Journal of Computer Science

Original Research Paper

Anonymous and Distributed Authentication for Peer-to-Peer Networks

[1]Pasan Tennakoon, [1]Supipi Karunathilaka, [1]Rishikeshan Lavakumar and [2,3]Janaka Alawatugoda

[1]Department of Computer Engineering, University of Peradeniya, Sri Lanka
[2]Research and Innovation Centers Division, Faculty of Resilience, Rabdan Academy, United Arab Emirates
[3]Institute for Integrated and Intelligent Systems, Griffith University, Australia

Article history
Received: 07-09-2022
Revised: 22-11-2022
Accepted: 07-12-2022

Corresponding Author:
Janaka Alawatugoda
Research and Innovation Centers Division, Faculty of Resilience, Rabdan Academy, United Arab Emirates
Email: jalawatugoda@ra.ac.ae

Abstract: Well-known authentication mechanisms such as Public-key Infrastructure (PKI) and Identity-based Public-key Certificates (ID-PKC) are not suitable for integration in a Peer-to-Peer (P2P) network environment, the reason being either the lack of or the difficulty in maintaining a centralized authority to manage the certificates. Authentication becomes even harder in anonymous environments. In this study, we present three authentication protocols such that the users can authenticate themselves in an anonymous P2P network, without revealing their identities. The first protocol uses existing ring signature schemes to obtain anonymous authentication, the second is an anonymous authentication protocol utilizing secret sharing schemes, and lastly a zero-knowledge-based anonymous authentication protocol. We provide security justifications for the three aforementioned protocols in terms of anonymity, completeness, soundness, resilience to impersonation attacks, and resilience to replay attacks. We also provide examples of conceptual topologies and how the peers would behave and rearrange in case of failure.

Keywords: Anonymous Authentication, Peer-to-Peer Networks, Ring Signatures, Secret Sharing, Zero-Knowledge, Blockchains

Introduction

The concept of Peer-to-Peer (P2P) communication has gained significant attention in the networking community over the years. Since the release of Napster in 1998, many P2P applications and mechanisms have been introduced. BitTorrent, Bitcoin Nakamoto (2009), and TOR (Dingledine *et al*., 2004) are some of the more popular P2P protocols and applications. The absence of a centralized authority and censorship is the main reason behind the popularity of P2P applications. This eliminates the need for an expensive central service as well as removes the vulnerability of a single point of failure. The P2P networks are considered to be more efficient and scalable than traditional client-server applications.

The decentralized nature of the P2P networks makes it inappropriate to integrate with traditional authentication mechanisms such as Public-Key Infrastructure (PKI) and Identity-based Public-key Certificates (ID-PKC). The reason is the difficulty in maintaining a centralized authority to manage and attest the certificates or the lack thereof in the protocol. Therefore, many such networks focus on providing user anonymity rather than peer authentication. The reduced security of these networks makes the networks more vulnerable to attacks which are typically nonexistent in a centralized system (Wallach, 2002). The anonymity features of these networks have created a safe house for malicious and illegal behavior (Jardine, 2015). Being unaccountable for their actions, P2P users have had the freedom to behave maliciously and therefore some of them have exercised it, without recourse. This can cause harm to the network as well as its users. Accountability can be achieved through authentication. To integrate an authentication mechanism into an anonymous P2P environment, we need to solve two of the main challenges:

1. Authentication in a decentralized environment and
2. Authentication without revealing the identity

The above two challenges have been discussed since the dawn of the Internet: Authentication needs to address the issues such as the absence of a central server, certificate management in a distributed environment, the semi-trusted nature of peers, and the unpredictable availability of the peers. Moreover, authentication needs to hide the authenticating party's identity, and be secure against misbehaving peers (malicious verifiers and provers). In this study, we present multiple approaches to solve the problems encountered when malicious users are present in P2P networks.

We present three approaches for anonymous authentication in P2P networks to solve the aforementioned challenges:

1. Ring signature approach
2. Authenticated secret-sharing approach and
3. Zero-knowledge proof approach

We provide security justifications for the three protocols in terms of anonymity, completeness, soundness, resilience to impersonation attacks, and resilience to replay attacks. Thus, our contribution through this study is to notify the identified challenges arising when integrating an authentication mechanism into an anonymous P2P environment and propose three justifiable approaches for anonymous authentication in P2P. We also present, along with the protocol, a conceptual network design in which it is optimal.

Authentication in P2P

The absence of a central service makes authentication in P2P networks complex: Existing mechanisms like PKI or ID-PKC are based on trusted third parties. Establishing a trusted third party in a semi-trusted network like P2P is a difficult undertaking. Many P2P networks propose trust and reputation management schemes to solve this problem. Some works (Gokhale and Dasgupta, 2003; Wang *et al*., 2010; Tsang and Smith, 2008) use trust and reputation schemes to discover peers that can be considered as trusted peers of the network. These trusted peers are used in authentication as trusted third parties. The idea of reputation management systems is to evaluate a peer's trustworthiness based on its interactions with other peers (Kamvar *et al*., 2003; Lee *et al*., 2003; Sabater and Sierra, 2002; Xiong and Liu, 2003). P2P systems that use reputation management schemes to assist in authentication suffer from a trivial flaw; these schemes assume that the reputation system is intelligent enough not to select malicious users as trusted peers. Trusting malicious peers to protect sensitive information can harm the system.

