password $pw_{\mathsf{P}}{}^{i,j} := \mathsf{GTOTP.PdGen}(sd_{\mathsf{P}}{}^{i}, T_{current})$;
   - Encrypts the location $\mathsf{C_P} := \mathsf{PKE.Enc}(\mathsf{L_P}, \mathsf{epk_V}^{ctr})$;
   - Broadcasts location proof request $(\mathsf{C_P}, pw_{\mathsf{P}}{}^{i,j})$ to nearby witnesses via a short-range communication channel.
2. After receiving the request $(\mathsf{C_P}, pw_{\mathsf{P}}{}^{i,j})$, each witness verifies the password $pw_{\mathsf{P}}{}^{i,j}$, and if the $\mathsf{GTOTP.Verify}(pw_{\mathsf{P}}{}^{i,j}, T_{current}, \mathsf{vk_{PoL}}) = 1$, then it does the following:
   - Generates the currently used secret seed $sd_{\mathsf{W}_x}{}^{i} := \mathsf{GTOTP.GetSd}(sk_{\mathsf{W}_x}, T_{current})$, the GTOTP password $pw_{\mathsf{W}}{}^{i,j} := \mathsf{GTOTP.PdGen}(sd_{\mathsf{W}}{}^{i}, T_{current})$;
   - Encrypts the location $\mathsf{C}_{\mathsf{W}_x} := \mathsf{PKE.Enc}(\mathsf{L}_{\mathsf{W}_x}, \mathsf{epk_V}^{ctr})$;
   - If $T_{curent} + \Delta_{pw}$ belongs to current secret seed period, then calculates password $pw_{\mathsf{W}}{}^{i,j+1} := \mathsf{GTOTP.PdGen}(sd_{\mathsf{W}}{}^{i}, T_{current} + \Delta_{pw})$ as location proximity commitment key, otherwise the proof generation fails;
   - Calculates the location proximity commitment $\mathsf{LPC}_{\mathsf{W}_x}$ by $\mathsf{LPC}_{\mathsf{W}_x} := \mathsf{H}(pw_{\mathsf{W}}{}^{i,j+1} \| \mathsf{C_P} \| \mathsf{C}_{\mathsf{W}_x} \| T_{current})$;
   - Sends the location proof piece $\mathsf{LPP}_{\mathsf{W}_x} = (pw_{\mathsf{W}}{}^{i,j}, \mathsf{C}_{\mathsf{W}_x}, \mathsf{LPC}_{\mathsf{W}_x})$ to the prover via a short-range communication channel.
3. After receiving the proof pieces $\{\mathsf{LPP}_{\mathsf{W}_x}\}_{x\in X}$ of all witnesses, the prover generates location proof $\mathsf{LP_P}$ as follows:
   - Verifies the password $pw_{\mathsf{W}}{}^{i,j}$ of each witness, and if $\mathsf{GTOTP.Verify}(pw_{\mathsf{W}_x}{}^{i,j}, T_{current}, \mathsf{vk_{PoL}}) = 0$, the prover removes the proof piece of the corresponding witness;
   - If $T_{curent} + \Delta_{pw}$ belongs to current secret seed period, and the prover calculates password $pw_{\mathsf{P}}{}^{i,j+1} := \mathsf{GTOTP.PdGen}(sd_{\mathsf{P}}{}^{i}, T_{current} + \Delta_{pw})$ as the location commitment key;
   - Computes the location commitment $\mathsf{LC_P} := \mathsf{H}(\{\mathsf{LPP}_{\mathsf{W}_x}\}_{x\in[X]} \| \mathsf{L_P} \| T_{current} \| \mathsf{epk_V}^{ctr} \| pw_{\mathsf{P}}{}^{i,j+1})$;
   - Sends location proof $\mathsf{LP_P} = (\{\mathsf{LPP}_{\mathsf{W}_x}\}_{x\in[X]}, \mathsf{C_P}, \mathsf{LC_P}, pw_{\mathsf{P}}{}^{i,j}, T_{current}, \mathsf{epk_V}^{ctr})$ to the verifier via a unilateral authentication channel.

Figure 7. Location proof generation of PPPoL.

5.2.3. Location Proof Verification of PPPoL

The verifier checks the validity of the location proof using the group verification key $\mathsf{vk_{PoL}}$ and the expired commitment keys of the prover and witnesses, as shown in Figure 8. As the validity of the GTOTP password is short, the prover and related witnesses can quickly send the related expired commitment keys to the verifier for commitment verification, i.e., the verifier can instantly verify the location of the prover. The verifier uses its private key that decrypts the location ciphertext of the witnesses and the prover and calculates the proximity between the locations. If most of the witnesses and the prover are in proximity and if the commitment verification is successful, then the location proof is considered to be valid.

