Anonymous and Distributed Authentication for Peer-to-Peer Networks

Pasan Tennakoon[1] Supipi Karunathilaka[2]
Rishikeshan Lavakumar[3] Janaka Alawatugoda[4]

Department of Computer Engineering, Faculty of Engineering, University of Peradeniya, Peradeniya 20400, Sri Lanka

[1]pasan96tennakoon@gmail.com, [2]supipivirajini@gmail.com, [3]public@ris.rocks, [4]alawatugoda@eng.pdn.ac.lk

Abstract

Well-known authentication mechanisms such as Public-key Infrastructure (PKI) and Identity-based Public-key Certificates (ID-PKC) are not suitable to integrate with the peer-to-peer (P2P) network environment. The reason is the difficulty in maintaining a centralized authority to manage the certificates. The authentication becomes even harder in an anonymous environment. We present three authentication protocols such that the users can authenticate themselves in an anonymous P2P network, without revealing their identities. Firstly, we propose a way to use existing ring signature schemes to obtain anonymous authentication. Secondly, we propose an anonymous authentication scheme utilizing secret sharing schemes. Finally, we propose a zero-knowledge-based anonymous authentication protocol. We provide security justifications of the three protocols in terms of anonymity, completeness, soundness, resilience to impersonation attacks, and resilience to replay attacks.

Keywords: Anonymous authentication, Peer-to-peer networks, Ring signatures, Secret sharing, Zero knowledge

1 Introduction

The concept of Peer-to-peer (P2P) communication has gained significant attention in the network community over the years. Since the release of Napster in 1998 many P2P applications have been introduced. Bitcoin [24], TOR [10], Freenet [8], etc, are some of the more popular P2P applications. The absence of centralized authority is the main reason behind the popularity of P2P applications. This eliminates the need for an expensive central server 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 the 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 certificates, because the availability of the peers cannot be guaranteed. Therefore, many such networks focus on providing user anonymity rather than authentication. The reduced security of these networks opens up vulnerabilities [35]. The anonymity feature of these networks has created a safe house for cybercriminals [15]. Being unaccountable for their actions, P2P users have the freedom to misbehave. This can cause harm to the network as well as its users. Accountability can be achieved through authentication. In order to integrate an authentication mechanism into an anonymous P2P environment, we need to solve two main challenges;

This article is based on the final year undergraduate research project carried out by Pasan, Supipi, and Rishikeshan under the supervision of Janaka, at the Department of Computer Engineering, University of Peradeniya, Sri Lanka. The authors acknowledge Douglas Stebila for his valuable comments.
Project page: https://cepdnaclk.github.io/e15-4yp-anonymous-authentication
Git repository: https://github.com/cepdnaclk/e15-4yp-anonymous-authentication

1. authenticate in a decentralized environment, and
2. authenticate without revealing identity.

The above two challenges have been discussed since the start 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, be secure against misbehaving parties (malicious verifiers and provers), etc.

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 of the three protocols in terms of anonymity, completeness, soundness, resilience to impersonation attacks, and resilience to replay attacks.

Thus, our contribution through this work is to notify the identified challenges arising when integrating an authentication mechanism into an anonymous P2P environment, and propose three approaches for anonymous authentication in P2P networks with security justifications.