Some researchers suggest using a modified PKI for authentication in P2P networks (Oh *et al*., 2008; Josephson *et al*., 2004). Rather than having a single centralized authority, its responsibility is distributed across multiple peers in the network. This improves the scalability, trustworthiness, and robustness of the authentication process. The downside of using modified PKI in P2P is that certificate management becomes complex. As a solution, the one proposed by Josephson *et al*. (2004) uses a set of peers as Authentication Servers (AS). Even though it improves the scalability of the network, it introduces new security risks such as unreliability in certificate access and verification.

To solve the problem of the absence of a centralized authority and at the same time to make the authentication process reliable, some modern authentication schemes utilize blockchains (Papageorgiou *et al*., 2020; Karaarslan and Adiguzel, 2018; Yakubov *et al*., 2018; Orman, 2018). Blockchain can make the process of a CA in a distributed, immutable and transparent manner. Therefore, can successfully solve the problems of malicious CAs, MITM attacks, and single points of failure. Blockchain is used as a distributed key-value data storage. The data is public and readable to everyone. Sivakumar and Singh (2017) proposed the idea of using smart contracts for certificate management. The decentralized PKI is secure as long as honest nodes control collectively more than 50% of the computational power. Moreover, the need for blockchain to decentralize PKI has been argued multiple times (Alilwit, 2020; Asif *et al*., 2022; Umoren *et al*., 2022), since the technology of blockchain was introduced to the industry, although it's comparatively newer.

PGP’s Web of Trust (WoT) (Bob *et al*., 2005) is another way to navigate the problem of not having a trusted central authority. WoT distributes the responsibility of CAs among users. The core concept of WoT is trust chains. For a simpler explanation, if we assume A wants to authenticate themself to B and there is a user C who is trusted by B then C can sign A’s certificate after verifying its authenticity. Then A can send the signed certificate to B. Since C has signed A’s certificate and B trusts C, B can trust that A’s certificate is authentic. Using indirect trust chains, WoT creates a community of trusted users. However, WoT is not suitable for anonymous P2P networks, because it is difficult for a new peer to join the network without personally knowing an existing user of the network and getting the identity attested.

Anonymous Authentication in P2P

The concept of anonymous authentication has been around for over a decade. Pseudo Trust (PT) by Lu *et al*. (2007) has been one of the more popular works on this topic. PT utilizes the concept of double pseudonyms combined with zero knowledge proofs to authenticate users anonymously. PT also uses onion routing (Dingledine *et al*., 2004) and Eigen Trust (Kamvar *et al*., 2003) trust management to provide a complete file delivery system with anonymous authentication. The anonymity comes from the one-way property of the cryptographic hash functions. However, the PT neglects one important feature of using the concept of pseudonyms to obtain anonymity: PT does not change the Pseudo

Identity (PI) before each authentication process. The PT protocol requires the certificate of Pseudo-Identity (PIC) to be sent to the other party to start the authentication. Since the PIC is the same for a particular user, an eavesdropper can link two communication sessions to the particular user. Han *et al.* (2020) presents a similar authentication scheme to the PT for Internet of Vehicles (IoV) and that also suffers from the same vulnerabilities as the one in the PT.

Tsang and Smith (2008) present an interesting approach to anonymous authentication; P2P Anonymous Authentication (PPAA) uses tags to obtain anonymity and at the same time link the communication sessions. The idea is to use the IDs of the two parties involved in the communication session to create a tag. The two parties will not learn any information except that the tag is from the execution of the protocol. To avoid having the same tag for different communication sessions between the same parties, the PPAA includes an event id in the tag. Therefore, a party which previously involved in the communication will be able to link a communication session to a previous session with the same party. The PPAA is proven to be secure in the Random Oracle Model (ROM).

Wang *et al.* (2010) present Collaboration Signature Trust (CST) to authenticate users anonymously. However, this mechanism is not safe in a semi-trusted environment such as a P2P network. Another one by Wang and Sun (2009) presents a similar method to the CST; they use Fair Blind Signature Trust (FBST) (Stadler *et al.* 1995) to present a novel authentication scheme that keeps the anonymity of honest users. Similar to the CST, this uses a trust management system called SOBIE to elect peers as Super Peers (SPs) and Reputed Peers (RPs). They are assumed to be trustworthy and play an important role in the authentication. However, as mentioned earlier, trust management systems are not perfect. Malicious peers can get elected as SPs or RPs and they are capable of revoking the users' anonymity. Similar to the CST, Wang and Sun (2009) use the concept of secret sharing (Shamir, 1979) to reduce the vulnerability of exposed RPs. Shamir presents a way to break a key into several parts and store it in multiple places and then recreate the key when required. Wang and Sun (2009) technique use this to break the key (the link between ID and pseudo-ID) and store it among multiple RPs. Therefore, even if a few RPs get compromised it does not reveal the user's identity. Further, a user uses an anonymous multi-cast to communicate with an SP. This makes it impossible for an SP to reveal the identity of a user.

Materials and Methods

Now we briefly recall the cryptographic preliminaries that we have used for our work.

Ring Signatures

The notion of ring signatures was first introduced by Rivest *et al.* (2001). Ring signatures are used to digitally sign messages on behalf of a group, in a way that it is computationally hard to find the exact signer. The ring signatures are designed to provide unconditional anonymity to the message signer and the ring signatures do not depend on a third party to generate a signature. Over the years different ring signature schemes have been published with different features: Threshold ring signatures by Bresson *et al.* (2002), linkable ring signatures by Liu and Wong (2005), revocable ring signatures by Liu *et al.* (2007), etc.