Input: Group authentication key $\mathsf{vk}_{\mathsf{PoL}}$, location proof LP_{P}
Output: Verification result $vrt \in \{0,1\}$

1. After the passwords used for commitment expire, the prover and the relevant witnesses involved in the location proof generation send the expired commitment keys to the verifier over a unilateral authentication channel.
2. Upon receiving the location proof LP_{P}, the commitment keys of the relevant witnesses and prover, the verifier verifies the location proof LP_{P} steps as follows:
   - Verifies the GTOTP passwords of the prover $\mathsf{GTOTP.Verify}(pw_{\mathsf{P}}{}^{i,j}, T_{current}, \mathsf{vk}_{\mathsf{PoL}})$, and if the passwords are invalid, then returns 0, i.e., the location proof LP_{P} is invalid;
   - Verifies the GTOTP passwords of relevant witnesses $\mathsf{GTOTP.Verify}(pw_{\mathsf{W}_x}{}^{i,j}, T_{current}, \mathsf{vk}_{\mathsf{PoL}})$, and removes the proof pieces that fails verification;
   - Verifies each location proximity commitment $\mathsf{LPC}_{\mathsf{W}_x}$. If $\mathsf{LPC}_{\mathsf{W}_x} \neq \mathsf{H}(pw_{\mathsf{W}}{}^{i,j+1} \| \mathsf{C}_{\mathsf{P}} \| \mathsf{C}_{\mathsf{W}_x} \| T_{current})$, then verifier removes the proof piece that verifies invalid;
   - If $\mathsf{LC}_{\mathsf{P}} \neq \mathsf{H}(\{\mathsf{LPP}_{\mathsf{W}_x}\}_{x \in \mathsf{X}} \| \mathsf{L}_{\mathsf{P}} \| T_{current} \| \mathsf{epk}_{\mathsf{V}}{}^{ctr} \| pw_{\mathsf{P}}{}^{i,j+1})$, then returns 0;
   - Decrypts the location ciphertext C_{P} and $\mathsf{C}_{\mathsf{W}_x}$ of the prover and the relevant witnesses using the ephemeral private key $\mathsf{esk}_{\mathsf{V}}{}^{ctr}$;
   - Calculates the Euclidean distance between the prover and each witness based on their location and records the location proof pieces that are within the proximity threshold θ.
3. If all the above verifications pass and the number of valid location proof pieces $l \geq q$, then 1 is returned, i.e., the location proof is valid.

Figure 8. Location proof verification of PPPoL.

5.3. Security Analysis

In this section, we informally discuss how our scheme achieves the three aforementioned security properties, i.e., anonymity, traceability, and location privacy. The security of our PPPoL scheme relies on standard cryptographic protocols, including the GTOTP scheme, PKE scheme, and collision-resistant hash function. Suppose that the PKE encryption algorithm can resist chosen-plaintext attacks and that the used hash function H is collision-resistant. The cryptographic building blocks employed in constructing the GTOTP, consisting of an ASE encryption algorithm resistant to chosen-ciphertext attacks, a secure Merkle tree scheme, a secure TOTP scheme, an unpredictable permutation function, and a secure pseudo-random function, are assumed to be provably secure.

Anonymity. The authentication between the prover and witnesses, as well as between them and the verifier, is achieved through the employment of the GTOTP passwords. A GTOTP password comprises three components: a TOTP password, a Merkle proof, and a group membership ciphertext. The hash chain-based TOTP instances utilize a pseudo-random function to generate the secret seed, and employ a collision-resistant hash function to compute the TOTP password and the verify point. Because the output of the hash function and the ASE encryption algorithm incorporate randomness, an attacker cannot distinguish the updated verify points of the honest group members. By employing an unpredictable permutation function to shuffle the order of all updated verify points, attackers are unable to associate the Merkle proofs with the updated verify points of the honest group members, and consequently cannot distinguish the correspondence between the verify points indicated by the leaf nodes of the Merkle tree and the honest group members. In addition, each Merkle tree root is inserted into the Bloom filter, while multiple Merkle tree roots may be inserted into the same location in the Bloom filter; thus, the attacker is unable to recover the relationship between Merkle tree roots from the Bloom filter. As the RA encrypts the group membership identity using an ASE encryption algorithm capable of resisting chosen-ciphertext attacks, the attacker cannot extract any valid information about the identity from the group membership ciphertext. In summary, GTOTP offers a provably secure anonymity property, and utilizing the GTOTP password ensures that other participants cannot acquire any additional identity information of group members from the GTOTP password. Consequently, the PPPoL scheme achieves anonymity between entities and guarantees identity privacy for both the prover and witnesses. For interested readers, further details are available in Section 4 of [10].

Traceability. Attackers cannot forge a GTOTP password that is valid and traceable to an honest group member. Thanks to the one-way hash function and the security of the Merkle tree scheme, attackers are unable to forge a both valid TOTP password and a Merkle proof. In addition, because the ASE encryption algorithm resists chosen-ciphertext attacks, attackers are unable to distinguish between group memberships, and as a result are unable to forge a valid GTOTP password that binds to the identity of an honest group member. The RA, on the other hand, can correctly trace the identity of the corresponding group member through the GTOTP password. Thus, the GTOTP password provides the traceability property. Utilizing the provably secure traceability property provided by GTOTP ensures that any valid GTOTP password can be accurately traced back to honest provers and witnesses. Utilizing a hash commitment scheme binds the unexpired GTOTP passwords of the provers and witnesses to the location proofs, preventing an attacker from obtaining unexpired GTOTP passwords in order to impersonate the relevant honest provers and witnesses. In this way, the privacy location proof scheme provides traceability of location proofs, where a valid location proof traces back accurately to the prover who generated it, while an attacker's forged location proof is either invalid or can be traced back to a dishonest user. We recommend that interested readers consult Section 4 of [10] for further details.

Location Privacy. The locations of the witnesses are encrypted using a PKE scheme that can resist chosen-plaintext attacks; the location cipher is then committed with a GTOTP password. As GTOTP provides traceability, it prevents attackers from forging a valid GTOTP password that can be traced back to an honest witness. By employing GTOTP passwords as the commitment keys for the location ciphertext of witnesses, the PKE scheme is secure against chosen-plaintext attacks, ensuring the location privacy of witnesses. In this way, our PPPoL scheme is able to ensure the witnesses' location privacy.