2 Related Works

2.1 Authentication in P2P

Absence of a central server makes authentication in P2P networks complex. PKI or ID-PKC are based on a trusted third party. Establishing a trusted third party in a semi-trusted network like P2P is a problematic task. Many P2P networks propose trust and reputation management schemes to solve this problem. Some works [12, 37, 34] 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 the other peers [18, 20, 29, 39]. P2P systems that use reputation managements 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 researches suggest that using a modified PKI for authentication in P2P networks [25, 17]. Rather than having a single centralized authority, it's responsibility is distributed across multiple peers in the network. This improves the scalability and robustness of the authentication process. The downside of using modified PKI in P2P is certificate management becomes complex. As a solution, Josephson et al. [17] uses a set of peers as Authentication Servers (ASs). 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, modern authentication schemes utilize blockchain technology [27, 19, 40, 26]. Blockchain can make the process of a CA in distributed, immutable and transparent manner. Therefore, can successfully solve the problems of malicious CAs, MITM attacks and single point of failure. Blockchain is used as a distributed key-value data storage. The data is public and readable to everyone. Sivakumar and Singh [1] propose the idea of using smart contracts to certificate management. The decentralized PKI is secure as long as honest nodes control collectively more than 51% of computing power. Moreover, some argues the need of blockchain to decentralized PKI, since the technology of blockchain is still new to the industry.

The PGP Web of Trust (WoT) [6] is another way to navigate the problem of not having a trusted central authority. WoT distribute the responsibility of a CA among users. The core concept of WoT is trust chains. For a simpler explanation, assume A wants to authenticate himself to B. There is a user C who trusts B. 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 A's certificate is authentic. Using indirect trust chains WoT creates a community of trusted users. However, WoT is not suitable for P2P networks, because it is difficult for a new peer to join the network without personally knowing a existing user of the network.

2.2 Anonymous Authentication in P2P

The concept of anonymous authentication has been around for sometime. Pseudo Trust (PT) [23] has been one of the more popular work under this topic. The PT utilizes the concept of double pseudonyms combine with zero knowledge proofs to authenticate users anonymously. The PT also uses onion routing [10] and EigenTrust [18] trust managements 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. The PT does not change the pseudo identity (PI) prior to each authentication process. The PT protocol requires 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. [14] presents a similar authentication scheme to the PT for Internet of Vehicles (IoV), that also suffers from the same vulnerabilities as in the PT.

Tsang and Smith [34] presents an interesting approach to anonymous authentication; P2P Anonymous Authentication (PPAA) uses tags to obtain the 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 into 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 random oracle model (ROM).

Wang et al. [37] uses collaboration signature trust (CST) to authenticate users anonymously. However, this mechanism is not safe in a semi-trusted environment such as P2P networks. Wang and Sun [36] presents a similar method as to the CST; they use fair blind signature trust (FBST) [16] to present novel authentication scheme that keep 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 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 [36] uses the concept of secret sharing [31] to reduce the vulnerability of exposed RPs. Shamir [31] presents a way to break a key into number of parts and store it in multiple places, and then recreate the key when required. Wang and Sun [36] uses this technique to break the key (link between ID and pseudo-ID) and store it among multiple RPs. Therefore, even if few RPs get compromised it does not reveal the user's identity. Further, a user uses anonymous multi-cast to communicate with a SP. This makes it impossible for a SP to reveal the identity of a user.

3 Cryptographic Preliminaries

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

3.1 Ring Signatures

The notion of ring signatures was firstly introduced by Rivest et al. [28]. 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 [7], linkable ring signatures [22], revocable ring signatures [21], etc.

Let there be a group of k number of entities where each entity $i \in \{1, \ldots, k\}$ has a public key P_i and a corresponding secret key S_i. An entity $r \in \{1, \ldots, k\}$ (with the public key P_r and the corresponding secret key S_r) can generate a ring signature on a message m using $(m, P_1, \ldots, P_k, S_r)$. Anyone with the knowledge of $m, P_1, \ldots, P_k$ 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.

3.2 Secret Sharing Schemes

In 1979 Shamir introduced the concept of secret sharing [31]. This allows a secret to be divided into n parts. The secret can be reconstructed with at least t parts where $(1 \leq t \leq n)$. No knowledge about the secret can be learnt 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 $(a_1, a_2, \ldots, a_{t-1})$. Then, set a_0 to the secret we want to share. These points are used to generate the polynomial $f(x)$.

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{t-1}x^{t-1}$$

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 [31] 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 P2P environment is a semi-trusted environment. For an example, when a peer requests a key from another peer, it 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 [37, 36].