Let there be a group of k number of entities where each entity $i \in \{1,...,k\}$ has a public key P_i and a corresponding secret key S_i. An entity $r \in \{1,..., k\}$ (with the public key P_r and the corresponding secret key S_r) can generate a ring signature on a message m using (m, P_1,..., P_k, S_r). Anyone with knowledge of m, P_1,..., and P_k can verify the ring signature. No one outside the group (without a secret key S_i) can generate a valid ring signature for the same group.

Secret Sharing Schemes

In 1979, Shamir introduced the concept of secret sharing. This allows a secret to be divided into n parts. The secret can be reconstructed with at least t parts ($1 \leq t \leq n$). No knowledge about the secret can be learned with (t-1) parts.

The concept is based on polynomial interpolation. The idea is to generate a polynomial $f(x)$ of (t-1) points. First, we select (t-1) random positive integers such that (a_1, a_2,...,a_{t-1}). Then, set a_0 to the secret we want to share. These points are used to generate the polynomial $f(x)$:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}$$

Then, we get n points (x_i, y_i) corresponding to the polynomial. Given any subset of t points a_0 can be found by Lagrange basis interpolation:

$$\ell_i = \frac{x - x_0}{x_i - x_0} \times \frac{x - x_1}{x_i - x_1} \times \cdots \times \frac{x - x_{t-1}}{x_i - x_{t-1}}$$

$$f(x) = \sum_{i=0}^{t-1} y_i \ell_i(x)$$

The idea of Shamir's secret sharing (Shamir, 1979) is a popular concept in P2P systems. A P2P network does not have a centralized database to store peers' keys. Storing keys in a selected set of peers might not be a good idea since this P2P environment is semi-trusted. For example, when a peer requests a key from another peer, they may not get a response. Therefore, keys need to be broken into parts and distributed among multiple peers and a peer should be able to reconstruct a key without the knowledge of all the distributed parts. There are P2P anonymous authentication mechanisms that use the concept of secret sharing: Wang *et al.* (2010); Wang and Sun (2009).

Zero Knowledge Proofs

A Zero-Knowledge Protocol (ZKP) allows a prover to prove the possession of some secret to a verifier without revealing the secret or any information related to the secret. The idea of a ZKP was first introduced by Goldwasser *et al*. (2019). Since then, many different ZKPs have been presented (Tang *et al*., 2003; Feige *et al*., 1988; Cramer and Damgard, 1997; Sahai and Vadhan, 2000). A ZKP must satisfy soundness, completeness, and zero-knowledge properties. There are two types of ZKP systems; interactive zero-knowledge proofs and non-interactive zero-knowledge proofs stated by Wu and Wang (2014).

Network Design

In this section, we detail the network design, the conceptual design, and the distributed certificate management regarding our work.

Conceptual Design

We employ a hybrid P2P network (Beverly Yang and Garcia-Molina, 2003). A traditional hybrid P2P network consists of peers and super peers. Hybrid P2P systems are a combination of purely distributed P2P systems and mediated P2P systems. The hybrid systems are designed to overcome the problems of the two aforementioned systems. These systems provide search efficiency of mediated P2P systems while maintaining the reliability of decentralization similar to pure P2P systems (Backx *et al*., 2002).

Our P2P network consists of three types of entities; the main server, the ordinary peers (hereafter mentioned as peers), and the super peers. A peer communicates with the main server only at the time of registration. Users join the network as peers. Peers are ordinary service requesters. They are connected to the system through their super peers. Every peer is assumed to be behind a Network Address Translation (NAT) environment. Peers with public IP addresses and higher computational power are promoted to the super peer status.

Super peers have more responsibility for the system. A super-peer is connected to one or more other super peers in the network and responsible for one or more peers. They can communicate with other super peers using the super-peer network. Super peers can join or leave the network at any time. The dynamic behavior of super peers should not affect the connectivity of the network. Our design of the network is capable of changing the topology according to this dynamic behavior of peers and maintaining connectivity among the existing super peers. A super-peer is the only responsible entity for the nodes under its scope and does not know any information regarding the other peers of the system. Therefore, the node discovery process becomes an exhaustive task. This can be accomplished in two ways: A flooding search or a random walk. We utilize flooding search in this project since the random walk method is not guaranteed to produce the results (Ahmed and Boutaba, 2011).

Distributed Certificate Management

The decentralized nature of the P2P networks makes it difficult to integrate traditional authentication mechanisms into them. Distributing certificates among super peers is not a viable solution since the super peers are not always available; at times all the certificates under a particular super peer may not be accessible. Moreover, malicious super peers might delete certificates from the network. We propose a different solution using the secret sharing scheme of (Shamir, 1979).

During the initial interaction with a peer, the corresponding super peer obtains the peer's certificate. The super peer breaks the certificate into n parts using Shamir's algorithm. The super peer then floods the parts across the network. Once a certificate recreation request is received, the super peer again floods the request across the network to collect the parts of the certificate. The super peers that are holding the parts of the certificate will send them to the corresponding super peers. The original certificate can be recreated as long as r parts are received by the super peer ($r \leq n$).