6. Incentive Mechanism

Most decentralized proof of location schemes ignore witnesses' motivation to engage with the location proof, which may cause the leakage of witnesses' private information as well as increased computational resources overhead. To solve this issue, we designed an incentive mechanism that rewards witnesses who behave honestly and provide valid location proofs, thereby increasing the motivation of witnesses to engage with our PPPoL scheme.

When designing an incentive mechanism, potential attacks and threats should first be considered. Potential threats include an attacker impersonating the identity of witnesses to receive rewards, witnesses providing invalid location proofs in order to receive rewards, or attackers trying to obtain the private information of the prover and the witnesses. Simultaneously, the incentive mechanism should accurately reward honest witnesses who provide valid location proofs.

Overview. Smart contracts are decentralized and tamper-proof; the two parties to the contract agree on the transaction rules and triggering conditions, which automatically executes the relevant transfers. Therefore, we incorporate smart contracts to complete our incentive mechanism using a blockchain. Figure 9 describes the system model of the incentive mechanism.

To protect the privacy of the witnesses, we verify the identity of the RA in the smart contract, then the smart contract automatically transfers the contract deposit to the RA, which provides rewards to those witnesses who honestly provide valid location proofs. We assume that RA serves as a trusted third party who will not collude with any participant or attacker and will honestly transfer rewards to witnesses. Our incentive mechanism consists of three main phases: contract deployment Setup, contract interaction $\mathsf{Interaction}$, and rewards Reward, with the details described below.

Setup. The prover deploys the smart contract on the blockchain before initiating the location proof request, which requires the prover to pay the deposit to the contract advance to ensure the incentives for the witnesses. Following successful deployment, the prover broadcasts the location request to nearby witnesses together with the address of the smart contract.

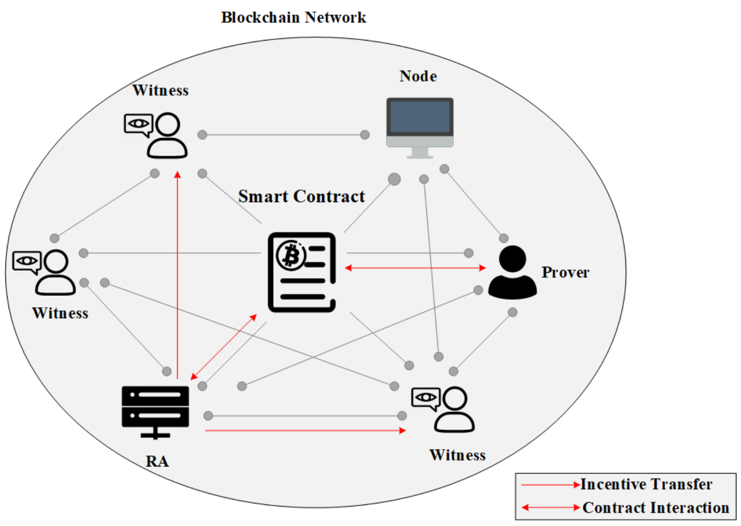


Figure 9. System model diagram of incentive mechanism.

Interaction. After receiving a request from the prover to generate a location proof, the witness first checks whether the deposit stored by the prover in the smart contract satisfies the condition $r \geq \delta$ to ensure that the corresponding reward can be obtained upon completing the location proof. Here, r represents the deposit made by the prover into the smart contract and δ is the threshold, which is the minimum amount to be rewarded to the witness. Only when the deposit stored by the prover in the smart contract meets the payment condition will the witness respond to the prover's request to generate a location proof; otherwise, the witness will reject the request for location proof generation. If this verification is successful, then the prover sends the location proof to the verifier while sending the location cipher commitment $com_{\mathsf{P}} = \mathsf{H}(pw_{\mathsf{P}}{}^{i,j+1}||\mathsf{C}_{\mathsf{P}})$ to the smart contract. During the location proof verification phase, the prover sends the expired commitment key $pw_{\mathsf{P}}{}^{i,j+1}$ to the smart contract. The details are described in Algorithm 1 below.

Algorithm 1 Interaction

```
Input: Location proof requests
Output: $\mathsf{LP}_{\mathsf{P}}, com_{\mathsf{P}}$
1: for $i := 1$ to $n$ do
2:   if $r \geq \delta$ then
3:     return $\mathsf{LP}_{\mathsf{P}} \leftarrow \mathsf{LPGen}(\mathsf{L}_{\mathsf{P}}, sk_{\mathsf{P}}, \{\mathsf{L}_{\mathsf{W}_x}\}_{x\in[X]}, \{sk_{\mathsf{W}_x}\}_{x\in[X]}, T_{current})$
4:     return $com_{\mathsf{P}} = \mathsf{H}(pw_{\mathsf{P}}{}^{i,j+1}||\mathsf{C}_{\mathsf{P}})$
5:   end if
6:   ContractAddr.transfer(PoverAddr) // The prover sends $com_{\mathsf{P}}$ to the smart contract
7: end for
```

Reward. The RA verifies the location cipher commitment com_{P} uploaded by the prover in the smart contract. If the com_{P} is valid, then the RA signs the com_{P} using its private key and sends the signature σ_{RA} to the smart contract. As the smart contract is triggered by the incoming data from the RA and the prover, the contract automatically verifies the validity of the signature σ_{RA} of the RA and the location cipher commitment com_{P} of the prover. If both verifications pass, the deposit in the contract is transferred to the RA. Finally, the RA combines the valid location proof pieces for tracing back to the witnesses and pays the rewards to them. The details are described in Algorithm 2 below.