3.3 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 firstly introduced by Goldwasser et al. [13]. Since then, many different ZKPs have been presented [33, 11, 9, 30]. 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 [38].

4 Network Design

In this section we detail the network design; conceptual design and the distributed certificate management in regard to our work.

4.1 Conceptual Design

We employ a hybrid P2P network [5]. A traditional hybrid P2P network consists of peers and super peers. Hybrid P2P systems is 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 [4].

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 the ordinary service requesters. They are connected to the system through the 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 among other super peers using the super peer network. Super peers can join or leave the network at any time. Dynamic behaviour 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 behaviour of peers and maintaining connectivity among the existing super peers. A super peer is the only responsible entity for the nodes under it's scope, and does not know any information regarding other peers of the system. Therefore, node discovery process becomes an exhaustive task. This can be accomplished in two ways; flooding search or random walk. We utilize flooding search in this project since the random walk is not guaranteed to produce the results [2].

4.2 Distributed Certificate Management

The decentralized nature of the P2P networks makes it difficult to integrate the 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 [31].

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 the Shamir's algorithm. The super peer then floods the parts across the network. Once a certificate recreation request 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 distributing certificates in a dynamic way. 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 a constant will increase the average key storage size in the super peers.

5 Authentication Schemes

In this section we discuss the details about the authentication schemes that we propose.

5.1 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. [21], 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 is able to 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. User picks a random number r_u and generates the private key S_u using a hash function `H1` such that $S_u = \texttt{H1}(ID, r_u)$. 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 $n - 1$ certificates from the set of k certificates. After verifying the authenticity of the selected certificates, the prover generates $CT = \{Cert_1, Cert_2, \ldots, Cert_n\}$, which includes the prover's certificate $Cert_p$ as well (totally n number of certificates now). Then, the prover obtains each corresponding public key from the certificates to generate $P = \{P_1, P_2, \ldots, P_n\}$. After that, encrypts CT with 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 $H = \texttt{Hash}(P_1, P_2, \ldots, P_n)$. Then, sends H and a random nonce N to the prover.

3. Prover generate $H' = \texttt{Hash}(P)$ and if $H \neq H'$ terminates the authentication. Otherwise, uses his secret key S_p, P and P_s to sign N and generates ring signature σ using ring signature scheme of Liu et al. [21]. Then, encrypts σ and N with verifier's public key P_v, and sends to the verifier.

4. The verifier decrypts the message to obtain σ and N. Then, verify 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:** Anonymity of the protocol depends on the properties of the ring signature scheme. The scheme proves it obtains signer anonymity. The proposed protocol does not reveal any information other than the set of public keys P. The only information verifier can deduce is prover's public key is among the set P. Therefore, our protocol obtains k-anonymity.
- **Completeness:** If a protocol has the completeness property, the protocol is said to be comprehensive; an honest verifier will always be able to authenticate himself. The completeness property of the protocol comes from the underlying ring signature scheme of Liu et al. [21]. Therefore, our protocol satisfies completeness property.
- **Soundness:** If a protocol has the soundness property, the protocol is said to be truthful; a cheating prover will never be able to authenticate himself. Since the underlying ring signature scheme satisfies the unforgeability property, a cheating prover is unable to forge. Therefore, our protocol satisfies 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 himself maliciously. This is known as reply attack. Let's assume a scenario where a malicious user M is eavesdropping on an authentication session. M can save the message $Msg1$ in the step 1 (of the Authentication process of the protocol) and message $Msg3$ in the step 3, replay them later hoping to authenticate himself maliciously.

  In our protocol, $Msg1$ is encrypted. Therefore, M will not be able to reveal its content. When $Msg1$ 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, he will not be able to authenticate himself. Replaying $Msg3$ will not gain anything unless the verifier generates the same N as the original authentication.