This technique allows for dynamically distributing certificates. As long as r super peers can be accessed, the certificate can be recreated. This method only requires minimal storage; the size of a single part does not exceed the size of the original certificate. This is also the more flexible approach. The parameters n and r can be changed for each certificate without affecting the other certificates. However, then it needs a way to identify n and r for each certificate. Increasing n while keeping r constant will increase the average key storage size in the super peers.

Authentication Schemes

In this section, we discuss the details of the authentication schemes we propose.

Ring Signature Approach

Ring signatures allow a message to be signed by a group of public keys while making it impossible to identify the exact signer. The ring signatures provide complete anonymity. However, ring signatures are not suitable for authentication, and because of that, it is impossible to revoke the anonymity of malicious peers. Therefore, we use the revocable ring signature scheme of (Liu *et al*., 2007), to create a simple authentication protocol that protects the users' privacy. The underlying idea is to challenge the prover to generate a ring signature using a random nonce generated by a verifier. If the prover can accomplish this task, it can successfully authenticate itself. The protocol is explained below.

Registration

1. A user has an *ID* which can be anything related to the identity of the user. The user picks a random number r_u and generates the private key S_u using a hash function H_1 such that $S_u = H_1 (ID, r_u)$. Then, the user generates the public key P_u corresponding to the S_u. After that, the user sends the registration request along with his *ID* and P_u to the main server
2. The main server verifies the identity of the user. Then, the server signs P_u with his private key S_s of the main server to generate user certificate $Cert_u$ and sends $Cert_u$ to the user

Authentication

1. The prover collects k number of certificates from the super peer. Then, randomly selects *n*-1 certificate from the set of *k* certificates. After verifying the authenticity of the selected certificates, the prover generates $CT = \{Cert_1, Cert_2,..., Cert_n\}$, which includes the prover's certificate $Cert_p$ as well (total *n* number of certificates now). Then, the prover obtains each corresponding public key from the certificates to generate $P = \{P_1, P_2,..., P_n\}$. After that, encrypts *CT* with the verifier's public key P_v and sends it to the verifier
2. The verifier decrypts the message to obtain *CT*. After verifying the authenticity of each $Cert_i$, the verifier generates each P_i using the main server's public key P_s. Then, using another hash function Hash generates $H = \text{Hash} (P_1, P_2,..., P_n)$. Then, sends *H* and a random nonce N to the prover
3. Prover generates $H' = \text{Hash} (P)$ and if $H \neq H'$ terminates the authentication. Otherwise, uses his secret keys S_p, *P*, and P_s to sign *N* and generates ring signature σ using the ring signature scheme (Liu *et al.*, 2007). Then, encrypts σ and *N* with the verifier's public key P_v and sends it to the verifier
4. The verifier decrypts the message to obtain σ and *N*. Then, verifies whether σ corresponds to *N*. If the verification is successful, the prover is successfully authenticated. Otherwise, the verifier sends a failure message

Security Justification

- Anonymity: The anonymity of the protocol depends on the properties of the ring signature scheme. The scheme proves that it obtains signer anonymity. The proposed protocol does not reveal any information other than the set of public keys *P*. The only information the verifier can deduce is that the prover's public key is among the set *P*. Therefore, our protocol obtains k-anonymity
- Completeness: If a protocol has completeness property, the protocol is said to be comprehensive; an honest verifier will always be able to authenticate themself. The completeness property of the protocol comes from the underlying ring signature scheme of (Liu *et al.*, 2007). Therefore, our protocol satisfies the completeness property
- Soundness: If a protocol has soundness property, the protocol is said to be truthful; a cheating prover will never be able to authenticate themself. Since the underlying ring signature scheme satisfies the unforgeability property, a cheating prover is unable to forge. Therefore, our protocol satisfies the soundness property
- Impersonation: Impersonation means a malicious user can impersonate another user. A protocol that accomplishes soundness and completeness is secure against impersonation attacks. Therefore, our protocol is secure against impersonation
- Replay attacks: An adversary can eavesdrop on an authentication session, save the transferring messages and resend them later to gain an advantage in authenticating themself maliciously. This is known as the reply attack. Let's assume a scenario where a malicious user M is eavesdropping on an authentication session. M can save the message Msg 1 in step 1 (of the Authentication process of the protocol) and message Msg 3 in step 3, replay them later hoping to authenticate themself maliciously
  In our protocol, Msg 1 is encrypted. Therefore, *M* will not be able to reveal its content. When Msg 1 is replayed, the verifier will respond with a random *N* and *H*. Without the knowledge of *P* or *C*, the prover will not be able to generate the correct ring signature. Therefore, they will not be able to authenticate themself. Replaying Msg 3 will not gain anything unless the verifier generates the same N as the original authentication

Authenticated Secret Sharing Approach

The basic idea of this approach is to present a prover with a set of public keys and a challenge to prove the knowledge of at least one secret key which corresponds to a public key from the set. However, the protocol should not reveal any information related to the prover's identity, and a prover without a valid key pair should not be able to authenticate themself. To accomplish this, we adopt an authenticated key exchange protocol (Alawatugoda, 2017). The protocol is explained below:

Registration

1. A user has an ID which can be anything related to the identity of the user. The user picks a random number r_u and using a hash function H_1 generates $S_u = H_1(ID, r_u)$. S_u is the private key of the user. The user then generates the public key P_u corresponding to S_u. Then, the user sends the registration request along with his *ID* and P_u to the main server