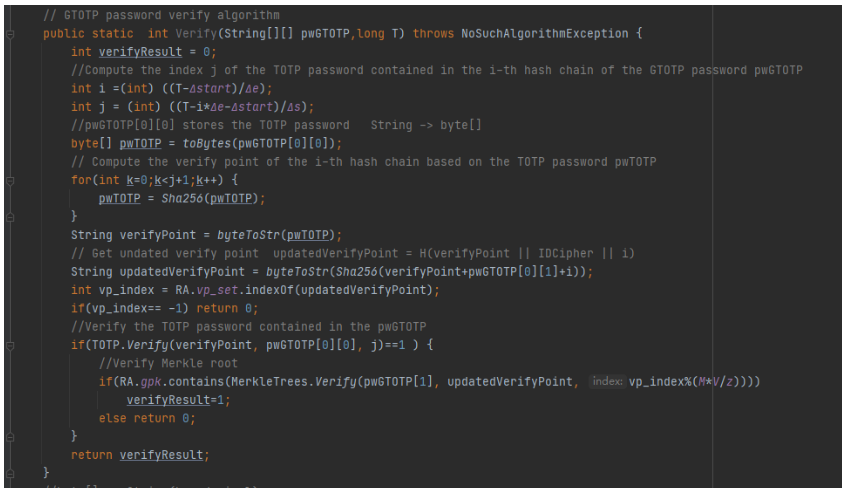
Algorithm 2 Reward

```
Input: $pw_{\mathsf{P}}{}^{i,j+1}, \mathsf{C}_{\mathsf{P}}, \sigma_{\mathsf{RA}}, comp_{\mathsf{P}}$
Output: Incentive fees paid to witnesses
1: if $comp_{\mathsf{P}} \neq \mathsf{H}(pw_{\mathsf{P}}{}^{i,j+1}||\mathsf{C}_{\mathsf{P}})$ then
2:     return "Failed authentication of the prover"
3: end if
4: if $\mathsf{ecrecove}(\sigma_{\mathsf{RA}}) = \mathsf{false}$ then
5:     return "Failed authentication of the RA"
6: end if
7: RAAddr.transfer(ContractAddr) // Contract deposit transferred to RA
8: for $i := 1$ to $n$ do
9:     WinnessAddr[$i$].transfer(RAAddr) // RA pays rewards to relevant witnesses
10: end for
```

7. Practical Evaluation

We ran the prover, witnesses, and verifier on a real Android smartphone with a Snapdragon 855 CPU (up to 2.84 GHz) and 8 GB RAM, and ran the RA on a 64-bit PC with an Intel(R) Core(TM) i7-10700 at 2.90 GHz and 16 GB RAM. We used Android Studio http://developer.android.com/studio/ (accessed on 8 July 2023) for programming and installed it on the devices of the witnesses, prover, and verifier. Our PPPoL scheme is programmed in Java https://www.java.com/en/ (accessed on 8 July 2023) .

We used SHA256 to implement the hash function in our PPPoL scheme and implemented the effective GTOTP scheme $\mathsf{GTOTP\text{-}MT}$ from [10]; the implementation of PKE scheme used ElGamal encryption [32]. Figure 10 shows the code of the GTOTP password verification algorithm. The verify points can be used as pseudonyms for group members. Based on the recommended validity of a pseudonym by the European Telecommunications Standards Institute (ETSI), we set the validity of the verify points to $\Delta_{vp} = 5$ min. We considered the selection of parameter values based on previous PoL and GTOTP schemes and the GTOTP password generation interval $\Delta_{pw} = 5$ s; thus, each TOTP instance consisted of $K = 60$ one-time passwords. We set the proximity threshold in our PPPoL scheme to $\theta = 50$ m, and the verify points of all members were divided into $z = 2^{13}$ subsets. Unless otherwise specified, we set V = 105,120, which is enough to run our PPPoL for a year. We set $M = 20$ and $X = 5$. Of course, these parameters could be set to other values; however, as the selected test values have been previously validated in the existing literature [10] and are sufficient to demonstrate the feasibility of our scheme, we did not perform testing with other parameter values.



```java
// GTOTP password verify algorithm
public static  int Verify(String[][] pwGTOTP,long T) throws NoSuchAlgorithmException {
    int verifyResult = 0;
    //Compute the index j of the TOTP password contained in the i-th hash chain of the GTOTP password pwGTOTP
    int i =(int) ((T-Δstart)/Δe);
    int j = (int) ((T-i*Δe-Δstart)/Δs);
    //pwGTOTP[0][0] stores the TOTP password   String -> byte[]
    byte[] pwTOTP = toBytes(pwGTOTP[0][0]);
    // Compute the verify point of the i-th hash chain based on the TOTP password pwTOTP
    for(int k=0;k<j+1;k++) {
        pwTOTP = Sha256(pwTOTP);
    }
    String verifyPoint = byteToStr(pwTOTP);
    // Get undated verify point  updatedVerifyPoint = H(verifyPoint || IDCipher || i)
    String updatedVerifyPoint = byteToStr(Sha256(verifyPoint+pwGTOTP[0][1]+i));
    int vp_index = RA.vp_set.indexOf(updatedVerifyPoint);
    if(vp_index== -1) return 0;
    //Verify the TOTP password contained in the pwGTOTP
    if(TOTP.Verify(verifyPoint, pwGTOTP[0][0], j)==1 ) {
        //Verify Merkle root
        if(RA.gpk.contains(MerkleTrees.Verify(pwGTOTP[1], updatedVerifyPoint,  index: vp_index%(M*V/z))))
            verifyResult=1;
        else return 0;
    }
    return verifyResult;
}
```

Figure 10. Code of the GTOTP password verification algorithm.