5.2 Authenticated Secret Sharing Approach

The basic idea of this approach is to present prover a set of public keys and challenge to prove the knowledge of at least one secret key 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 himself. To accomplish this, we adopt an authenticated key exchange protocol of Alawatugoda [3]. The protocol is explained below:

Registration

1. A user has an ID which can be anything related to the identity of the user. User picks a random number r_u and using a hash function $\mathtt{H1}$ generates $S_u = \mathtt{H1}(ID, r_u)$. 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. 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 $CT = \{Cert_1, Cert_2, \ldots, Cert_n\}$, 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. Verifier decrypts the message using his secret key S_v and generates $H = \texttt{Hash}(CT)$ using a hash function `Hash`. Then, picks a random number x and computes $X = g^x$. After that, generates n ciphertexts $\{C_1, C_2, \ldots, C_n\} = C$ 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 corresponds to his public key, decrypts it using his secret key S_p to obtain X and H. Then, generates $H' = \texttt{Hash}(CT)$ 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. 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 $H1 = \texttt{Hash}(R||K)$. Then, sends the encryption of R (that is computed above), and $H1$ to the prover.

5. The prover decrypts the message using K and obtains R. Then, uses R and K to generate $H1' = \texttt{Hash}(R||K)$. If $H1 = H1'$, prover sends R back to the verifier. Otherwise, terminate 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 prover's identity. The verifier will generate a set of $x = \{x_1, x_2, \ldots, x_n\}$ and generates $X = \{X_1, X_2, \ldots, X_n\}|X_i = g^{x_i}$. Then, the verifier can generate $C = \{C_1, C_2, \ldots, C_n\}$. 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 corresponding P_i to reveal provers identity.

  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 himself, 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 $K = \{K_1, K_2, \ldots, K_n\}$, where each K_i correspond 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 $H1$ 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 terminate 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 an example, the verifier generates C as half of the C_is are 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 misformed 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 the step 5. Since the prover generates the correct key K, he can successfully decrypt the encrypted R. Therefore, can successfully authenticate himself.

- **Soundness:** A cheating prover does not have a secret key corresponding to any of the public keys in P. To authenticate himself as a member he has to correctly guess X at the step 3 of the protocol or correctly guess R at the step 5 of the protool. 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 himself.

- **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 $Msg1$ in the step 1 of the protocol, message $Msg3$ in the step 3 of the protocol and/or message $Msg5$ in the step 5 of the protocol, replay it later hoping to authenticate himself.

  If $Msg1$ was replayed this will not gain any advantage for M. Since M does not know any secret key corresponding to the set P, he will not be able to authenticate unless by correctly guessing X or R. Storing $Msg3$ 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 $Msg5$ to successfully authenticate himself as a valid prover.

5.3 Zero Knowledge Proof Approach

ZKP is a popular approach to obtain anonymous authentication in P2P networks. This technique has been utilized in many research works [23, 14, 34]. 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 an modification of a non-interactive zero knowledge proof protocol [32].

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 $a_u = \mathtt{H1}(ID, r_u)$ using a hash function $\mathtt{H1}$. Using a_u as the private key of the user, the public key A_u is computed as $A_u = g^{a_u}$. Then, the user sends the registration request along with his ID, A_u to the main server.
2. 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, prover generates $CT = \{Cert_1, Cert_2, \ldots, Cert_{n-1}\}$. Then, the prover obtains each corresponding public keys from the certificates to generate $P = \{A_1, A_2, \ldots, A_{n-1}\}$. The prover then picks a random number s from the range and picks another $n-1$ random numbers from the range to generate the $V = \{v_1, v_2, \ldots, v_{n-1}\}$. Then, the prover calculates $U = g^s {A_1}^{v_1} {A_2}^{v_2} \ldots {A_{n-1}}^{v_{n-1}}$, 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 $v_p = v_1 \oplus v_2 \oplus \ldots v_{n-1} \oplus c$ and inserts v_p to the vector V such that $V = \{v_1, \ldots, v_p \ldots, v_{n-1}\}$. The prover also updates $CT = \{Cert_1, \ldots, Cert_p, \ldots, Cert_{n-1}\}$ where $Cert_p$ is prover's certificate. Then, using the prover's private key a_p, calculates $r = s - a_p v_p$, and sends r, V, CT to the verifier.
4. After verifying the authenticity of the certificates in CT, the verifier calculates c' such that $\oplus$ of the each value in V. If $c \neq c'$, terminate the authentication session. Otherwise calculates $U' = g^r {A_1}^{v_1} {A_2}^{v_2} \ldots {A_n}^{v_n}$. If $U = U'$, authentication is successful. Otherwise, terminates the authentication.