2. The main server verifies the identity of the user. Then the server signs P_u with his private key S_s to generate $Cert_u$. Then, sends $Cert_u$ to the user

Authentication

1. Prover collects k certificates from the super-peer. Then, randomly selects n-1 certificates from the set of k certificates. After verifying the authenticity of the selected certificates prover generates $CT = \{Cert_1, Cert_2, ..., Cert_n\}$, which includes the prover's certificate $Cert_p$ as well (total n number of certificates now). Then, encrypts CT using the verifier's public key P_v and sends it to the verifier
2. The verifier decrypts the message using his secret key S_v and generates H = Hash (CT) using a hash function Hash. Then, picks a random number x, and compute $X = g^x$. After that, generates n ciphertexts $\{C_1, C_2, ..., C_n\} = C$ where each C_i is encryption of (X||H) using corresponding public key P_i. Then, the verifier sends C to the prover
3. The prover selects the C_i that corresponds to his public key and decrypts it using his secret key S_p to obtain X and H. Then, generates H' = Hash (CT) and checks whether $H = H'$. If not, terminates the session. Otherwise, picks a random number y to generate $Y = g^y$. Then, computes $K = X^y$, encrypts Y using the public key of the verifier P_v, and sends it to the verifier
4. The verifier decrypts the message and obtains Y. Then, computes $K = Y^x$. Then, picks another random number R, and encrypts it using the key K (note that the encryption scheme is a symmetric-key encryption scheme). After that, generates H_1 = Hash (R||K). Then, sends the encryption of R (that is computed above) and H_1 to the prover
5. The prover decrypts the message using K and obtains R. Then, uses R and K to generate H_1'= Hash (R||K). If $H_1 = H_1'$, the prover sends R back to the verifier. Otherwise, terminates the authentication session
6. Authentication is successful if the verifier obtains the same R. Otherwise, the verifier sends a failure message to the prover

Security Justification

- Anonymity: The protocol hides the identity of the prover among a group of selected peers that is selected by the prover at random. Therefore, the verifier cannot manipulate to obtain knowledge about the prover. A cheating verifier may use different X values to obtain the prover's identity. The verifier will generate a set of $x = \{x_1, x_2, ..., x_n\}$ and generates $X = \{X_1, X_2, ..., X_n\} | X_i = g^{xi}$. Then, the verifier can generate $C = \{C_1, C_2, ..., C_n\}$. By doing so, the verifier hopes to identify which C_i prover was able to decrypt. Then, the verifier can link the C_i to the corresponding P_i to reveal the identity of the prover
  However, this will not allow the verifier to reveal the prover's identity since, at step 4 of the protocol, the verifier needs to generate K without the knowledge of the exact X that the verifier received. Therefore, the prover will not reveal any information about themself, unless the verifier can successfully guess the X_i that the prover decrypted
  Another possibility is using the above method and generating a vector of $K = \{K_1, K_2, ..., K_n\}$, where each K_i corresponds to a different X_i. Then, at step 4 of the protocol, it selects a random K_v and sends the encryption of R using K_v as the symmetric key. By this, the verifier hopes to find which K_i the prover generated. This can be done by replicating the decryption process using the elements of the K vector. Then, checks what K_i generates similar output. However, this is not possible due to the H_1 hash value, since this must include the correct key, the prover will know the malicious intentions of the verifier and terminate the authentication process
  This method does not provide k-anonymity, since the prover always terminates the authentication whenever the protocol was not correctly followed; the verifier can use this knowledge to reduce the scope of the prover's identity. For example, the verifier generates C as half of the C_i is incorrectly formed and the other half is correctly formed. If the prover terminates the authentication process, the prover's public key is one of the mis formed public keys. Otherwise, the prover's public key is one of the correctly formed public keys
- Completeness: If the prover indeed has a secret key corresponding to any one of the public keys in set P, the prover can successfully decrypt the encrypted (X||H). Therefore, can obtain the correct key K for step 5. Since the prover generates the correct key K, they can successfully decrypt the encrypted R. Therefore, can successfully authenticate themself
- Soundness: A cheating prover does not have a secret key corresponding to any of the public keys in P. To authenticate themself as a member they have to correctly guess X at step 3 of the protocol or correctly guess R at step 5 of the protocol. Both it is statistically negligible since X and R are generated using randomly picked values by the verifier for each communication session. Therefore, unless the prover can obtain a secret key and a corresponding public key from another registered user, it is not possible to authenticate themself
- Impersonation: Since the protocol accomplishes both soundness and completeness, this protocol is secure against impersonation attacks
- Replay Attacks: Let's assume a scenario where a malicious user M is eavesdropping on a communication session. The M can save the message Msg 1 in step 1 of the protocol, message Msg 3 in

step 3 of the protocol, and/or message Msg 5 in step 5 of the protocol, replay it later hoping to authenticate themself

If Msg 1 was replayed this will not gain any advantage for *M.* Since M does not know any secret key corresponding to the set *P*, they will not be able to authenticate unless by correctly guessing *X* or *R*. Storing Msg 3 will not help because, without the knowledge of *y*, *M* will not able to compute *K*. Only possibility of succeeding in a replay attack is if the verifier guesses the *R* correctly. Then, *M* can replay Msg 5 to successfully authenticate themself as a valid prover.