We used the Solidity programming language to write and compile the smart contracts within the Remix integrated development environment, and deployed the smart contracts onto the Goerli test network for evaluation. Considering that the interaction time between the entities and the smart contract is contingent upon the computational power of the current blockchain network and the associated gas price, we gauged performance by calculating the gas fees incurred during interactions between the entities and the smart contract.

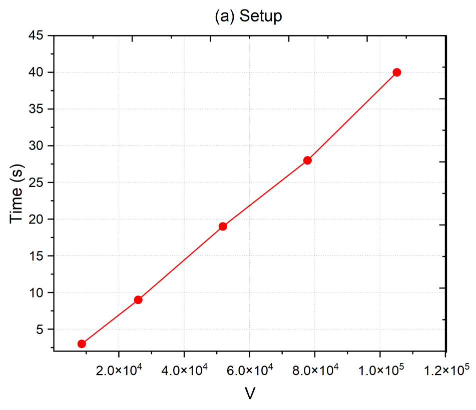
7.1. Performance of PPPoL Scheme

Setup Time. During the initialization phase, the overhead consists of the RA running the GTOTP-MT.GVKGen and GTOTP-MT.RASetup algorithms and each group member ID_u running its initialization algorithm GTOTP-MT.GMInit.

The RA performs the $M \cdot V$ encryption algorithm of ASE, $2M \cdot V$ hash calculations, and z interpolation operations of the Bloom filter. Each group member ID_u performs $V \cdot K$ hash calculations and V pseudo-random function (PRF).

Figure 11a shows the overhead of the initialization operations. It can be seen that the overhead grows linearly with V, while we usually only set V smaller (e.g., a PPPoL scheme with a five-day lifetime) with milliseconds of overhead.

Proof Size. Each hash value takes 32 bytes, a timestamp takes 8 bytes, each location cipher C takes 64 bytes, and an ephemeral public key takes 64 bytes, for a total proof size of $32(X+1) + 2(X+1) * C + 8 + 64 = 160X + 232$ bytes. Figure 11b shows that the proof size grows linearly with X when fixing V. It can be seen that the proof size is less than 3 KB even when $X = 16$, which is practical.



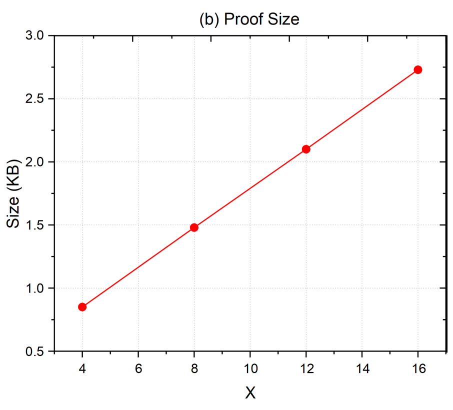


Figure 11. Runtime of Setup and proof size.

Proof Generation Time. During the location proof generation phase, the prover and witnesses run the secret seed generation algorithm GTOTP-MT.GetSd once, the password generation algorithm GTOTP-MT.PdGen twice, and the ElGamal encryption algorithm Enc once. In addition, the prover runs the verification algorithm GTOTP-MT.Verify X times, while each witness only needs to run it once.

The prover performs two exponential and two multiplicative calculations, one PRF, X queries of the Bloom filter, and $2K + X \cdot K' + X \cdot h + X + 1$ hash calculations. Each witness performs two exponential and two multiplicative calculations, one PRF operation, and $2K + K' + h + 2$ hash calculations. In the average case, it takes $K/2$ hash calculations ($K' = K/2$) to compute the verify point based on the GTOTP password, while in the worst case it takes K hash calculations ($K' = K$).

Figure 12a illustrates the linear growth of the location proof generation time with the number of witnesses X and that it varies inversely with the password generation interval Δ_{pw}. However, even when setting $\Delta_{pw} = 1$ s, the overall proof generation overhead is only a few tens of milliseconds. This feature enables our PPPoL scheme to support instant location verification.

Proof Verification Time. During the verification phase, the verifier runs the verification algorithm GTOTP-MT.Verify $X + 1$ times, the hash calculations X times, and the ElGamal decryption algorithm Dec $X + 1$ times.

The verifier performs $2X$ exponential calculations, $3X$ multiplicative calculations, and $X \cdot K' + X \cdot h + 2X + K' + h + 2$ hash calculations.

Figure 12b shows the linear growth of the location proof verification time with the number of witnesses X and that it varies inversely with password generation interval Δ_{pw}. The experiments indicate that, when setting Δ_{pw}, verifiers can verify location proofs after the expiration of the GTOTP passwords, with the verification overhead taking only tens of milliseconds. This ensures instant location verification. Moreover, Δ_{pw} can be configured based on the delay requirements for location verification.