Security Justification

- **Anonymity:** The only information the protocol reveals is that the prover has the knowledge of a_p. The protocol hides 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 to the verifier at the step 1. It can be illustrated as follows;

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

- **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 $r = s - a_p v_p$. At the 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 $g^{a_p v_p}$ 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 $Msg1$ at the step 1 and $Msg3$ at the step 3, and replay the messages later hoping to authenticate himself maliciously.

  When M replays $Msg1$ 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, the only replaying $Msg1$ will not be successful. Replaying $Msg3$ 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 $Msg3$ will not gain any advantage to M. As mentioned under soundness proof, without a valid a_p authenticating will not be possible.

5.4 Practical Information

Details of performance analysis is given in the `project page`, and the souse code is in the `Git repository`.

6 Conclusions and Future Works

We have proposed three protocols to achieve anonymous authentication in the 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.

As for future works, 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 in the real-world P2P transactions would be a useful project.

References

[1] Privacy based decentralized Public Key Infrastructure implementation using Smart contract in Blockchain. *2nd Advanced Workshop on Blockchain: Technology, Applications, Challenges*, 2(1):1–6, 2017.

[2] Reaz Ahmed and Raouf Boutaba. A survey of distributed search techniques in large scale distributed systems. *IEEE Communications Surveys and Tutorials*, 13(2):150–167, 2011.

[3] Janaka Alawatugoda. Generic construction of an eck-secure key exchange protocol in the standard model. *Int. J. Inf. Secur.*, 16(5):541–557, October 2017.

[4] Peter Backx, Tim Wauters, Bart Dhoedt, and Piet Demeester. A comparison of peer-to-peer architectures A comparison of peer-to-peer architectures. (January), 2002.

[5] B Beverly Yang and Hector Garcia-Molina. Design a super-peer networks.pdf. *Data Engineering, 2003. Proceedings. 19th International Conference on*, pages 49–60, 2003.

[6] Alice Bob, A. David, B. Greg, and Eric Fred. The pgp trust model. 2005.

[7] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. Threshold ring signatures and applications to ad-hoc groups. volume 2442, pages 465–480, 08 2002.

[8] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *International Workshop On Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability*, pages 46–66. Springer-Verlag New York, Inc., 2001.

[9] Ronald Cramer and Ivan Damgård. Linear zero-knowledge – a note on efficient zero-knowledge proofs and arguments. volume 3, pages 436–445, 05 1997.

[10] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. *Paul Syverson*, 13, 06 2004.

[11] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1:77–94, 06 1988.

[12] S. Gokhale and Partha Dasgupta. Distributed authentication for peer-to-peer networks. pages 347–353, 02 2003.

[13] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. *The knowledge complexity of interactive proof-systems.* 10 2019.

[14] Mu Han, Zhikun Yin, Pengzhou Cheng, Xing Zhang, and Shidian Ma. Zero-knowledge identity authentication for internet of vehicles: Improvement and application. *PLoS ONE*, 15, 9 2020.

[15] Eric Jardine. The dark web dilemma: Tor, anonymity and online policing. *Global Commission on Internet Governance Paper Series*, page 24, 09 2015.

[16] Piveteau Jean-Marc. Fair blind signatures, 2000.