Zero Knowledge Proof Approach

ZKP is a popular approach to obtaining anonymous authentication in P2P networks. This technique has been utilized in many research works (Lu *et al*., 2007; Han *et al*., 2020; Tsang and Smith, 2008). These approaches rely on pseudonyms to hide the identity. We propose an authentication protocol that uses zero-knowledge proofs to hide the identity among a group of users. This is a modification of a non-interactive zero-knowledge proof protocol.

Registration

1. A user has an ID which can be anything related to the identity of the user. The user picks a random integer r_u and generates $a_u = H_1(ID, r_u)$ using a hash function H1. Using a_u as the private key of the user, the public key A_u is computed as $A_u = g^{au}$. Then, the user sends the registration request along with his *ID*, A_u to the main server
2. The main server verifies the identity of the user. Then, the server signs A_u with his private key K_s to generate $Cert_u$ and sends $Cert_u$ to the user

Authentication

1. Prover collects *k* number of certificates from the super-peer. Then, randomly selects *n*-1 certificates from the set of *k* certificates. After verifying the authenticity of the selected certificates, the prover generates $CT = \{Cert_1, Cert_2,..., Cert_{n-1}\}$. Then, the prover obtains each corresponding public key from the certificates to generate $P = \{A_1, A_2,..., A_{n-1}\}$. The prover then picks a random number *s* from the range and picks other *n*-1 random numbers from the range to generate the $V = \{v_1, v_2,...,v_{n-1}\}$. Then, the prover calculates $U = g^s A_1^{v1} A_2^{v2} ... A_{n-1}^{vn-1}$ and sends *U* to the verifier to initiate the authentication
2. Verifier selects a random number c and sends it to the prover
3. Prover computes $v_p = v_1 \oplus v_2 \oplus ... v_{n-1} \oplus c$ and inserts v_p to the vector *V* such that $V = \{v_1,...,v_p ...,v_{n-1}\}$. The prover also updates $CT = \{Cert_1,..., Cert_p,..., Cert_n-_1\}$ where $Cert_p$ is the prover's certificate. Then, using the prover's private key a_p calculates $r = s\text{-}a_p v_p$ and sends *r*, *V*, *CT* to the verifier
4. After verifying the authenticity of the certificates in CT, the verifier calculates *c* such that it is $\oplus$ of each value in *V*. If $c \neq c'$, terminates the authentication session. Otherwise calculates $U' = g^r A_1^{v1} A_2^{v2} ... A_n^{vn}$. If $U = U'$, authentication is successful. Otherwise, terminates the authentication

Security Justification

- Anonymity: The only information the protocol reveals is that the prover knows a_p. The protocol hides that the A_p (public key) corresponds to that a_p among the set of public keys. Identifying the exact public key of the prover is not possible. Therefore, the protocol obtains k-anonymity
- Completeness: If the prover possesses the correct a_p (the secret key corresponding to A_p), then the prover will be able to generate r such that the *U* generated by the verifier will be equal to the *U* received by the verifier at step 1. It can be illustrated as follows:

$$U' = g^r A_1^{v_1} \dots A_2^{v_p} \dots A_{n-1}^{v_{n-1}}$$
$$U' = g^{(s-a_p v_p)} A_1^{v_1} \dots A_p^{v_p} \dots A_{n-1}^{v_{n-1}}$$
$$U' = g^s g^{-a_p v_p} A_1 v_1 \dots \left(g^{a_p}\right)^{v_p} \dots A_{n-1}^{v_{n-1}}$$
$$U' = g^s g^{-a_p v_p} A_1^{v_1} \dots g^{a_p v_p} \dots A_{n-1}^{v_{n-1}}$$
$$U' = g^s A_1^{v_1} \dots A_{n-1}^{v_{n-1}}$$
$$U' = U$$

- Soundness: Let’s consider a cheating prover as a prover who does not possess a private key a_p corresponding to a public key A_p. Without the a_p, a prover will not be able to generate $r = s\text{-}a_p v_p$. In step 3, the prover is required to generate v_p by XORing elements of V with the challenge *c*. This operation ensures that XORing elements in the *V* vector (including v_p) at the verifier side would generate *c*. Therefore, to pass the first step of verification V must be well-formed. Without the knowledge of the valid a_p, a prover will not be able to generate r to cancel out the g^{apvp} component at the last step of the verification. The only possibility is random guessing, in which the probability is negligible
- Impersonation: As we explained previously, a protocol that accomplishes soundness and completeness is secure against impersonation attacks. Therefore, this protocol is secure against impersonation
- Replay attacks: Let's assume a scenario where a malicious user *M* is eavesdropping on a communication session. *M* can save Msg 1 at step 1 and Msg 3 at step 3 and replay the messages later hoping to authenticate themself maliciously

When M replays Msg 1 verifier will respond with a random challenge. Without the knowledge of s, a_p, V, and P vectors, M will not able to continue further. Therefore, only replaying Msg 1 will not result in a successful attack. Replaying Msg 3 as the response for the challenge will cause the first step of the verification to fail. Since c is chosen randomly by the verifier, the old v_p will not correspond to the new c. Therefore, XORing elements of V will not be equal to c and the verifier will terminate the authentication process. This will only be successful if the same c is chosen at the two authentication processes, in which the probability is negligible.