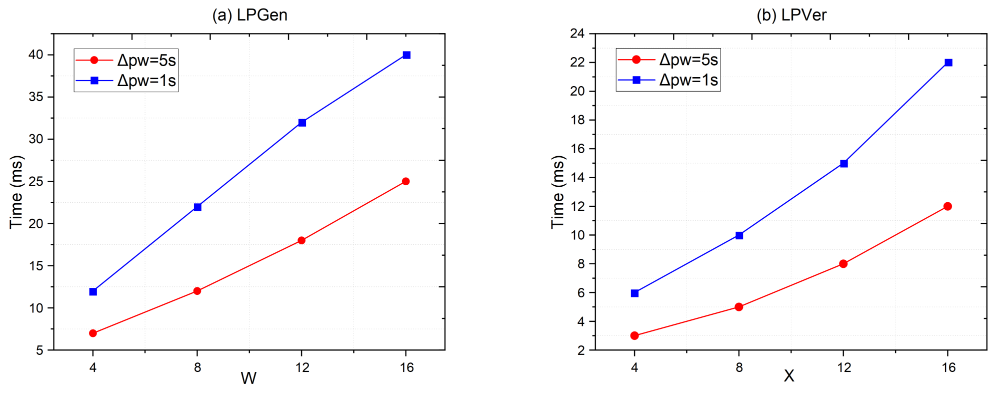


Figure 12. Runtime of LPGen and LPVer.

Communication Overhead. Communication overhead is determined by the number of group members M and of TOTP instances V. When setting $M = 20$ and V = 105,120, the communication overhead was only about 3 KB. Current communication techniques (such as 5G) are so fast that the communication overhead is on the order of milliseconds.

7.2. Performance of Incentive Mechanism

In this paper, the feasibility of this incentive mechanism is assessed through gas fees. The incentive mechanism has the following gas fee overhead: the Setup phase. in which the prover deploys a smart contract into the blockchain; the Interaction phase, in which the contract verifies the RA's identity and transfers deposits to the RA; and the Reward phase, in which rewards are transferred to the honest witnesses on behalf of the RA. We assume that the entire gas fees for the incentive mechanism are paid by the prover; thus, the deposits transferred to the RA include the RA's subsequent gas fee overhead in the Reward phase. Referring to the number of witnesses X used in previous PoL schemes [10,33], we ultimately set X to 4, 8, and 12, respectively, to evaluate the impact on the gas fee overhead of the incentive mechanism. While X can be set to other values, the conclusions from the gas fee overhead analysis remain consistent. As depicted in Table 3, the gas fees during the Setup and Interaction phases are independent of the number of witnesses, whereas the gas fees during the Reward phase increase linearly with the number of witnesses. When the number of witnesses equals 12, the total overhead of the incentive mechanism is about ETH 0.0011, which shows the feasibility of our incentive mechanism.

Table 3. Performance of the incentive mechanism.

Witness	Cost of 10^{-3} ETH			
	Setup	Interaction	Reward	Total
4	0.59	0.07	0.21	0.87
8	0.59	0.07	0.31	0.97
12	0.59	0.07	0.47	1.13

7.3. Comparison

For simplicity and fairness, we instantiated the public key encryption used by the compared schemes with [32] and the digital signature and group signature schemes used by the compared schemes with [34,35], respectively. We denote the exponentiation operation by EXP, the elliptic curve point multiplication operation by ECM, and the pairing of bilinear groups by BGP, as shown in Table 4. As many prior schemes have not been practically implemented, we only compared our PPPoL scheme with related PoL schemes in terms of the security properties, which include entity privacy (EntPri), location privacy (LocPri), and entity traceability (EnTra), as shown in Table 5, as well as the existence of incentives for witnesses, which we denote as Incent. We use "✓" to indicate that the corresponding property is achieved and "×" to indicate the contrary. We additionally compared the main computational overheads of the related works in Table 6.

Table 4. Abbreviation notations for compared PoL schemes.

Notation	Description
P, W, V	Identities of prover, witness and verifier, respectively.
EntPri	Entity privacy.
EnTra	Entity traceability.
LocPri	Location privacy.
Incent	Incentive mechanism.
EXP	Exponentiation operation.
ECM	Elliptic curve point multiplication operation.
BGP	Pairing of bilinear groups.

Table 5. Comparison of security attributes.

PoL Schemes	EnTra		EntPri		LocPri		Incent
	P	W	P	W	P	W	W
[18]	×	×	✓	×	✓	×	×
[36]	✓	✓	×	×	×	×	×
[17]	×	×	✓	✓	✓	×	×
[37]	✓	✓	✓	✓	×	✓	×
[10]	✓	✓	✓	✓	×	✓	×
Our	✓	✓	✓	✓	×	✓	✓

Table 6. Comparison of performance.

PoL Schemes	LPGen		LPVer
	P	W	V
[18]	28ECM	56ECM	5ECM + 11BGP
[36]	ECM + 3EXP	ECM + 3EXP	12ECM + 18EXP
[17]	8ECM	ECM + 3EXP	10ECM + 15EXP
[37]	252ECM + 34EXP	42ECM + 2EXP	126ECM + 12EXP + 24BGP
[10]	15ECM	12ECM	5ECM
Our	2EXP	2EXP	10EXP

From Table 5, it can be seen that only [10,37] and our PPPoL consider entity privacy and traceability. Wu et al. [18] used the zero-knowledge proof scheme to achieve location privacy of the prover with respect to the verifier. However, in real applications the prover might allow the verifier to obtain its location, such as in contact tracing, where the verifier enacts measures to mitigate the spread of infectious diseases using the location of the infected person. The PoL scheme of Zhu et al. [17] combines mathematical methods to achieve entity privacy and location privacy; however, an adversary may have a non-negligible probability of breaking both of the above security properties. On the other hand, we use provably secure cryptographic building blocks, i.e., our PPPoL scheme is more suitable in PoL application scenarios where participants need privacy protection. The PoL scheme of Dupin et al. [37] achieves similar security properties as ours; however, the building blocks are more expensive than our PPPoL scheme, and their scheme may not be applicable to resource-constrained devices. The PoL scheme of Yang et al. [10] achieves similar security properties to our scheme; however, their scheme requires each witness to generate their own set of random proximity distances, resulting in larger proof sizes and increased performance overhead for both witnesses and prover. Furthermore, we have designed an incentive mechanism to promote the participation of witnesses in the PPPoL scheme, which has not been considered in other schemes. This incentive mechanism can promote the application of PoL schemes in wider domains, e.g., LBS and IoT. Furthermore, our PPPoL scheme supports real-time location verification, which is suitable for PoL applications that require low latency. For example, a bank may require location proofs that can be verified in real time in order to validate a user's transaction.