[17] William Josephson, Emin Sirer, and Fred Schneider. Peer-to-peer authentication with a. 03 2004.

[18] Sepandar Kamvar, Mario Schlosser, and Hector Garcia-molina. The eigentrust algorithm for reputation management in p2p networks. *The EigenTrust Algorithm for Reputation Management in P2P Networks*, 04 2003.

[19] Enis Karaarslan and Eylul Adiguzel. Blockchain based dns and pki solutions. *IEEE Communications Standards Magazine*, 2:52–57, 09 2018.

[20] Seungjoon Lee, Rob Sherwood, and Bobby Bhattacharjee. Cooperative peer groups in nice. volume 50, pages 1272 – 1282 vol.2, 03 2006.

[21] Dennis Liu, Joseph Liu, Yi Mu, Willy Susilo, and Duncan Wong. Revocable ring signature. *J. Comput. Sci. Technol.*, 22:785–794, 11 2007.

[22] Joseph Liu and Duncan Wong. Linkable ring signatures: Security models and new schemes. volume 3481, pages 614–623, 05 2005.

[23] Li Lu, Jinsong Han, Lei Hu, Jinpeng Huai, Yunhao Liu, and Lionel Ni. Pseudo trust: Zero-knowledge based authentication in anonymous peer-to-peer protocols. pages 1 – 10, 04 2007.

[24] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Cryptography Mailing list at https://metzdowd.com*, 03 2009.

[25] Byeong-Thaek Oh, Sang-Bong Lee, and Ho-Jin Park. A peer mutual authentication method using pki on super peer based peer-to-peer systems. volume 3, pages 2221 – 2225, 03 2008.

[26] Hilarie Orman. Blockchain: The emperors new pki? *IEEE Internet Computing*, 22:23–28, 03 2018.

[27] Alexander Papageorgiou, Antonis Mygiakis, Konstantinos Loupos, and Thomas Krousarlis. Dpki: A blockchain-based decentralized public key infrastructure system. pages 1–5, 06 2020.

[28] Ronald Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. volume 2248, pages 552–565, 07 2001.

[29] Jordi Sabater-Mir and Carles Sierra. Reputation and social network analysis in multi-agent systems. pages 475–482, 01 2002.

[30] Amit Sahai and Salil Vadhan. A complete problem for statistical zero knowledge. *IACR Cryptology ePrint Archive*, 2000:56, 01 2000.

[31] Adi Shamir. How to share a secret, 1979.

[32] Anna Paula Soares. Schnorr Non-interactive Zero-Knowledge Proof. *Journal of Chemical Information and Modeling*, 53(9):1689–1699, 2013.

[33] Chunming Tang, Zhuojun Liu, and Jinwang Liu. The statistical zero-knowledge proof for blum integer based on discrete logarithm. 12 2003.

[34] Patrick Tsang and Sean Smith. Ppaa: Peer-to-peer anonymous authentication. pages 55–74, 06 2008.

[35] Dan Wallach. A survey of peer-to-peer security issues. volume 2609, pages 42–57, 01 2002.

[36] Xiaoliang Wang and Xingming Sun. Fair blind signature based authentication for super peer p2p network(2). *Information Technology Journal*, 8:887–894, 2009.

[37] Xiaoliang Wang, Sun Xingming, Guang Sun, and Dong Luo. Cst: P2p anonymous authentication system based on collaboration signature. *2010 5th International Conference on Future Information Technology, FutureTech 2010 - Proceedings*, 01 2010.

[38] Huixin Wu and Feng Wang. A survey of noninteractive zero knowledge proof system and its applications. *TheScientificWorldJournal*, 2014:560484, 05 2014.

[39] Li Xiong and Ling Liu. A reputation-based trust model for peer-to-peer ecommerce communities. pages 275 – 284, 07 2003.

[40] Alexander Yakubov, Wazen Shbair, Anders Wallbom, David Sanda, and Radu State. A blockchain-based pki management framework. 04 2018.