Modifying the Msg 3 will not gain any advantage to M. As mentioned under soundness proof, without a valid a_p, authenticating will not be possible.

Practical Information

Details of performance analysis are given on the project page, and the source code is in the Git repository.

Conclusion

We have proposed three protocols to achieve anonymous authentication in P2P networks. Firstly, we propose a protocol that utilizes already implemented ring signatures to obtain anonymous authentication. Secondly, we propose a protocol that utilizes a secret sharing mechanism to obtain anonymous authentication. However, this protocol does not provide the zero-knowledge property. In other words, a verifier can obtain some knowledge about the prover's identity. To overcome this issue, we thirdly introduce a protocol based on the zero-knowledge proofs, that utilizes Schnorr's protocol to achieve anonymous authentication. We have justified the security of each protocol in terms of anonymity, completeness, soundness, resilience to impersonation, and resilience to replay attacks. This is a different set of techniques than the other blockchain-based ones (Alilwit, 2020; Asif *et al*., 2022; Umoren *et al*, 2022) as ours mostly pertains to decentralized fault-tolerant networks although they do share some similarities regarding forgery resistance.

As for future work, there are several things to be done. It is worthwhile to implement the proposed protocols and test them against the attack scenarios. Moreover, modifying the proposed protocols for certificate revocation and integrating them into real-world P2P transactions would be a useful project.

Acknowledgment

The authors acknowledge Dr. Douglas Stebila of the University of Waterloo, Canada for his valuable comments.

Funding Information

The authors acknowledge Rabdan Academy, UAE for Open Access to publication support.

Author's Contributions

Pasan Tennakoon and Supipi Karunathilaka: Research work, write the paper.

Janaka Alawatugoda: Supervised the project, guiding the paper write.

Rishikeshan Lavakumar: Project proposal, Research work, coded.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Ahmed, R., & Boutaba, R. (2010). A survey of distributed search techniques in large scale distributed systems. *IEEE Communications Surveys & Tutorials*, *13*(2), 150-167. https://ieeexplore.ieee.org/abstract/document/5473882

Alawatugoda, J. (2017). Generic construction of an eCK-secure key exchange protocol in the standard model. *International Journal of Information Security*, *16*(5), 541-557. https://doi.org/10.1007/s10207-016-0346-9

Alilwit, N. (2020). Authentication based on blockchain. Doctoral Dissertations and Master's Theses, EmbryRiddle Aeronautical University, (548).

Asif, M., Aziz, Z., Bin Ahmad, M., Khalid, A., Waris, H. A., & Gilani, A. (2022). Blockchain-Based Authentication and Trust Management Mechanism for Smart Cities. *Sensors*, *22*(7), 2604. https://doi.org/10.3390/s22072604

Backx, P., Wauters, T., Dhoedt, B., & Demeester, P. (2002, October). A comparison of peer-to-peer architectures. In *Eurescom Summit* (Vol. 2). Citeseer.

Beverly Yang, B., & Garcia-Molina, H. (2003). Design a super-peer network. Data Engineering, 2003. Proceedings. 19th International Conference on, p, 49-60.

Bob, A., David, A., Greg, B., & Fred, E. (2005). The PGP trust model.

Bresson, E., Stern, J., & Szydlo, M. (2002, August). Threshold ring signatures and applications to ad-hoc groups. In *Annual International Cryptology Conference* (pp. 465-480). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/3-540-45708-9_30

Cramer, R., & Damgård, I. (1997, May). Linear zero-knowledge-A note on efficient zero-knowledge proofs and arguments. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing* (pp. 436-445). https://dl.acm.org/doi/pdf/10.1145/258533.258635

Dingledine, R., Mathewson, N., & Syverson, P. (2004). *Tor: The second-generation onion router*. Naval Research Lab Washington DC. https://apps.dtic.mil/sti/citations/ADA465464

Feige, U., Fiat, A., & Shamir, A. (1988). Zero-knowledge proofs of identity. *Journal of cryptology*, *1*(2), 77-94. https://link.springer.com/article/10.1007/BF02351717

Gokhale, S., & Dasgupta, P. (2003). Distributed authentication for peer-to-peer networks. *2003* Symposium on Applications and the Internet Workshops, 2003. p, 347- 353.

Goldwasser, S., Micali, S., & Rackoff, C. (2019). The knowledge complexity of interactive proof-systems. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali* (pp. 203-225). https://dl.acm.org/doi/pdf/10.1145/3335741.3335750

Han, M., Yin, Z., Cheng, P., Zhang, X., & Ma, S. (2020). Zero-knowledge identity authentication for internet of vehicles: Improvement and application. *PloS one*, *15*(9), e0239043. https://doi.org/10.1371/journal.pone.0239043

Jardine, E. (2015). The Dark Web dilemma: Tor, anonymity and online policing. *Global Commission on Internet Governance Paper Series*, (21). https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2667711

Stadler, M., Piveteau, J. M., & Camenisch, J. (1995, May). Fair blind signatures. In *International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 209-219). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/3-540-49264-X_17

Sahai, A., & Vadhan, S. (2000). A Complete Problem for Statistical Zero Knowledge. *Journal of the ACM*. 50(2), 196–249. http://web.cs.ucla.edu/~sahai/work/web/2003%20Publications/J.ACM2003.pdf