8. Conclusions

In this paper, we have proposed an efficient privacy-preserving proof of location scheme (PPPoL) based on GTOTP, achieving entity privacy, location privacy, and traceability of entity identity for tracing identity when entities perform the protocol dishonestly. In order to encourage the proactive participation of witnesses in providing location proofs, we designed an incentive mechanism based on smart contracts. We implemented a proof-of-concept (PoC) for PPPoL on an Android device and show the theoretical and practical overhead of the scheme.

The proposed PPPoL scheme can achieve instant location verification, which only requires setting the GTOTP password generation interval (e.g., one second) and performing additional hash calculations. Thus, the verifier can obtain the commitment keys (GTOTP passwords) of the prover and the related witnesses much sooner, complete the location verification, and send the verification result to the prover in an instant, which is necessary for certain application scenarios that require timely location verification.

Regarding future work, we plan to implement dynamic management of witnesses to permit revocation of the witnesses' identity if they behave dishonestly. In addition, we plan to design more detailed incentives and penalties in the incentive mechanism to better motivate witnesses to participate in the PPPoL scheme, as well as to address the issue of cryptocurrency volatility involved in the overhead of the incentive mechanism.

Author Contributions: Conceptualization, C.M.; methodology, C.M.; software, C.M.; validation, C.M. and Z.Y.; investigation, C.M., Y.L., and J.M.; writing—original draft preparation, C.M.; writing—review and editing, C.M., Y.L, Z.Y., and J.M.; supervision, Z.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Natural Science Foundation of China (Grant No. 62372386) and the Natural Science Foundation of Chongqing (Grant No. CSTB2022NSCQ-MSX0437).

Data Availability Statement: All data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Huang, H.; Gartner, G.; Krisp, J.M.; Raubal, M.; de Weghe, N.V. Location based services: ongoing evolution and research agenda. *J. Locat. Based Serv.* **2018**, *12*, 63–93. [CrossRef]
2. Junglas, I.A.; Watson, R.T. Location-based services. *Commun. ACM* **2008**, *51*, 65–69. [CrossRef]
3. Jiang, D.; Guo, X.; Gao, Y.; Liu, J.; Li, H.; Cheng, J. Locations recommendation based on check-in data from Location-Based Social Network. In Proceedings of the 2014 22nd International Conference on Geoinformatics, Kaohsiung, Taiwan, 25–27 June 2014; pp. 1–4.
4. Hakobyan, L.; Lumsden, J.; O'Sullivan, D.; Bartlett, H. Mobile assistive technologies for the visually impaired. *Surv. Ophthalmol.* **2013**, *58*, 513–528. [CrossRef] [PubMed]
5. Zafar, F.; Khan, A.; Anjum, A.; Maple, C.; Shah, M.A. Location Proof Systems for Smart Internet of Things: Requirements, Taxonomy, and Comparative Analysis. *Electronics* **2020**, *9*, 1776. [CrossRef]
6. Yan, S.; Malaney, R.A.; Nevat, I.; Peters, G.W. Location Verification Systems for VANETs in Rician Fading Channels. *IEEE Trans. Veh. Technol.* **2014**, *65*, 5652–5664. [CrossRef]
7. Koh, J.Y.; Nevat, I.; Leong, D.; Wong, L.W.C. Geo-Spatial Location Spoofing Detection for Internet of Things. *IEEE Internet Things J.* **2016**, *3*, 971–978. [CrossRef]
8. Ferretti, L.; Wymant, C.; Kendall, M.; Zhao, L.; Nurtay, A.; Abeler-Dörner, L.; Parker, M.J.; Bonsall, D.; Fraser, C. Quantifying SARS-CoV-2 transmission suggests epidemic control with digital contact tracing. *Science* **2020**, *368*, eabb6936. [CrossRef]
9. Bellovin, S.M.; Hutchins, R.M.; Jebara, T.; Zimmeck, S. When Enough is Enough: Location Tracking, Mosaic Theory, and Machine Learning. *N. Y. Univ. J. Law Lib.* **2013**, *8*, 556. [CrossRef]
10. Yang, Z.; Jin, C.; Ning, J.; Li, Z.; Dinh, A.; Zhou, J. Group Time-based One-time Passwords and its Application to Efficient Privacy-Preserving Proof of Location. In Proceedings of the Annual Computer Security Applications Conference, Virtual, 6–10 December 2021.
11. Hamari, J.; Sjöklint, M.; Ukkonen, A. The sharing economy: Why people participate in collaborative consumption. *J. Assoc. Inf. Sci. Technol.* **2016**, *67*, 2047–2059. [CrossRef]
12. Lamport, L. Constructing Digital Signatures from a One Way Function. In Proceedings of the HICSS-43, Koloa, HI, USA, 5–8 January 2010.
13. Kogan, D.; Manohar, N.; Boneh, D. T/Key: Second-Factor Authentication From Secure Hash Chains. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017.
14. Li, Y.; Li, X.; Zhong, L.; Jing, Y. Research on the S/KEY one-time password authentication system and its application in banking and financial systems. In Proceedings of the 6th International Conference on Networked Computing and Advanced Information Management, Seoul, Republic of Korea, 16–18 August 2010; pp. 172–175.
15. Javali, C.; Revadigar, G.; Rasmussen, K.B.; Hu, W.; Jha, S. I Am Alice, I Was in Wonderland: Secure Location Proof Generation and Verification Protocol. In Proceedings of the 2016 IEEE 41st Conference on Local Computer Networks (LCN), Dubai, United Arab Emirates, 7–10 November 2016; pp. 477–485.
16. Li, Y.; Zhou, L.; Zhu, H.; Sun, L. Privacy-Preserving Location Proof for Securing Large-Scale Database-Driven Cognitive Radio Networks. *IEEE Internet Things J.* **2016**, *3*, 563–571. [CrossRef]
17. Zhu, Z.; Cao, G. Toward Privacy Preserving and Collusion Resistance in a Location Proof Updating System. *IEEE Trans. Mob. Comput.* **2013**, *12*, 51–64. [CrossRef]
18. Wu, W.; Liu, E.; Gong, X.; Wang, R. Blockchain Based Zero-Knowledge Proof of Location in IoT. In Proceedings of the ICC 2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; pp. 1–7.
19. Järvinen, K.; Kiss, Á.; Schneider, T.; Tkachenko, O.; Yang, Z. Faster Privacy-Preserving Location Proximity Schemes. *IACR Cryptol. ePrint Arch.* **2018**, *2018*, 694.
20. Zou, W.; Lo, D.; Kochhar, P.S.; Le, X.B.D.; Xia, X.; Feng, Y.; Chen, Z.; Xu, B. Smart Contract Development: Challenges and Opportunities. *IEEE Trans. Softw. Eng.* **2021**, *47*, 2084–2106. [CrossRef]

21. Saha, R.; Kumar, G.; Conti, M.; Devgun, T.; Kim, T.-h.; Alazab, M.; Thomas, R. DHACS: Smart Contract-Based Decentralized Hybrid Access Control for Industrial Internet-of-Things. *IEEE Trans. Ind. Inform.* **2021**, *18*, 3452–3461. [CrossRef]
22. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 557–564.
23. Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*; USSC: Washington, DC, USA, 2009.
24. Gueron, S.; Langley, A.; Lindell, Y. AES-GCM-SIV: Specification and Analysis. *IACR Cryptol. ePrint Arch.* **2017**, *2017*, 168.
25. Puniya, P. New Design Criteria for Hash Functions and Block Ciphers. Ph.D. Thesis, New York University, New York, NY, USA, 2007.
26. Liu, H.; Luo, X.; Liu, H.; Xia, X. Merkle Tree: A Fundamental Component of Blockchains. In Proceedings of the 2021 International Conference on Electronic Information Engineering and Computer Science (EIECS), Changchun, China, 23–26 September 2021; pp. 556–561.
27. Fumagalli, G.; Raimondi, D.; Giancarlo, R.; Malchiodi, D.; Frasca, M. On the Choice of General Purpose Classifiers in Learned Bloom Filters: An Initial Analysis Within Basic Filters. In Proceedings of the International Conference on Pattern Recognition Applications and Methods, Online, 4–6 February 2021.
28. Liu, P. Public-Key Encryption Secure Against Related Randomness Attacks for Improved End-to-End Security of Cloud/Edge Computing. *IEEE Access* **2020**, *8*, 16750–16759. [CrossRef]
29. Adeniyi, E.A.; Falola, P.B.; Maashi, M.S.; Aljebreen, M.; Bharany, S. Secure Sensitive Data Sharing Using RSA and ElGamal Cryptographic Algorithms with Hash Functions. *Information* **2022**, *13*, 442. [CrossRef]
30. Rescorla, E. The Transport Layer Security (TLS) Protocol Version 1.3. *RFC* **2018**, *8446*, 1–160.
31. Gomez, C.; Oller, J.; Aspas, J.P. Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. *Sensors* **2012**, *12*, 11734–11753. [CrossRef]
32. Mohammed, S.J.; Taha, D.B. Performance Evaluation of RSA, ElGamal, and Paillier Partial Homomorphic Encryption Algorithms. In Proceedings of the 2022 International Conference on Computer Science and Software Engineering (CSASE), Duhok, Iraq, 15–17 March 2022; pp. 89–94.
33. Akand, M.M.R.; Safavi-Naini, R.; Kneppers, M.; Giraud, M.; Lafourcade, P. Privacy-Preserving Proof-of-Location with Security Against Geo-Tampering. *IEEE Trans. Dependable Secur. Comput.* **2023**, *20*, 131–146. [CrossRef]
34. Johnson, D.B.; Menezes, A.; Vanstone, S.A. The Elliptic Curve Digital Signature Algorithm (ECDSA). *Int. J. Inf. Secur.* **2001**, *1*, 36–63. [CrossRef]
35. Emura, K.; Hayashi, T.; Ishida, A. Group Signatures with Time-Bound Keys Revisited: A New Model, an Efficient Construction, and its Implementation. *IEEE Trans. Dependable Secur. Comput.* **2020**, *17*, 292–305. [CrossRef]
36. Nosouhi, M.R.; Sood, K.; Yu, S.; Grobler, M.; Zhang, J. PASPORT: A Secure and Private Location Proof Generation and Verification Framework. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 293–307. [CrossRef]
37. Dupin, A.; Robert, J.M.; Bidan, C. Location-Proof System based on Secure Multi-Party Computations. In Proceedings of the International Conference on Provable Security, Jeju, Republic of Korea, 25–28 October 2018.