Josephson, W. K., Sirer, E. G., & Schneider, F. B. (2004, February). Peer-to-peer authentication with a distributed single sign-on service. In *International Workshop on Peer-to-Peer Systems* (pp. 250-258). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/978-3-540-30183-7_24

Kamvar, S. D., Schlosser, M. T., & Garcia-Molina, H. (2003, May). The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web* (pp. 640-651). https://dl.acm.org/doi/abs/10.1145/775152.775242

Karaarslan, E., & Adiguzel, E. (2018). Blockchain based DNS and PKI solutions. *IEEE Communications Standards Magazine*, *2*(3), 52-57. https://ieeexplore.ieee.org/abstract/document/8515149

Lee, S., Sherwood, R., & Bhattacharjee, B. (2003, March). Cooperative peer groups in NICE. In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)* (Vol. 2, pp. 1272-1282). IEEE. https://ieeexplore.ieee.org/abstract/document/1208963

Liu, D. Y., Liu, J. K., Mu, Y., Susilo, W., & Wong, D. S. (2007). Revocable ring signature. *Journal of Computer Science and Technology*, *22*(6), 785-794. https://link.springer.com/article/10.1007/s11390-007-9096-5

Liu, J. K., & Wong, D. S. (2005, May). Linkable ring signatures: Security models and new schemes. In *International Conference on Computational Science and Its Applications* (pp. 614-623). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/11424826_65

Lu, L., Han, J., Hu, L., Huai, J., Liu, Y., & Ni, L. M. (2007, March). Pseudo trust: Zero-knowledge based authentication in anonymous peer-to-peer protocols. In *2007 IEEE International Parallel and Distributed Processing Symposium* (pp. 1-10). IEEE. https://ieeexplore.ieee.org/abstract/document/4228012

Nakamoto, S. (2009). Bitcoin: A peer-to-peer electronic cash system. cryptography mailing list.

Oh, B. T., Lee, S. B., & Park, H. J. (2008, February). A peer mutual authentication method using PKI on super peer-based peer-to-peer systems. In *2008 10th International Conference on Advanced Communication Technology* (Vol. 3, pp. 2221-2225). IEEE. https://ieeexplore.ieee.org/abstract/document/4494231/

Orman, H. (2018). Blockchain: The emperor’s new PKI? *IEEE Internet Computing*, *22*(2), 23-28. https://ieeexplore.ieee.org/abstract/document/8345567/

Papageorgiou, A., Mygiakis, A., Loupos, K., & Krousarlis, T. (2020, June). DPKI: A blockchain-based decentralized public key infrastructure system. In *2020 Global Internet of Things Summit (GIoTS)* (pp. 1-5). IEEE. https://ieeexplore.ieee.org/abstract/document/9119673

Rivest, R. L., Shamir, A., & Tauman, Y. (2001, December). How to leak a secret. In *International conference on the theory and application of cryptology and information security* (pp. 552-565). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/3-540-45682-1_32

Sabater, J., & Sierra, C. (2002, July). Reputation and social network analysis in multi-agent systems. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: Part 1* (pp. 475-482). https://doi.org/10.1145/544741.544854

Shamir, A. (1979). How to share a secret. https://doi.org/10.1145/359168.359176

Sivakumar, P., & Singh, K. (2017). Privacy based decentralized public key infrastructure (PKI) implementation using smart contract in blockchain. *National Institute of Technology*, 6.

Tang, C., Liu, Z., & Liu, J. (2003). The Statistical Zero-knowledge Proof for Blum Integer Based on Discrete Logarithm. *Cryptology ePrint Archive*. https://eprint.iacr.org/2003/232

Tsang, P. P., & Smith, S. W. (2008, June). PPAA: Peer-to-peer anonymous authentication. In *International Conference on Applied Cryptography and Network Security* (pp. 55-74). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/978-3-540-68914-0_4

Umoren, O., Singh, R., Awan, S., Pervez, Z., & Dahal, K. (2022). Blockchain-Based Secure Authentication with Improved Performance for Fog Computing. *Sensors*, *22*(22), 8969. https://doi.org/10.3390/s22228969

Wu, H., & Wang, F. (2014). A survey of noninteractive zero knowledge proof system and its applications. *The Scientific World Journal*, *2014*. https://doi.org/10.1155/2014/560484

Wallach, D. S. (2002, November). A survey of peer-to-peer security issues. In *International symposium on software security* (pp. 42-57). Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/3-540-36532-X_4

Wang, X., Sun, X., Sun, G., & Luo, D. (2010, May). CST: P2P anonymous authentication system based on collaboration signature. In *2010 5th International Conference on Future Information Technology* (pp. 1-7). IEEE. https://ieeexplore.ieee.org/abstract/document/5482740

Wang, X., & Sun, X. (2009). Fair blind signature-based authentication for super peer P2P network. *Information Technology Journal*, *8*(6), 887-894.

Xiong, L., & Liu, L. (2003, June). A reputation-based trust model for peer-to-peer e-commerce communities. In *EEE International Conference on E-Commerce, 2003. CEC 2003*. (pp. 275-284). IEEE. https://ieeexplore.ieee.org/abstract/document/1210262

Yakubov, A., Shbair, W., Wallbom, A., & Sanda, D. (2018). A blockchain-based PKI management framework. In *The First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block) colocated with IEEE/IFIP NOMS 2018, Tapei, Tawain 23-27 April 2018*. https://orbilu.uni.lu/handle/10993/35468